



DOCTORAL THESIS

ASSESSMENT OF CYBERSECURITY BASED ON RISK AND UNCERTAINTY PROPAGATION IN DISTRIBUTED NETWORKED SYSTEMS

by Dipl.-Ing. Michael Krisper BSc

DOCTORAL THESIS

to achieve the university degree of
Doktor der technischen Wissenschaften

submitted to

Graz University of Technology

and conducted at the

Institute of Technical Informatics

Supervisor:

Univ.-Prof. Dipl.-Inform. Dr.sc.ETH Kay Uwe Römer

Advisors:

Dipl.-Ing. Dr.techn. Georg Macher

Dipl.-Ing. Dr.techn. Christian Kreiner

External Examiner:

Prof. Indrajit Ray, Colorado State University

Graz, May 2021

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present doctoral thesis.

Date

Signature

Acknowledgements

Working on this dissertation has been a long, bumpy road over many years, requiring full commitment and perseverance. It was a seemingly endless turmoil of literature research, writing articles, discussions, conferences, project work, teaching, and administrative work, including some wrong turns and dead ends. I am incredibly proud that I was finally able to complete the dissertation and would like to thank everyone who accompanied me along the way. Writing this thesis would not have been possible without the support of my wife, friends, colleagues and the many other inspiring and helpful people I have met in my life.

First and foremost, I would like to thank my wife, **Melanie Krisper**, for being endlessly patient with me during the many days, nights, weekends, months, and years I spent at the institute. Especially during stressful times, she supported, encouraged and spurred me on to continue working on the dissertation. Thank you for the love you gave me during those stressful times.

Next, I would like to thank my parents, **Renate Krisper** and **Hermann Krisper**, for their support during my prolonged studies. They were the most loving parents a person could imagine. I want to thank **Otto Platzer** for always being a close family friend and supporting us after the sad loss of my father. I also want to thank my brother **Martin Krisper** for taking care of my mother and the family home during the times when I could not.

Without him, I would not have the opportunity of writing a dissertation, and for that, I am infinitely grateful: **Christian Kreiner**. We shared the enthusiasm and passion for design patterns, critical thinking and insightful discussions. Tragically, he also passed away far too early and could not see the end of my dissertation. He was one of the most inspiring teacher and human I have ever met. Besides him, I would also like to thank my teachers and role models from whom I learned to think critically and humanistic, to be tolerant and open-minded towards other opinions, and to be interested in new knowledge and respectful to others: **Johann Götschl**, **Enrique Grabl**, **Wolfgang Woess**, **Christian Magele**, **Johann Hagauer**, **Reinhard Kamitz**, **Manfred Schantl**, **Manfred Wilfing**, and **Roswitha Fink**. Also, thank you, **Taz Daughtrey**, your interest in my work has motivated me to keep going.

I want to thank all colleagues at the institute for the many enjoyable hours we had together. The academic discussions in the clubroom were the best, and for that, I would like to thank the “ELITI Boys and Girls”. Thank you for quenching my thirst (for knowledge, of course): **Michael Spörk**, the man with the strongest forearm in the institute; **Rainer Hofmann**, who has the finest taste in music; **Martin Erb**, who seems to multitask effortlessly and never gets distracted; **Lukas Gressl**, who is always up for a good and loud discussion; **Jürgen Dobaj**, the connector of all ideas; **Alexander Rech**, the best-dressed businessman at the institute; **Johannes Iber**, who likes the simple life; **Tobias Rauter**, an exceptional hacker; **Andrea Höller**, who always finds the right words; **Thomas Ulz**, the social driver and one of the most intelligent minds; **Markus Schuss**, the last universal genius and jack-of-all-

trades; **Michael Stocker**, who likes to discuss everything; and **Bernhard Großwindhager**, the man who loves practical things. I also thank **Andreas Sinnhofer**, the mediator; **Felix Oppermann**, the man for the details; and **Ralph Weissnegger**, the savourer for the beautiful things in life.

Furthermore, I thank my friends and colleagues which have been part of my life for so many years now, and went with me through thick and thin: **Anton Sax**, **Hans-Peter Poglitsch**, **Armin Schaberl**, **Markus Schreiner**, **Christian Seidl**, **Markus Harb**, **Teresa and Bernhard Francisci**, **Ulla Mayrhofer & Anton Straka**, and **Sebastian Scheiber**. I hope that we will see each other more often again after this year of separation due to the Corona crisis.

Finally, I would like to thank my supervisors at the Institute of Computer Engineering. Thank you, **Kay Römer**, for your constructive, honest and fair feedback – I can imagine that it is not an easy job to supervise so many students, to keep our anthill of institute together, and still have enough time for everyone to be helpful, constructive, and respectful as you always are. Thank you, **Georg Macher**, for giving me the chance to continue working at the institute and on my dissertation. You taught me to see a bit of sarcasm and fun everywhere and go through the world with open eyes. Thank you, **Christian Steger** and **Eugen Brenner**, for the emotional and organizational support in stressful times and the insightful and interesting discussions at the institute.

*Graz, May 2021
Michael Krisper*

Danksagung

Die Arbeit an dieser Dissertation war ein langer, steiniger Weg über viele Jahre, der vollen Einsatz und Ausdauer erforderte. Es war schier endloses Wirrwarr bestehend aus Literaturrecherchen, Schreiben von Artikeln, Diskussionen, Konferenzen, Projektarbeiten, Lehre und administrativer Arbeit, das auch einige Irrwege und Sackgassen beinhaltet hat. Ich bin außerordentlich stolz, dass ich die Dissertation nach alledem endlich fertigstellen konnte und möchte mich bei allen danken, die mich auf diesem Weg ein Stück weit begleitet haben. Das Schreiben dieser Arbeit wäre ohne die Unterstützung meiner Frau, meiner Freunde, Kollegen und der vielen anderen inspirierenden und hilfreichen Menschen, die ich in meinem Leben getroffen habe, nicht möglich gewesen.

Zuallererst möchte ich meiner Frau **Melanie Krisper** dafür danken, dass sie während der vielen Tage, Nächte, Wochenenden, Monate und Jahre, die ich am Institut verbrachte, unendlich geduldig mit mir war. Vor allem in stressigen Zeiten hat sie mich unterstützt, ermutigt und angespornt, die Arbeit an der Dissertation fortzusetzen. Danke für die Liebe, die du mir während dieser stressigen Zeit gegeben hast.

Als Nächstes möchte ich mich bei meinen Eltern, **Renate Krisper** und **Hermann Krisper**, für die Unterstützung während meines langwierigen Studiums bedanken. Sie waren die liebenswertesten Eltern, die sich ein Mensch vorstellen kann. Ich danke **Otto Platzer**, dass er immer ein enger Freund der Familie war und uns nach dem traurigen Verlust meines Vaters unterstützt hat. Außerdem möchte ich meinem Bruder **Martin Krisper** dafür danken, dass er sich in den Zeiten, in denen ich nicht konnte, um meine Mutter und das Familienhaus gekümmert hat.

Ohne ihn hätte ich nicht die Möglichkeit, eine Dissertation zu schreiben, und dafür bin ich unendlich dankbar: **Christian Kreiner**. Wir teilten die Begeisterung und Leidenschaft für Design Patterns, kritisches Denken und aufschlussreiche Diskussionen. Tragischerweise ist auch er viel zu früh verstorben und konnte das Ende meiner Dissertation nicht mehr miterleben. Er war einer der inspirierendsten Lehrer und Menschen, die ich je getroffen habe. Neben ihm möchte ich auch meinen Lehrern und Vorbildern danken, von denen ich gelernt habe, kritisch und humanistisch zu denken, tolerant und aufgeschlossen gegenüber anderen Meinungen zu sein, sowie interessiert an neuem Wissen und respektvoll im Umgang zu sein: **Johann Götschl**, **Enrique Grabl**, **Wolfgang Woess**, **Christian Magele**, **Johann Hagauer**, **Reinhard Kamitz**, **Manfred Schantl**, **Manfred Wilfing**, und **Roswitha Fink**. Vielen Dank auch an **Taz Daughtrey**, dein Interesse an meiner Arbeit hat mich motiviert, weiterzumachen.

Ich möchte mich bei allen Kolleginnen und Kollegen am Institut für die vielen schönen Stunden bedanken, die wir zusammen hatten. Die akademischen Diskussionen im Clubraum waren die besten, und dafür möchte ich mich bei den "ELITI Boys and Girls" bedanken. Danke, dass ihr meinen Durst (nach Wissen) gestillt habt: **Michael Spörk**, der Mann mit dem stärksten Unterarm im Institut; **Rainer Hofmann**, der den feinsten Musikgeschmack hat; **Martin Erb**, der scheinbar mühelos multitasken kann und sich nie ablenken lässt; **Lukas Gressl**, der immer für eine gute und laute Diskussion zu haben

ist; **Jürgen Dobaj**, der Verbinder aller Ideen; **Alexander Rech**, der bestgekleidetste Geschäftsmann am Institut; **Johannes Iber**, der das einfache Leben mag; **Tobias Rauter**, ein außergewöhnlicher Hacker; **Andrea Höller**, die immer die richtigen Worte findet; **Thomas Ulz**, der soziale Antreiber und einer der intelligentesten Menschen; **Markus Schuss**, das letzte Universalgenie und Tausendsassa; **Michael Stocker**, der gerne über Details diskutiert; und **Bernhard Großwindhager**, der Mann, der die praktischen Dinge liebt. Außerdem danke ich **Andreas Sinnhofer**, dem Vermittler; **Felix Oppermann**, dem Mann für die Details; und **Ralph Weissnegger** dem Genießer der schönen Dinge des Lebens.

Ein großes Danke geht auch an meine Freunde die nun schon so viele Jahre Teil meines Lebens sind und mit mir durch dick und dünn gegangen sind: **Anton Sax**, **Hans-Peter Poglitsch**, **Armin Schaberl**, **Markus Schreiner**, **Christian Seidl**, **Markus Harb**, **Teresa und Bernhard Francisci**, **Ulla Mayrhofer** & **Anton Straka** und **Sebastian Scheiber**. Ich hoffe, dass wir uns nach diesem Jahr der Trennung durch die Corona-Krise wieder öfter sehen werden.

Abschließend möchte ich mich bei meinen Betreuern am Institut für Technische Informatik bedanken. Vielen Dank, **Kay Römer**, für Ihr konstruktives, ehrliches und faires Feedback – ich kann mir vorstellen, dass es kein leichter Job ist, so viele Studenten zu betreuen, unseren Ameisenhaufen von Institut zusammenzuhalten und trotzdem noch genug Zeit für jeden einzelnen zu haben, um konstruktiv und respektvoll zu sein, wie Sie es immer sind. Danke, **Georg Macher**, dass du mir die Chance gegeben hast, am Institut und an meiner Dissertation weiterzuarbeiten. Du hast mich gelehrt, überall ein bisschen Sarkasmus und Spaß zu sehen und mit offenen Augen durch die Welt zu gehen. Danke, **Christian Steger** und **Eugen Brenner**, für die emotionale und organisatorische Unterstützung in stressigen Zeiten und die aufschlussreichen und interessanten Diskussionen am Institut.

*Graz, Mai 2021
Michael Krisper*

Abstract

Cybersecurity incidents cause tremendous costs for the economy and damage for individuals, e.g. through identity theft, data loss, ransomware or bribery. To find appropriate measures to reduce or prevent such incidents, a system must first be assessed regarding its risks. In domains such as safety, harmful events can be predicted by looking at past events, modelling them and applying these models to the future. For cybersecurity, however, such incidents are much harder to predict because they depend mainly on the motivation and decisions of humans. To evaluate this, one has to resort to expert judgments, which are unfortunately subject to large uncertainties. In this thesis, the structured expert judgment method is used to estimate the risks for cybersecurity incidents. The risks are calculated by forward and backward propagation of specific risk attributes along with their uncertainties. This is done on risk graphs in which all attack paths are mapped. The result is a risk distribution that can be traced back to the individual components. This supports making better decisions on the necessary measures to reduce risk. Correctness, applicability, and usefulness were demonstrated using an implemented prototype. For this purpose, a comparison of 45 publicly available studies was made using structured expert judgment and RISKEE. Furthermore, the created RISKEE method was applied in an international workshop to investigate the cybersecurity risk of car theft. Finally, the implemented prototype was used to find secure solutions for chip designs in a design space exploration study.

Kurzfassung

Cybersecurity-Vorfälle verursachen enorme Kosten für die Wirtschaft und Schaden für den Einzelnen, z. B. durch Identitätsdiebstahl, Datenverlust, Ransomware oder Bestechung. Um geeignete Maßnahmen für die Verringerung oder Vermeidung solcher Vorfälle zu finden, muss ein System zuerst hinsichtlich seiner Risiken bewertet werden. In Bereichen wie der Ausfallsicherheit können Schadensereignisse gut vorhergesagt werden, indem man sich vergangene Ereignisse ansieht, sie modelliert und diese Modelle auf die Zukunft anwendet. Für Cybersecurity sind solche Vorfälle jedoch viel schwieriger vorherzusagen, weil sie hauptsächlich von der Motivation und den Entscheidungen von Menschen abhängen. Um das zu bewerten, muss man auf Expertenurteile zurückzugreifen, die leider mit großen Unsicherheiten behaftet sind. In dieser Arbeit wird die Methode des strukturierten Expertenurteils verwendet, um die Risiken für Cybersecurity-Vorfälle abzuschätzen. Die Risiken werden berechnet durch Vorwärts- und Rückwärtspropagation spezieller Risikoattribute mitsamt ihrer Unsicherheiten. Dies geschieht auf Risikographen, in denen alle Angriffspfade abgebildet sind. Das Ergebnis ist eine Risikoverteilung über ein System, die auf die einzelnen Komponenten rückverfolgbar ist. Dies ermöglicht bessere Entscheidungsfindung für die erforderlichen Maßnahmen zur Risikominderung. Korrektheit, Anwendbarkeit, und Nützlichkeit wurden mithilfe eines implementierten Prototyps gezeigt. Dazu wird ein Vergleich der Resultate von 45 öffentlich zugänglichen Studien gezogen, die mittels strukturiertem Expertenurteil und RISKEE durchgeführt wurden. Weiters wurde die RISKEE Methode in einem internationalen Workshop angewandt, um das Cybersecurity-Risiko von Autodiebstahl zu untersuchen. Letztlich wurde der erstellte Prototyp in einer Design-Space-Exploration Studie eingesetzt, um sichere Lösungen für Chip-Designs zu finden.

List of Figures

2.1	Examples for Common Probability Distributions	13
2.2	Normal Distribution	13
2.3	Uniform Distribution	14
2.4	Lognormal Distribution	14
2.5	PERT Distribution	15
2.6	Examples for the PERT-beta Distribution	16
2.7	Kernel-Density-Estimation	18
2.8	Different Types of Estimates	19
2.9	Examples of Inequalities Between Different Means.	22
3.1	Overview over Relevant Standards	27
3.2	NIST SP-800-30 Likelihood Scale	29
3.3	FMVEA Risk Matrix	32
3.4	HEAVENS Risk Matrix	32
3.5	Cyber-Kill-Chain Example	33
3.6	Illustration of the Diamond Model Capabilities	34
3.7	FAIR Ontology	36
4.1	RISKEE Method for Expert Elicitation	40
4.2	Risk Graph Example	41
4.3	Risk Graph Structure and Node Types	42
4.4	RISKEE Entry Node	42
4.5	RISKEE Intermediate Node	42
4.6	RISKEE Impact Node	42
4.7	Examples of Simple Risk Graphs.	43
4.8	Merging Attack Paths.	44
4.9	Branching Attack Paths.	44
4.10	RISKEE Risk Model and Ontology	45
4.11	Risk Visualisation as Distribution and Loss-Exceedance-Curve	46
4.12	Increasing Uncertainty over Time and Time-Dependent Attack-Frequency	47
4.13	Vulnerability Calculation	49
4.14	Example for Risk Propagation	51
4.15	Example for a Single Attack Path	52
4.16	Example for RISKEE Risk Calculation	53
4.17	Risk Graph Example	54
4.18	Two Examples for Stratified Sampling	57
4.19	Arithmetic Operations on Probability Distributions	58
4.20	Normal Distribution and Intervals for Standarddeviations.	59
4.21	Reflection and Truncation on Probability Distributions	60

4.22	Comparison of Properties for Good and Bad Experts.	63
5.1	Dataflow for Expert Judgment	69
5.2	Dataflow for the RISKEE Propagation Algorithm	70
5.3	Probability Distribution of Possible Losses	71
5.4	Probability Density versus Survival Function (Loss Exceedance Curve)	71
5.5	Linear scaling of expert judgments.	72
5.6	Logarithmic scaling of expert judgments.	72
5.7	The ProbabilityDistribution Class Diagram	74
6.1	Models for Estimations in the Classical Model and in RISKEE.	78
6.2	Correlation of Information Score	80
6.3	Correlation of Calibration Score	80
6.4	Correlation of Resulting Weights	81
6.5	Questionnaire for Expert Judgment.	82
6.6	Calibration Results for Experts	83
6.7	Comparison of Results for the Expert Judgment Use-Case	84
6.8	Scatter Plot for Design Space Exploration Using RISKEE	85
A.1	Paper-Overview: An overview of the included publications.	94

List of Tables

1.1	Problems, Challenges, and Contributions	4
2.1	Confidence Classes for the PERT Distribution	17
2.2	Examples for Common f-Mean Parametrizations	22
3.1	Comparison of Related Standards	28
3.2	HARA Risk Matrix	30
3.3	HARA Assessment Parameters	30
3.4	SAHARA Risk Matrix for Calculation of Security Level	32
3.5	SAHARA Parameters	32
6.1	Weighting-Factors for the Experts	83

List of Algorithms

1	RISKEE Propagation Algorithm	51
---	--	----

List of Abbreviations

- ACQ** Acquisition.
- ADAS** Advanced Driver Assistant System.
- ALARP** As Low As Reasonably Possible.
- ASIL** Automotive Safety Integrity Level.
- ASPICE** Automotive SPICE.
- ATA** Attack-Tree Analysis.
- BBN** Bayesian Belief Network.
- BDMP** Boolean logic Driven Markov Process.
- BRA** Binary Risk Analysis.
- C2** Command and Control.
- CC** Common Criteria.
- CDF** Cummulative Distribution Function.
- CISO** Chief Information Security Officer.
- CMMI** Capability Maturity Model Integration.
- CPS** Cyber-Physical Systems.
- CTA** Cyber Threat Actors.
- DAG** Directed Acyclic Graph.
- EAL** Evaluation Assurance Level.
- FAIR** Factor Analysis of Information Risk.
- FIT** Failures In Time.
- FMEA** Failure Mode and Effects Analysis.
- FMEDA** Failure Mode, Effects and Diagnostics Analysis.
- FMVEA** Failure Mode, Vulnerabilities and Effects Analysis.
- FTA** Fault-Tree Analysis.
- HARA** Hazard Analysis and Risk Assessment.
- IIoT** Industrial Internet-of-Things.
- IL** Impact Level.
- IoT** Internet-of-Things.
- IT** Information Technology.
- KDE** Kernel-Density Estimation.
- LEC** Loss Exceedance Curve.
- LEF** Loss Event Frequency.

LM Loss Magnitude.

NIST National Institute of Standards and Technology.

OCTAVE Operationally Critical Threat, Asset, and Vulnerability Evaluation.

OMG Object Management Group.

OOD Object-Oriented Design.

PDF Probability Density Function.

PERT Program Evaluation and Review Technique.

PP Protection Profiles.

PPF Percent Point Function.

QoS Quality of Service.

RMS Root Mean Square.

RPN Risk Priority Number.

RS Resistance Strength.

SAHARA Security-Aware Hazard and Risk Analysis.

SAR Security Assurance Requirement.

SCADA Supervisory Control And Data Acquisition.

SecL Security Level.

SEI Software Engineering Institute (Carnegie Mellon University).

SF Survival Function.

SFR Security Functional Requirement.

SL Security Level.

SOA Service-Oriented Architecture.

SPD Security, Privacy, Dependability.

SPICE Software Process Improvement and Capability dEtermination.

SPoF Single Point of Failure.

ST Security Targets.

STRIDE Spoofing, Tampering, Repudiation, Information disclosure, Denial of service, and Elevation of privileges.

SUP Supporting Process.

SWE SoftWare Engineering.

SYS System Engineering.

TARA Thread Analysis and Risk Assessment.

TC Threat Capability.

TCap Thread Capability.

TEF Threat Event Frequency.

THROP THreat and OPerability analysis.

TL Threat Level.

TOE Target of Evaluation.

VaR Value at Risk.

Vuln Vulnerability.

Contents

1	Introduction	1
1.1	Context and Domain	2
1.2	Problem and Challenges	2
1.3	Hypothesis	7
1.4	Structure of the Thesis	8
2	Background	9
2.1	Risk Management and Risk Assessment	9
2.2	Types of Measurement Scales	11
2.3	Probability Distributions	13
2.4	Gaussian Error Propagation: The Law of Propagation of Uncertainty	19
2.5	Opinion Pools	20
3	Related Work on Risk Assessment in Cyber-Security	23
3.1	RISKEE and Graphical Models	23
3.2	RISKEE and the FAIR Method	25
3.3	RISKEE and Structured Expert Judgment	25
3.4	RISKEE and the IDEA Protocol for Expert Elicitation	25
3.5	RISKEE and Qualitative Assessments	26
3.6	Uncertainty in Distributed Control Systems	26
3.7	RISKEE and Risk Management Standards	26
3.8	Other Related Scientific Work	37
4	Design of the RISKEE Method	39
4.1	The RISKEE Method	40
4.2	Risk Graphs	41
4.3	The RISKEE Propagation Algorithm	50
4.4	Distribution Arithmetic	55
4.5	Expert Judgment and Consolidation	60
5	Implementation of the RISKEE Framework	67
5.1	Requirements	67
5.2	Framework Architecture and Design	68
5.3	Implementation	72
6	Evaluation	77
6.1	Evaluation Part 1: Comparison to the Classical Model	77
6.2	Evaluation Part 2: Use Case: FAST, FURIOUS and INSECURE	82
6.3	Evaluation Part 3: Providing a Cyber-Security Metric for Design Space Exploration	84

7	Conclusion	87
7.1	Future Work	87
7.2	Conclusion	90
A	Appendix: Publications	91
A.1	[P1] Physical Quantity: Towards a Pattern Language for Quantities and Units in Physical Calculations	96
A.2	[P2] Use-Cases for Uncertainty Propagation in Distributed control Systems	118
A.3	[P3] Patterns for Implementing Uncertainty Propagation	132
A.4	[P4] Towards Integrated Quantitative Security and Safety Risk Assessment	140
A.5	[P5] RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber-Security	156
A.6	[P6] Patterns for Communicating Numerical Uncertainty	170
A.7	[P7] Assessing Risk Estimations for Cyber-Security Using Expert Judgment	186
A.8	[P8] Expert Judgment for Cyber-Security-Risk	202
A.9	[P9] Problems with Risk Matrices using Ordinal Scales	208
A.10	[P10] Towards an Automated Exploration of Secure IoT/CPS Design Variants	220
A.11	[P11] Towards Security Attack and Risk Assessment during Early System Design	236
	Bibliography	245

Introduction

We can be blind to the obvious, and we are also blind to our blindness.

Daniel Kahnemann

Summary: *This chapter introduces the topics of this dissertation and summarizes the problem, challenges, and contributions. After introducing and motivating the topic of cyber-security for a general audience, the context and domain for this thesis are described. Then the scientific problem, challenges, and hypothesis are outlined. At the end of this chapter, the structure of the remaining thesis is shown.*

◇◇◇

What is the risk of your car killing you? This question might sound strange at first, but it is not as far-fetched as it might sound. Cars are good examples to elaborate on cyber-security risks in Cyber-Physical Systems (CPS). With more and more driver assistance systems in place, the risk that a malicious attacker is taking over the control of a car while driving is getting more and more realistic. Already today, complex driver assistance systems are built into cars and help us navigate, stay on track, adapt the speed to the driver in front of us, or help us find a parking space. In a few years, when the era of autonomous self-driving and connected cars arrives, we will drive in cars with autonomy levels of 3, 4, and even 5 [1, 2]. This means that the car will be in control of the driving. Since such cars are also connected to the Internet or allow spontaneous wireless connections, they open up an attack surface targeted by hackers. The connectivity combined with the control over a car allows remote control of cars and even whole fleets. Hackers could initiate spontaneous accelerations, control the steering wheel or brake unexpectedly. Also, theft or damaging components could be possible. Such attacks have been made in the past [3, 4] and this trend will continue. *If we do not prepare and harden our systems against such attacks, this could result in catastrophic events with high casualties – financially as well as harm to human lives.*

These threats and risks introduced here for the automotive domain also apply to many other domains. According to the current *State of Phish*-study by Proofpoint [5], 88% of all surveyed organizations from many domains have encountered some form of cyber-security attack in the year 2019. Furthermore, the *Cost of Insider Threats 2020*-study by IBM and Ponemon Institute stated that 60% of companies even had 20 or more attack attempts in the year 2020 [6], and the current *2021 Hacker Report* by HackerOne found an overwhelming 1000% increase in attacks on IoT devices in 2020 [7]. Also, CrowdStrike's *2021 Global Threat Report* found an alarming increase and focus on critical infrastructure, especially the health sector, due to phishing attacks hiding as COVID-19 information [8].

The current megatrend of digitalization in industrial domains opens up systems and critical infrastructures as possible targets for cyber-criminality. With the Internet of Things (IoT) and Cyber-Physical Systems (CPS), connected and smart devices will be everywhere around us. Organizations already began to open up their formerly closed and protected systems to be distributed, more flexible, and maintainable over the Internet. While the technological innovations in wireless communication, artificial intelligence, the IoT and CPS, improve our lives, they also increase the attack surface and bring dangers that we have never faced in history before. Cyber criminality in the forms of, e.g., malicious attacks, data theft, and ransomware, is already in daily news and will become even worse in the future [9, 10]. How can we protect our systems against cyber-attacks? How can we evaluate which protection measures are helpful and most effective? What is the risk of cyber-security incidents in our systems? In this thesis, we tackle the problem of how to assess the risk of cyber-security in distributed networked systems. We do this by using Structured Expert Judgment (SEJ) to assess specific risk attributes, including the uncertainty of the respective judgment. This is needed because estimating risk involves future predictions, which always contain uncertainty and the possibility of error.

We use a graph structure representing the attack paths in our systems and propagate the uncertain risk estimations through this graph. This allows us to model multi-step attacks over distributed and complex systems. In the end, the risk is the conglomerate of all possible outcomes, with some outcomes being more likely than others. By looking at this big informative picture of risk, it is possible to decide if it is acceptable or if we have to take measures to lower it.

1.1 Context and Domain

This thesis was written at the Institute of Technical Informatics at Graz University of Technology. It was financed by project funds from a project cooperation with Andritz Hydro AG in the HyUnify and DHYAMONT projects between 2016 and 2020. Andritz Hydro AG is the leading company in the hydropower domain - they build, maintain, operate, and optimize hydropower plants worldwide. Since hydropower plants are part of the worldwide power grid, they belong to critical infrastructure. The publications and scientific work in these projects focused on robustness, reliability, resilience, safety, and security in industrial informatics.

1.2 Problem and Challenges

Cyber criminality inflicts tremendous financial damage and, even worse, could pose a potential threat to human life if critical infrastructures are attacked. Current statistics show that this menace is prevalent in every domain, affects most companies, and will get even worse in the future [9, 10]. Therefore, it is of the utmost concern to prevent cyber criminality and avoid potential damage and harm. To prevent them, one has to find different mitigation and prevention strategies and evaluate their effectiveness to decide the appropriate ones. It is difficult to quantify the costs, threats, and mitigation effects because modern systems are highly sophisticated in their function, are distributed over the Internet, communicate with many other systems, and depend upon many other services. Examples for this would be distributed microservice architectures and cloud structures for big companies, connected devices in the IoT, dependable industrial systems, and Cyber-Physical-Systems. Furthermore, systems for critical infrastructure like transport, telecommunication, or government have even higher requirements on availability and reliability and other dependability properties, even in the presence of hazardous events. Their failure could result in fatal consequences. So besides being available, safety is also a concern. We do not want our systems to harm any living being or the environment during its op-

eration. Classical safety-and-system engineering takes care of these properties and ensures that the system keeps running safely. However, nowadays, most systems also face a danger that could break these properties: cyber-criminality. By being vulnerable to cyber-attacks, a system exposes itself to being compromised or corrupted and unable to guarantee the safety and uninterrupted functionality anymore. Systems have to be hardened against cyber-attacks to avoid this problem. One challenge here is to decide on appropriate mitigation strategies because there is an abundance of possibilities, attacks, and vulnerabilities. Mitigation and prevention strategies must be evaluated for their ability to increase systems' resilience and resistance against cyber-security-attacks. This evaluation of risks and the reduction thereof by mitigation strategies are the core problems discussed in this thesis:

Problem

How can the risk of cyber-security and the effects of mitigation strategies in a networked system be quantitatively evaluated?

This core problem can be broken down into two sub-problems:

1. *How to evaluate the current state of security in a system and assess the risks?* This involves estimating the plausible attackers, their capabilities, resources, and knowledge and comparing this to the system under investigation, including its mitigation and protection measures which are already in effect.
2. *How to evaluate the effectiveness of mitigation strategies?* Applying protection measures should reduce the total risk, but the question is, by how much? By recalculating the total risk with changed estimations due to the activated protection mechanisms, one can compare the scenarios and see if and how much the systematic risks change.

1.2.1 Challenges

In this section, the challenges for risk estimations of cyber-security in networked systems are described. Table 1.1 shows an overview of the challenges and the respective contributions and supporting publications. The publications for this cumulative thesis are listed and included in the Appendix. The main challenges of risk models for cyber-security are:

- **High Complexity:** Networked distributed systems can get very complex. Here, complexity means the unpredictable behaviour of individual components and their impact on the state of the networked system as a whole. For robust and dependable systems, predictability is key. When multiple components of a system communicate and influence each other, it is complicated to ensure that everything works predictably. Especially since Byzantine faults can not be ruled out, and the remaining system has to cope with such situations.
- **Huge Uncertainty:** It is already challenging and unintuitive to estimate the vulnerabilities and risks of single components, let alone for whole systems. Furthermore, risk assessments are predictions, which are always uncertain. Together, this means that the uncertainty is relatively high in risk estimations. Hence, it must be included in the model. If the uncertainty is neglected, the results depict a wrong image of confidence and precision.
- **Difficult Predictions:** Cyber-Security attacks are often a combination of several exploits and steps, done in very creative ways. Judging the probabilities and vulnerabilities of such involve high uncertainty. While known and already analysed attacks are easier to judge, unknown attacks that are not even invented are challenging to predict. To also include the unknowns in a risk assessment is crucial for getting a realistic prediction.

In the following sections, each challenge is discussed in more detail, and also our contributions to address them are described in summary. For a more detailed discussion about the contributions, see Chapters 4 and 5 about the design of the RISKEE method and the implementation of the RISKEE framework. These contributions are also published in the scientific articles included in the Appendix, while this thesis collects it in a more concise and continuous form.

Problems	Challenges	Contributions	Supporting Papers
How to assess risks in cyber security?	Figures of Merit: What are good metrics to evaluate systems?	Probabilistic ratio-scaled ranges with uncertainty.	P1 P2 P3 P4 P5 P9
	Qualitative vs. Quantitative Assessment: What is better?	Quantitative Assessments with Probability Distributions.	P2 P3 P4 P5 P7 P9
→ How to evaluate the security risks of a system?	System Model: How to model complex systems?	Risk graphs: Bipartite graphical model of a system using risk attributes.	P4 P5 P7 P10
	Attack Model: How to model multi-step attacks over multiple components?	Individual estimations of attack steps along the paths.	P5 P7 P8
Decomposition of attack frequency and vulnerability for each step.		P4 P5 P7	
→ How to evaluate the effectiveness of mitigation strategies?	Propagation and Aggregation: How to get total cumulated risk estimations for a system?	Systematic sampling for probability distributions.	P3 P5
		Forward and backward propagation.	P5
		Smoothing with reflected kernel-density-estimations.	P5
	Mitigation Effects: How to calculate the effects of mitigation strategies?	Resimulation and comparison to original state.	P5 P7 P10 P11
Data Sources: How to get reliable and defensible data?	Calibrated expert judgment with performance-based weighting.	P4 P5 P6 P7 P8	
Communication: How to communicate risk informatively and in an unbiased manner?	Graphical probabilistic loss exceedance curves.	P4 P5 P6 P7 P8	
		Numerical projection to descriptive statistics with statistical moments, or n-point percentile summaries.	P6

Table 1.1: Problems, challenges, and contributions with support of the included published papers (stated in the boxes on the right side).

Figures of Merit - What are useful metrics? Traditionally, the risk is calculated from the probability of a hazardous event happening and the consequential outcome after the hazardous event occurred. Consider the following example: In 2% of cases, a device is defective, and the replacement cost is € 5000. This would result in a risk value of € 100 (2% of € 5000). While this approach works quite well for project management, business risk, and even safety, such hazardous events are not easily modelable in cybersecurity. The problem lies in evaluating the event-probability for attacks, which has a nondeterministic nature involving the attacker's motivation and window of opportunity. This is different compared to the relative deterministic behaviour of failure rates.

The challenges here are as follows:

1. It is challenging to come up with a single metric that represents risks for different dimensions of qualities. But using multiple metrics makes deciding on the appropriate strategies exponentially more difficult.
2. Measurements, predictions, and estimations of these metrics involve significant uncertainty, which is often neglected in state-of-the-art methods.

The first point is difficult because several conceptions and misconceptions exist about what an important metric could be. In project management and business circles, the financial impact is of utmost importance, while in safety, the potential harm to humans is the value that must be assessed. In environmental studies, the impact on the environment is the most critical metric. Based on literature research, our contribution is that at least two types of metrics must be used for the impact: *quantitative impact in the form of financial losses and qualitative impact in the form of harm to humans or the environment*. While the former can be clearly stated on a ratio scale with money, the latter is not quantifiable. It depends on the cultural background, social norms, and ethics, which all are highly subjective. In our published work and this dissertation, we focus on the quantifiable financial losses (coming from several sources, e.g., replacements, response, productivity outage, damaged reputation).

The second point is a challenge because state-of-the-art methods often use ordinal scales and single-point metrics, thereby neglecting uncertainty at all or introducing uncertainty that does not correlate with the actual value and may lead to inconsistent results. Recently, Cooke et al. have shown that quantitative assessment using ranges are superior to single-point estimates [11]. The challenge is to initiate a rethinking of existing methods and introduce easy-to-use guidelines for improving them. This is a challenging task due to the steadiness and inflexibility of standards and established methods in the industry. *We use range predictions in the form of probability distributions that include the estimation's inherent uncertainty.*

Qualitative vs Quantitative Assessment - What is better? As mentioned before, traditional risk assessment methods use ordinal scales with qualitative estimations of the properties. The so-called “risk-matrices” and “traffic-light” models have become so popular that it is tough to replace them. However, they all have their flaws: Cox et al. [12], and others [13, 14], have proven that ordinal scales and risk matrices have severe flaws and problems. The most severe ones are the induced quantification errors, risk-inversion, range-compression, and the impression of benefit. We add to the corpus of scientific literature by showing that quantitative assessment (including uncertainty) is superior to qualitative assessment in the cyber-security domain. The quantities are best expressed in probability distributions or, respectively, probabilistic ranges.

System Model - How to model complex systems? Another challenge in the risk assessment domain for cyber-security is how to create a realistic model of the system. Using infrastructural network models alone is not sufficient anymore since side-channel attacks became much more prevalent. Especially in the security domain, social engineering and human error are pretty standard. Since Bruce Schneier coined the term attack trees in the 80s, computer security has gone a long way, and nowadays, Bayesian attack graphs, cyber-security kill chain models, and Markov chains are the state-of-the-art. However, they only consider the risk with single-point estimations and do not consider uncertainty at all. *We enhance existing graphical models of systems by using probability distributions as model parameters to calculate risk - we call this resulting model a risk graph.* Such a risk graph models all possible

communication and influence paths in a system, including possible attack paths, and allows for more realistic risk calculations by considering the inherent uncertainty in the input estimations.

Multi-Step Attacks - How to model complicated attacks over multiple hops? Since its beginnings, cyber-criminality abuses flaws and exploits gaps or errors in applications and operating systems to achieve a specific goal. Nowadays, cyber-attacks are not that simple anymore and contain many attacks and several exploits in unexpected combinations. Most attacks are, in fact, multi-step attacks that are difficult to detect and mitigate since they use specific combinations of vulnerabilities [15]. It is not enough for a holistic risk assessment to look at single events anymore. Instead, whole attack paths consisting of multiple steps must be considered. All those attack paths have to be combined to get a holistic view of the system's risks.

Furthermore, different attacks demand different attacker strengths, motivations, and opportunities to be successful. Such differences in attacker profiles also have to be considered. Looking at the system only from the defender side makes it difficult to predict all eventualities. *Therefore, we propose the modelling and estimation of individual attack paths over multiple steps for each plausible attacker profile in our system model.* For each attack path and attacker profile, the frequency and vulnerability have to be estimated on each step. This decomposed view makes it much easier to assess the respective values (always including the uncertainty).

Propagation and Aggregation - How to get total cumulated risk estimations for a system?

The next challenge is the actual calculation of the resulting total risk of a system. Since we use probability distributions, we cannot use trivial arithmetics anymore but have to apply sophisticated methods that consider uncertainty propagation. Moreover, we cannot rely on analytical methods since the input probability distribution could be of arbitrary shape and kind. *We solved this problem by applying Monte-Carlo methods with stratified and adaptive sampling.* Monte-Carlo sampling allows for arbitrary combinations of any probability distribution. Stratifying them is crucial to consider the long-tails, and adaptivity ensures that the precision and performance are balanced. *To avoid consequential quantification errors due to quantization, we apply smoothing via kernel-density estimations for the resulting distributions.* One requirement for this to work well is to have limited support on the probability distributions, which is a valid assumption since probability values are always between 0 and 1. Financial damage is also limited by 0 and some arbitrary upper bound.

Another challenge is the combination of multiple attack paths. Since these are modelled individually, we had to develop means to aggregate them for the whole system. *This is why we invented the RISKEE-propagation algorithm, which works by propagating attack frequencies and probabilities forward, calculating the risk for the individual path, and then propagating this risk backward again.* This allows for calculating the total risk using probability distributions and identifying the individual contribution of each node.

Mitigation Effects - How do changes contribute to the risk of the whole system?

The next difficulty is finding out the effects of applying a specific mitigation strategy on a system. This is difficult because mitigation strategies applied to a particular component in a system may have far-reaching consequences for the overall risk due to subsequent influence on dependent components. *In our risk graph, this is solved via adaptive recalculation for the changed attack paths.* In such a way, it is possible to make small changes according to the applied mitigation strategy and get fast results by not having to recalculate the whole tree but only doing the forward and backward propagation of the changed values on the affected parts. The effects of a mitigation strategy on the overall system risk can be calculated

and then compared to the original risk distribution to see if it achieves the intended risk reduction. It is also possible to make a cost-benefit analysis to get the risk “As Low As Reasonably Possible” (ALARP) - a common technique and expression often used in risk management.

Data Sources - How to get reliable and defensible data? One of the biggest challenges in cyber-security risk assessment is to get reliable data. While the impact or damage can be modelled in terms of costs, e.g., replacement, response, productivity outage, the attack frequency, and the success probability are more challenging to estimate. Firstly, attacks are instantiated by humans whose behaviour and motivations are complex, highly subjective, and nondeterministic, and therefore, are also difficult to model. However, there have been quite some efforts to do so [16, 17, 18]. However, even after considering several factors like motivation, opportunity, target attractiveness, or political intentions, it still not possible to pinpoint the amount and probability of attacks accurately. That is why probabilistic range estimations have to be used. Such models always involve assessing input values via historical data, inherited from other methods, or are directly judged using expert elicitation. All of these sources contain uncertainty that has to be considered. *By taking the sources’ uncertainty into account, a more realistic and defensible data corpus can be created. Specifically, for expert judgment, we propose to use calibrated expert elicitation with performance-based weighting. For other data sources, a projection has to be made, which considers and adds uncertainty.*

Communication - How to communicate risk informatively and in an unbiased manner? The final challenge is human bias in risk communication: How to communicate the risks calculated in the system so that stakeholders can understand and interpret them to make informed decisions based on that? The problem here is human bias. Humans are very biased when it comes to risks, money, and fears. That is why often wrong or inefficient decisions are made in this sector. *Our solution is to visualize the risk in the most informative and still intuitively understandable manner by using so-called loss exceedance diagrams. Numerical alternatives to this would be, e.g., percentile estimations, stating, e.g., the median, lower, and upper quartile of risk distributions.* This can then be compared to the risk-appetite (or risk-aversion), and based on that, one can decide if the risks are acceptable or not.

1.3 Hypothesis

Based on this problem and the respective challenges, we state the following hypothesis, which will be discussed in this thesis:

Hypothesis

Cyber-security risk can be evaluated using a graphical model of attack paths in a networked system and assessing the uncertain risk attributes of frequency, vulnerability, and impact with probability distributions based on combined structured expert judgment. By forward and backward propagation of these uncertain estimations, the total systematic risk, as well as the individual risks, can be evaluated.

This hypothesis consists of multiple parts and clauses, which are discussed here in more detail. The first statement is the basic assumption which our hypothesis is based on: *Cyber-security can be evaluated using a graphical model of attack paths in a system.* This is already established in the literature and proven in practice, which is why we use it as the base assumption for our hypothesis statement.

We enhance this basis by declaring the specific risk attributes *frequency*, *vulnerability*, and *impact*. These three attributes are needed to assess risk. Furthermore, these attributes are *assessments* which exhibit high *uncertainty* that can be modelled using *probability distributions*. It is important not to neglect the uncertainty or reduce it to ordinal scales since this could lead to problems. To get such assessments in a reliable and defensible way, we propose *structured expert judgment* which combines the estimations by multiple experts based on their judgment quality.

The last part of the hypothesis concerns how risk can be calculated on a risk graph with the risk attributes as a basis. It can be calculated by using *forward and backward propagation*. Forward propagation forwards the incoming attacks to all reachable nodes, which result in an impact. There, the risk emerges. This risk is then propagated back again over the paths to determine the *individual risk contribution* of each step compared to all other nodes in the risk graph.

1.4 Structure of the Thesis

The remainder of this thesis is structured as follows:

- **Chapter 2** describes the **background** knowledge for this thesis. This includes risk and risk management, statistics and probability distributions, uncertainty propagation, and expert elicitation.
- **Chapter 3** describes the **related work** for this thesis. This includes many standards and methodologies from the areas of risk management, cyber-security, and safety.
- **Chapter 4** discusses the **design** of the RISKEE method. It explains risk graphs and risk attributes, distribution arithmetic, and expert judgment. The structure, the attributes, and the dynamics of risk graphs are described. In this chapter, the contributions in the domain of expert judgment and expert elicitation are discussed, and the advancements to the classical method of structured expert judgment are shown.
- **Chapter 5** explains the more practical contributions of this thesis by showing and discussing the **implementation** of the RISKEE framework, which consists of a prototype written in Python. This prototype was the basis for the evaluations.
- In **Chapter 6**, an **evaluation** of the RISKEE method and framework is shown, and afterwards, several aspects of the method are discussed. The evaluation is done in three parts: First, a comparison of RISKEE expert judgment to the classical model is made. Second, a use case in the form of an actual expert elicitation for judging the risks of a cyber-security incident. The third part is an application of RISKEE in design space exploration to find secure solutions.
- The **conclusion** and **future work** in **Chapter 7** elaborate on the open issues and future work as well as some final thoughts and take away messages.
- In the **Appendix**, the full-texts of eleven published papers are attached, together with summaries and descriptions of the respective contributions.

Background

The most important questions of life are indeed, for the most part, really only problems of probability.

Pierre-Simon Laplace

Summary: Here, the background knowledge for the thesis is described. This chapter contains general knowledge about risk management and risk assessment, statistics, probability distributions, error propagation, and expert judgment. It serves as a primer to these topics and provides further references to dive into these topics more deeply if needed.

◇◇◇

2.1 Risk Management and Risk Assessment

Risk management and risk assessment are widely applied in many domains, especially the safety and security domains relevant to this thesis. First, we give an overview of the general definitions of risk, and afterwards, the relevant security standards and norms will be covered in more detail. This overview shows that most standards qualitatively evaluate risk using ordinal scales and risk matrices.

2.1.1 Risk

Let us first look at the definition of risk: In the Merriam-Webster Online Dictionary, **risk** is defined as the “*possibility of loss or injury*” [19], the Cambridge Online Dictionary defines it as “*the possibility of something bad happening*” [20], and the Oxford English Dictionary describes it as “*(Exposure to) the possibility of loss, injury, or other adverse or unwelcome circumstance; a chance or situation involving such a possibility*” [21]. The two fundamental standards on risk, the ISO/Guide 73 [22], and the ISO 31000 [23] describe risk similar as the “*effect of uncertainty on objectives*”, where an effect is very generally defined as a deviation from the expected, and objectives are different aspects and levels where this can be applied to (e.g., financial objectives, safety, health, organisation-wide, on the process- or product-level). Based on this definition, every deviation from the expected outcome is seen as a risk, critical for production, processes, and engineering. All definitions have in common that risk involves a possibility (uncertainty about some situation in the future) of a bad or unexpected result. This notion leads to an interesting insight: That risk actually “does not exist” because it is only a prediction. Risk researcher Paul Slovic puts it like this:

[Risk] does not exist “out there,” independent of our minds and cultures, waiting to be measured. Instead, the risk is seen as a concept that human beings have invented to help them understand and cope with the dangers and uncertainties of life. Although these dangers are real, there is no such thing as “real risk” or “objective risk.” [24]

Here, the metaphysical aspect of risk is emphasized. Risk is a concept of potential danger which exists only virtually. Only when the dangerous event occurs, the danger becomes real. However, albeit risk only being a virtual concept, it helps us to model, evaluate and compare dangers and threats. According to Murphy’s law, everything that can happen will happen eventually. Hence, we have to be prepared to face the risks and cope with them. Maybe we can decrease the probabilities, decrease the impact, or avoid some risks altogether by such preparations. This, at last, is the reason why it is required to think about risks.

Wild and Mild Risks: Black Swans, Grey Rhinos, and Elephants in the Room In his famous book *“The Misbehavior of Markets”* [25], Mandelbrot introduced the terms *mild* and *wild* risks to distinguish between two common risk categories: Mild risks can be modelled easily and behave according to a normal distribution with small uncertainties, meaning that the results are almost always relatively close to the mean value. On the contrary, wild risks behave very chaotically, are maybe multimodal, and have long-tailed or heavy-tailed distributions, which means that it is way harder to predict the outcomes inside a small tolerance range. Such rare but catastrophic events are called *Black Swan* events [26]. There are also other classes in the risk-zoo, namely, Grey Rhinos [27], and elephants in the room [28, 29], which represent events that have quite high probability and moderate or high impact but are neglected due to ignorance or social/political taboos. The difference is that grey rhino events did not occur yet, while elephants in the room already occurred but still are ignored. Here is a list of commonly used terms when talking about risks:

- **Mild Risks:** Easily to model, highly predictable.
- **Wild Risks:** Chaotic, difficult to predict, having heavy-tailed distributions with high uncertainty.
- **Black Swan:** A rare but catastrophic event.
- **Grey Rhino:** Obvious and dangerous but ignored events.
- **Elephant in the Room:** Events that occurred but are ignored.

Risk Management

According to the ISO 31000 [23], risk management has the goal of efficiently identifying and evaluating risks (**risk assessment**) and reducing them to a tolerable level (**risk treatment**). Risk assessment is the process of risk identification, analysis, and evaluation to get quantifiable values for the assets, threats, and mitigation strategies. Risk treatment is the process of selecting, applying, and monitoring appropriate mitigation techniques to lower the risks and ensure the effectiveness.

In business, the notion of risk describes possible financial losses, which is a significant factor for decision-making. Social and ecological factors like sustainability are only secondary factors. However, organizations slowly begin to recognize that a company’s financial sustainability is interwoven with social, ecological, and environmental sustainability and should therefore also be considered [30]. There are many models for market development and predictions of customer behaviour, estimating possible developments and market reactions, and making informed decisions based on those models. These models are often simplified using risk matrices and the well-known traffic-light concept for risk assessment: Green equals low risk, yellow or amber representing medium risk, and red colour signals

high-risk events. The classical formula for risk comes from this domain (see Equation 2.1):

$$\text{Risk} = \text{Probability} \cdot \text{Impact} \quad (2.1)$$

Here, the probability is the chance or likelihood that a risk event occurs, and impact is the financial loss for such an event. This is often not estimated with quantitative values but using ordinal scales for the respective factors, e.g., high, medium, or low probability; and high, medium, or low impact. The scales are specified completely arbitrarily and tailored to the respective context in the company. For example, the impact ratings for a small company could be: Low is between € 0 and € 1 000, the medium is between € 1 000 and € 10 000, and high impact is everything above € 10 000. For big companies, these values could be somewhere in the millions. The same holds for the probability: The exact ranges for the classes are completely arbitrarily chosen. To make things even worse, these ordinal classes are then combined using an arbitrary definition in the form of a risk matrix. Anthony Cox described these problems [31, 32], and we also summarized those flaws and pitfalls in risk matrices in [33]. However, risk matrices are already established in safety and security, and it will be a long way to eliminate them. As we will describe in later chapters, we propose incorporating them into a quantitative risk analysis by transforming the classes into value ranges and using them in the calculations. In such a way, the uncertainties are considered and can be propagated throughout the system. In the end, we can see what the resulting uncertainties are and incorporate this information into our decisions. Risk in other domains is commonly assessed using financial values. Therefore, the economic models and methods coming from the business domain can be applied, e.g., the Gordon-Loeb model for optimal investments in cyber-security [34].

2.2 Types of Measurement Scales

In his fundamental paper about scales, Stevens [35] defined four principal types of scales which differ by their capabilities. These scales are the nominal, ordinal, interval, and ratio scales. Each of the types has a set of capabilities that allow certain operations. We include them here because, in later chapters, we often refer to these scales. Especially the ordinal scale is very prevalent in existing methods of risk assessment, being called “qualitative rating”. In this thesis, however, we mainly use ratio scales – the details will be explained in Chapter 4 and Chapter 5.

2.2.1 Nominal Scale

A nominal scale is a classification. It allows differentiating objects and to classify them. The only relation it defines is equality, or the opposite, inequality. Examples of this are types and groups of objects. For risk estimations, this would be a classification for relevant risks and irrelevant risks. This often concerns specific vulnerabilities: Is a specific type of software used or not? Is a particular update installed or not?

Another example would be types of attacks: Is our encryption scheme susceptible to replay attacks? The answer is either yes or no. There are no further distinctions. This plays a significant role in cyber-security, especially for vulnerabilities: either a system is susceptible to a vulnerability or not. There is not “half-vulnerability”. However, for example, the strength of an encryption scheme is another story. This would belong to higher scales like the interval or ratio scale.

2.2.2 Ordinal Scale

The ordinal scale defines an order relation between the groups of objects. This means that the operations greater than or less than are defined. This allows ordering objects, but neither does it define how much they differ nor does it define an exact value for the object. Examples for this are sizes of clothing: small, medium, large, x-large. This example illustrates an inherent flaw of ordinal scales: A typical medium-sized T-Shirt has a different size based on the continent and country. It could even happen that an L-sized t-shirt from Asia is smaller than an M-sized t-shirt from America. We see that such scales are highly subjective and coined by the cultural background.

Unfortunately, these types of scales are very prevalent in risk estimations. To define risk as low, medium, or high is a standard way of stating risks. Even the infamous FMEA uses it in its estimations by defining, e.g., ten classes of severity types. Although each of the classes can be distinguished and can be ordered, it cannot be measured by how far they are apart. In the related work, we describe other methods, standards, and techniques that use such ordinal scales. In the included publication [P9], we made a strong point against ordinal scales by describing 24 flaws and problems when using them for risk assessments.

2.2.3 Interval Scale

An interval scale makes it possible to calculate differences between the groups of objects. A famous example is a temperature value stated in Celsius or Fahrenheit. Such scales allow to calculate the differences, e.g., today it is +5 °C warmer than yesterday, but they do not allow ratios or multiplicative relations. If it has 3 °C today and 9 °C tomorrow, the statement “it is three times as hot” is not valid because the point for 0 and the advancement on the scales were chosen arbitrarily. However, if the temperature is stated in Kelvin, this is another story: Here, we have an absolute point for 0, so this belongs to the ratio scales.

2.2.4 Ratio Scales

The ratio scales define a multiplicative relation among the objects on the scale. So, in addition to the difference, also a ratio could be calculated. This is the scale with the highest capabilities. Typical measurement systems, like the SI-system, are following this scale. For example, the temperature scale in Kelvin is such a scale. It has an absolute point for zero, which allows statements like 200 °K is twice as hot as 100 °K. Other examples would be, e.g., speed, distance, force, weight, electric current. Money, attack frequencies, and probabilities are examples of ratio scales. This is why we used this scale for our risk assessment.

2.3 Probability Distributions

Estimations about risks contain huge uncertainty since these are based on judgments about possible future events. Such uncertainty can be modelled using probability distributions that describe the likelihood of values over a specific value range (also called the distribution support). In this section, some common probability distributions are described: The normal distribution, the lognormal distribution, the uniform distribution, and the modified PERT-beta distribution, which is the distribution that was used in this thesis for modelling risk judgments. Figure 2.1 shows some examples of these distributions.

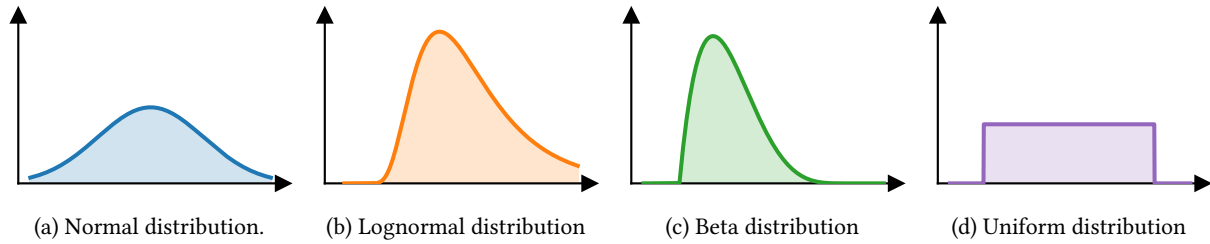


Figure 2.1: Examples for common probability distributions.

2.3.1 Normal Distribution

The normal distribution is the most commonly known and widespread distribution. It also goes under the names “Gauss-curve” or “bell-curve” due to its bell-like shape (see Figure 2.2). This distribution is a smooth, continuous distribution that has infinite support and is symmetric around the mean. Both sides of the distribution go until infinity, making it impossible to define a discrete maximum and minimum value. However, the likelihoods diminish the further away from the mean one gets. For example, after six standard deviations (6σ), the likelihood is already so small that it could be neglected for all practical purposes. However, for modelling risk judgments, it is more practical to take the 5% percentile as the minimum and the 95% percentile as the maximum. Thus, the estimated value range of an expert spans a probability space of 90%. By modelling it this way, there is a probability of 0.1 (10%) outside the assumed range, which goes on until infinity. Equation 2.2 shows the probability density function for the normal distribution:

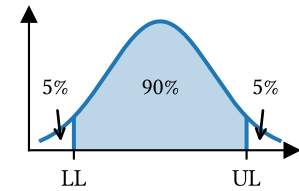


Figure 2.2: The normal distribution parametrized with a 90% risk interval between [LL, UL].

$$\mathcal{N}(x \mid \mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2.2)$$

Parameterisation Using Risk-Intervals: The normal distribution is parametrized using μ and σ . To model a risk estimation given by a lower limit (LL), the upper limit (UL) and confidence, we have to parametrize the normal distribution. Lower and upper bounds are projected onto points on the distribution, which reflect the respective confidence intervals. For example: If we have a 90% confidence, the lower bound represents the 5% percentile, and the upper bound represents the 95% percentile. Hence, the area inside represents a 90% confidence interval. Since the normal distribution is symmetric around the mean, which is the average of the lower and upper bounds, it already is fully specified, and the

mode estimation has to be dismissed since it is not applicable. The normal distribution does not allow further shape modifications. We can calculate the parameters as follows:

$$\mu = \frac{UL + LL}{2} + LL, \quad \sigma^2 = \frac{UL - LL}{P(-z\sigma \leq Z \leq z\sigma) = c} \quad (2.3)$$

where: UL ... is the upper limit of the estimation (e.g., 14).

LL ... is the lower limit of the estimation (e.g., 86).

c ... defines the confidence interval of the estimated values (e.g., 0.9 for 90%).

P ... is the cumulative distribution function $P(X \leq x^*) = p$ or shortly $F_X(p)$.

Example An expert states the 90% confidence interval for a value between 14 and 86, and the probability is normally distributed: $X \sim \mathcal{N}[14, 86](90\%)$. From that input, we can calculate the parameters for the normal distribution as follows:

$$\mu = \frac{86 + 14}{2} + 14 = 50, \quad \sigma^2 = \frac{86 - 14}{F_X(0.95) - F_X(0.05)} = \frac{72}{3.2897} = 21.9 \quad (2.4)$$

2.3.2 Uniform Distribution

The uniform distribution is also very commonly used. Its speciality is that it has the highest entropy a probability distribution can have – all values inside the range are equally likely (see Figure 2.3). Other aspects are that it is bounded on both sides. Therefore, the lower and upper limit can be taken as absolute limits, making the confidence parameter not applicable. A uniform distribution has no mode since all values are equally likely. Thus, the mode is also not applicable here, and the lower and upper limit can fully specify the uniform distribution.

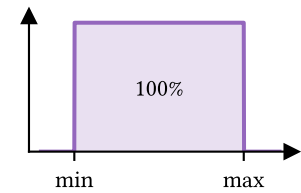


Figure 2.3: The uniform distribution.

$$\mathcal{U}(x | min, max) = \begin{cases} \frac{1}{max-min} & min \leq x \leq max \\ 0 & otherwise \end{cases} \quad (2.5)$$

Parametrization Using Risk-Intervals The parametrization of the uniform distribution for estimations of risk intervals is straightforward: The lower limit and the upper limits are precisely the parameters min and max for the distribution.

2.3.3 Lognormal Distribution

The lognormal distribution is included because it is well suited for estimations with long tails, e.g., modelling black-swan events. This can be seen in Figure 2.4. Due to this, it is commonly used in expert judgments. Here, the lower limit is bounded, while the upper limit goes until infinity. Here, also the limits must be projected onto an appropriate confidence interval, like in

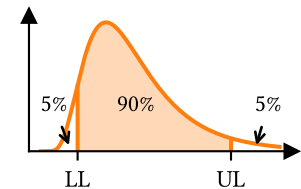


Figure 2.4: The lognormal distribution parametrized with a 90% risk interval between [LL, UL].

the normal distribution. Depending on the confidence in the judgment, the respective percentiles are chosen, e.g., 5% and 95% for a 90% confidence interval.

$$\mathcal{LN}(x | \mu, \sigma^2) = \frac{1}{x\sigma\sqrt{2\pi}} \cdot \exp - \frac{(\ln x - \mu)^2}{2\sigma^2} \quad (2.6)$$

Parameterization Using Risk-Intervals The lognormal distribution is parametrized using μ , and σ . To model a risk estimation, we have to project the estimated lower bound, mode, upper bound, and confidence as follows: The lower bound is the shift parameter *loc*, the upper bound is represented by the respective confidence interval, e.g., for a 90% confidence range, the upper bound represents the 90% percentile.

From that definition, we calculate the parameters as follows:

$$\mu = \frac{\log(UL) + \log(LL)}{2} + \log(LL), \quad \sigma^2 = \frac{\log(UL) - \log(LL)}{P(-z\sigma \leq Z \leq z\sigma) = c} \quad (2.7)$$

where: *UL* . . . is the upper limit of the estimation (e.g., 14).

LL . . . is the lower limit of the estimation (e.g., 86).

c . . . is the confidence level of the estimation (e.g., 0.9 for 90%).

P . . . is the cumulative distribution function $P(X \leq x^*) = p$ or shortly $F_X(p)$.

Example An expert states the 90% confidence interval of a lognormal distributed value to be between 14 and 86: $X \sim \text{Lognormal}[14, 86](90\%)$. From that input we can calculate

$$\mu = \frac{\log(86 - 14) + \log(14)}{2} + \log(14) = 3.5467, \quad \sigma^2 = \frac{\log(86 - 14)}{P(X \leq x^*) = 0.9} = \frac{4.28}{3.2897} = 0.5518 \quad (2.8)$$

2.3.4 Modified PERT-Beta Distribution

The modified PERT distribution is a very versatile distribution based on the beta distribution. It has finite lower and upper bounds, like the uniform distribution or the triangle distribution. Furthermore, it has a mode, and its shape is adjustable, making it more versatile than the uniform and the triangle distribution. Figure 2.5 shows an example. The PERT distribution was the primary distribution used for all surveys and experiments in this thesis. We decided for the PERT distribution because of its suitability for expert judgments. PERT (Project Evaluation and Review Technique) originated in project management and was first used in 1959 by the US Navy to estimate and evaluate time plans and progress for their missions [36]. Furthermore, the FAIR method recommends using the PERT distribution for risk estimations [37, 38]. Since RISKEE derived the concepts of uncertain expert judgments from FAIR, we also inherited the usage of PERT distributions and extended it by using Kernel Density Estimation after each arithmetic operation with uncertain estimates. Therefore, PERT is only used on the first estimates

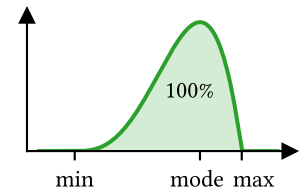


Figure 2.5: PERT Distribution parametrized with risk intervals [min, mode, max].

– internally RISKEE uses non-parametric kernel density estimations. Equation 2.9 shows the mathematical expression of the PERT distribution and Equation 2.10 shows the calculation of the parameters α and β .

$$\text{PERT}(x \mid \text{min}, \text{mode}, \text{max}, \lambda) = \frac{(x - \text{min})^\alpha \cdot (\text{max} - x)^\beta}{\text{B}(\alpha + 1, \beta + 1) \cdot (\text{max} - \text{min})^{\alpha + \beta + 1}} \quad (2.9)$$

$$\text{with } \alpha = \frac{\lambda \cdot (\text{mode} - \text{min})}{\text{max}}, \beta = \frac{\lambda \cdot (\text{max} - \text{mode})}{\text{max} - \text{min}}, \text{ and } \text{B}(\alpha, \beta) = \frac{\Gamma(\alpha) \cdot \Gamma(\beta)}{\Gamma(\alpha + \beta)} \quad (2.10)$$

The modified PERT distribution was used because it had several benefits which are useful for expert judgment and estimation of uncertain values:

- *It has a limited support range:* The PERT distribution has a finite lower and upper limit, which makes it much easier to take representative samples from it. Sampling is used to perform arithmetic operations on the distributions and propagating the uncertainties throughout the risk model.
- *It allows for an arbitrary finite range:* The PERT distribution allows positive and negative range values, which make it quite versatile.
- *The parameters are intuitive for humans:* Stating a three-point estimate using the lower limit, the upper limit, and the mode is quite intuitive for humans to understand and estimate, compared to other more abstract values like, e.g., the standard deviation, or shape and curvature parameters.
- *Polymorphic Shape:* The shape (skewness and kurtosis) of the distribution adapts appropriately to the parameters and can take on the same versatile shapes as the Beta-distribution, which it is based on.
- *It has a particular parameter for the certainty:* The λ -parameter in the modified PERT distribution allows adjusting the kurtosis of the curve according to the certainty of a given mode value. If an expert is very confident, then the λ value could be increased; if he or she is uncertain, the λ value could be decreased. This results in a more densely packed area around the mode or a more shallow curve. A λ of 0 would correspond to the uniform distribution.
- *Easy implementation and integration:* Since it is based on the beta distribution, it can be easily implemented and integrated into existing software.
- *The simple parametrization* with the parameters *min*, *mode*, and *max* makes it ideal for expert judgements over an arbitrary value range – useful for percentages, money, frequencies, or any other numeric judgement.

Parametrization Using Risk-Intervals The PERT distribution can take on many shapes, as can be seen in Figure 2.6. It is parameterized by minimum, maximum, mode, and confidence. With these four parameters, the respective α and β values for the underlying beta-distribution can be determined (see Equation 2.10), which is then scaled and translated to a distribution that is bounded by the minimum and maximum limit and has its highest probability at the mode, with broader or narrower kurtosis depending on the confidence. Equation 2.9 shows the probability density function for PERT, and Equation 2.10 shows how the parameters α and β for the Beta-function B are calculated [39].

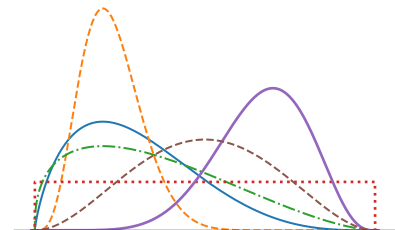


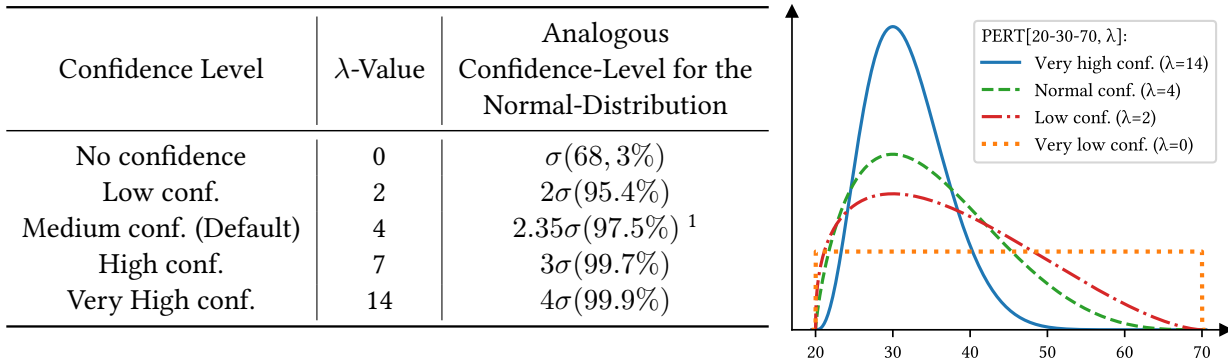
Figure 2.6: Examples for different shapes of the PERT distribution.

Example Using the PERT distribution, an expert could state a judgment like this: The value surely is between 14 and 86 and will most likely be 70 (with medium confidence).

$$X \sim PERT[14, 70, 86](\lambda = 4)$$

$$\alpha = \frac{4 \cdot (30 - 14)}{86} = 3.\dot{1} \approx 3.11, \quad \beta = \frac{4 \cdot (86 - 30)}{86 - 14} = 0.\dot{8} \approx 0.89 \tag{2.11}$$

Defining the Confidence Parameter Two aspects characterise confidence in a judgment: Firstly, the range between *min* and *max* defines a basic notion of confidence: the narrower the range, the higher the confidence. Secondly, the parameter λ additionally states the confidence in the mode value. The higher the λ value, the more emphasis is set on the mode value, which results in a narrower curvature (higher kurtosis) around this point. Lower values for λ flatten the curve, while higher values concentrate it around the mode. Figure 2.1b shows examples of this effect for the same judgment with different confidence levels. For example, the lowest possible confidence of $\lambda = 0$ completely dismisses the *mode* parameter resulting in a uniform distribution, where every value between *min* and *max* has the same likelihood. In contrast to that, a high confidence of, e.g., $\lambda = 7$ means, that the *mode* is weighted seven times as high as the *min* and *max* values. However, even knowing this, the values for λ are not intuitively explainable. Therefore, we proposed five qualitative classes to make it easier for the expert to state the confidence in their judgments: No confidence, Low confidence, Medium confidence, High confidence, Very high confidence. The classes are listed Table 2.1a. Being able to change the kurtosis around the most likely value is a useful feature of the PERT distribution and would not be possible in, e.g., the triangle distribution. Furthermore, the PERT distribution is smooth compared to the triangle, which always has an abrupt bend at the mode value that makes it not differentiable at this point. These features make the PERT distribution superior compared to the classic triangle distribution.



(a) Proposed qualitative classes for stating the confidence parameter λ in the modified PERT distribution.

(b) Examples for the same PERT distribution [20-30-70] with different confidence levels.

Table 2.1: Illustrations of the proposed qualitative values for the confidence value λ in a PERT distribution.

¹This value corresponds to the FWHM (Full-Width-at-Half-Maximum).

2.3.5 Kernel Density Estimation

Kernel Density Estimation (KDE) is a technique to approximate arbitrary distributions by combining multiple simple distributions, so-called kernels. This allows for representing arbitrary distributions – even multimodal distributions. Figure 2.7 shows an example for this. A difficulty here is deciding the number and shape of kernels to use (determined by a parameter called “bandwidth”). Too many or too narrow kernels result in overfitting, and too few or broad kernels smooth out the result, thereby averaging out the probabilities and imposing the underlying shape of the kernel. The default “rule of thumb” selects the bandwidth based on the number of data points (e.g., Scott’s rule [40]), but there are also other means to determine the optimal number, e.g., via cross-correlation or using plug-in methods. The goal is to approximate the target distribution smoothly. There are several optimizations to speed up the computation, e.g., different bandwidth estimations [40, 41], linear binning [42, 43], and convolution [44]. Depending on the used kernel, the resulting estimation range has to be truncated. This can be done by reflecting, truncating, and normalising the remainder (see Section 4.4.2 for a detailed description).

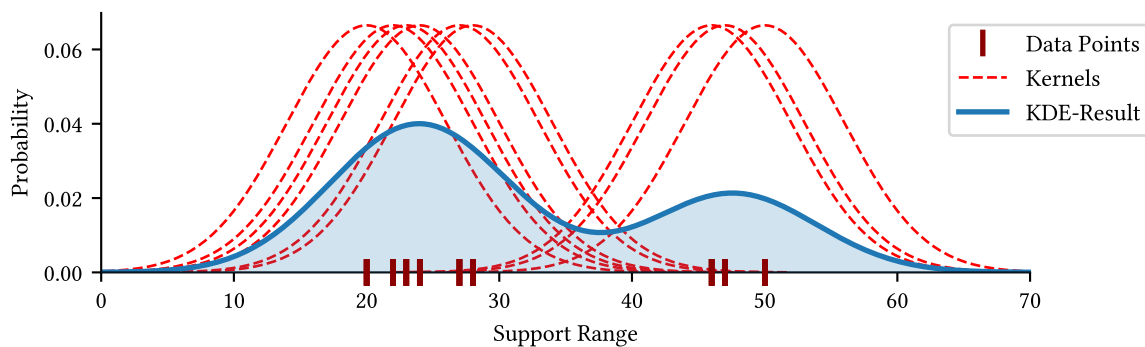


Figure 2.7: Kernel-Density-Estimation

2.3.6 Using Probability Distributions for Estimations

There are different ways how to specify risk attributes, ranging from single point estimates, to fully fledged probability distributions. Figure 2.8 depicts some examples for estimates, which are mentioned here.

- **Single point estimates** are the simplest of all estimations, consisting of a single numeric value representing mostly the mean or the mode. The problem here is that such estimates do not consider the uncertainty, which makes them completely unsuitable for risk estimations. Risk involves high uncertainty, which must not be neglected.
- **Range estimates (interval arithmetic)** are specified by a minimum and a maximum value, spanning up an interval (hence interval arithmetic). There is no explicit assertion about the values in between. Therefore, a uniform distribution is assumed, meaning that all values in between are equally likely – the width of the interval models the uncertainty.
- **Three-point estimates** specify the minimum value, the maximum value, and the most likely value in between (the mode). For example, the triangle distribution allows modelling such estimates. Sometimes this also specified by stating three or more percentiles, e.g., the 5%, the 50%, and the 95% percentile, assuming a linear interpolation in-between.

- **Histogram** is an estimate which specifies the probability of defined subdivisions (bins) over a specific range. Most often, these subdivisions are equally distributed, but this is not necessarily the case. A histogram can model any distribution, but depending upon the granularity of the subdivisions, this can be more or less accurate due to digitisation errors.
- **Probability distributions**, or more specifically, continuous probability distributions are mathematical functions that fulfil specific properties: probabilities must be non-negative, and the total area under the curve must sum up to 1 (representing 100%). Many functions fulfil these criteria, e.g., the normal distribution, lognormal, uniform, beta, gamma, and PERT, which is used in this thesis. Kernel density estimations (KDE) also fall into this category. They are superimposed combinations of probability distributions (kernels), e.g., multiple Gaussian kernels. See Section 2.3.5 for more information about KDEs.

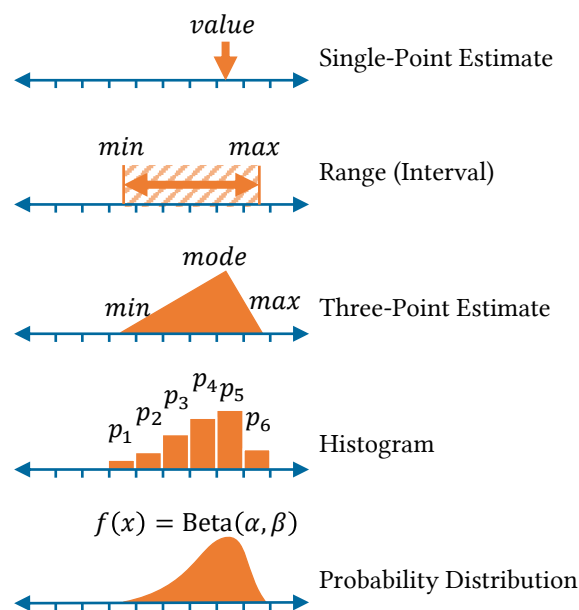


Figure 2.8: Different types of estimates: single point, range, three-point, histogram, and using a probability distribution function.

2.4 Gaussian Error Propagation: The Law of Propagation of Uncertainty

Another approach for calculating probability distributions is using descriptive statistics and applying the rules for Gaussian error propagation. It is done by assuming the error is normal-distributed and approximated by the first-order Taylor expansion (so the error behaves linearly around the value). This is applicable when the distribution is distributed like a normal distribution; the error is small compared to the estimated value; and that the value is not near 0 because there, the impact of the error becomes very high. To do it correctly, this approach needs automatic differentiation and symbolic references. It assumes that the error can be linearly approximated in close vicinity to the value and that the value function is differentiable everywhere on the support range.

In early implementations, we used this to calculate the uncertainty propagation. However, in later iterations, we dismissed this approach again because we had to consider different distributions (like the

PERT distribution), and therefore, Gaussian error propagation was no longer applicable. In addition to that, it does not work with multimodal distributions, and it suffers problems when the mean revolves around zero because the ratio of error to value gets very high there. Since risk estimations are based on expert judgments, which often are multimodal and non-Gaussian, Gaussian error propagation was not considered appropriate for RISKEE.

The Law of Propagation of Uncertainty (LPU) defines how the error propagates mathematically through calculations. Therefore it is often also called the Law of Propagation of Error [45]. The most generic formula for error propagation is as follows [46, 47]:

$$u(f) = \sqrt{\sum_{i=1}^n \sum_{j=1}^n \frac{\partial f}{\partial x_i} u(x_i) \cdot \frac{\partial f}{\partial x_j} u(x_j) \cdot r(x_i, x_j)} \quad (2.12)$$

where: $u(f)$... is the combined uncertainty for function f .
 f ... is an arbitrary differentiable function with n components: $f(x_1, \dots, x_n)$.
 x_i ... the respective component for the function f , which could also be a subfunction.
 $\partial f / \partial x_i$... is the partial derivative of f in respect to component x_i .
 $r(x_i, x_j)$... is the correlation coefficient between the components x_i and x_j .
 $u(x_i)$... is the respective uncertainty for component x_i .

If all variables are independent, we can neglect the correlations, which means that $r(x_i, x_j) = 0$; except for $i = j$, there $r(x_i, x_j) = 1$. Which means, that a variable only correlates with itself, but not with others. This neglects most of the summation terms except for the ones where the component is combined with itself, hence squared:

$$u(f) = \sqrt{\sum_{i=1}^n \left(\frac{\partial f}{\partial x_i} \right)^2 u(x_i)^2} \quad (2.13)$$

Assuming that all components are independent primitive variables, and substituting the uncertainty-function $u(f)$ by the standard deviation σ , it becomes the commonly known and used formula for the Gaussian error propagation:

$$\sigma_f = \sqrt{\left(\frac{\partial f}{\partial x_1} \right)^2 \sigma_1^2 + \dots + \left(\frac{\partial f}{\partial x_n} \right)^2 \sigma_n^2} \quad (2.14)$$

where: σ_f ... is the combined standard deviation for function $f(x_1, \dots, x_n)$.
 σ_i ... is the standard deviation for the component x_i .

2.5 Opinion Pools

Here, the mathematical methods of combining multiple expert judgments are discussed. The general expression for combining expert judgments is called **opinion pool** [48], and there are many ways to

do this. Anthony Cox [49, 50, 51] has analysed several different variants and concluded that they are more or less all equivalent in the sense that there is no single variant that is consistently better or worse than the others. They all have their limitations, and when used with manual weights, their differences diminish. However, the most common ways to calculate a mean are the arithmetic mean, the geometric mean, and the harmonic mean (these three are also called the Pythagorean means). They stem from the same family of means: the Generalised Means (also called Hölder-means, or power-means) [52]. The base formula for all of them is shown in Equation 2.15.

$$M_p(x_1, \dots, x_n) = \left(\frac{1}{n} \sum_{i=1}^n x_i^p \right)^{\frac{1}{p}} \quad (2.15)$$

They are called power-means because they all derive from the same expression taken to the power p . For example, M_1 is the arithmetic mean: $\frac{1}{n}(x_1 + \dots + x_n)$, and M_2 is the root mean square: $\sqrt{\frac{1}{n}(x_1^2 + \dots + x_n^2)}$. M_p is generally called the Hölder-mean with power p . Table 2.2 shows some more examples including their derivations. Means with higher powers like, e.g., the cubic mean or the quadratic mean, emphasize larger values, making them more prone to large outliers. Lower power means like the geometric mean, or harmonic mean emphasize smaller values more and are robust to large outliers but prone to small ones. They are more suitable for relative values, which may be multiplied, e.g., percentages, speedup, or relative performance. The arithmetic mean treats all values equally, which means that small and large outliers can distort the mean. The arithmetic mean is well suited for values summed up to a total value (e.g., money, length, or size). However, there are means which cannot be described using power-means, e.g., the logarithmic mean. The family of Stolarsky-means would already include those [53], but there is an even bigger generalization: the generalized f-Means [52].

2.5.1 Generalized f-Means

Generalized f-Means (also called Kolmogorov means [52], or quasi-arithmetic means) are further generalizations of the well-known family of Pythagorean-means [54], power-means [55], and Stolarski-means [53]. They work by using a function for projecting the individual elements onto an aggregation space, summing them up and averaging them, and then transforming it back into value space by using an inverse function. F-Means are defined as follows [56]:

$$M_f(x_1, \dots, x_n) = f^{-1} \left(\frac{1}{n} \sum_{i=1}^n f(x_i) \right) \quad (2.16)$$

We can derive the well-known arithmetic, harmonic, geometric, and other means from this simple definition. Table 2.2 shows some derived means.

Mean	Function f	Inverse f^{-1}	Result	Hölder	Stolarsky
Quadratic	$f(x) = x^2$	$f^{-1}(x) = \sqrt{x}$	$\sqrt{\frac{1}{n} \sum_{i=1}^n x_i^2}$	M_2	-
Square Root	$f(x) = \sqrt{x}$	$f^{-1}(x) = x^2$	$(\frac{1}{n} \sum_{i=1}^n \sqrt{x_i})^2$	$M_{1/2}$	$S_{1/2}$
LogSumExp	$f(x) = \exp x$	$f^{-1}(x) = \log(x)$	$\log(\frac{1}{n} \sum_{i=1}^n \exp x_i)$	-	S_0
Arithmetic	$f(x) = x$	$f^{-1}(x) = x$	$\frac{1}{n} \sum_{i=1}^n x_i$	M_1	S_2
Geometric	$f(x) = \log(x)$	$f^{-1}(x) = \exp x$	$\exp(\frac{1}{n} \sum_{i=1}^n \log(x_i)) = \sqrt[n]{\prod_{i=1}^n x_i}$	M_0	S_{-1}
Harmonic	$f(x) = 1/x$	$f^{-1}(x) = 1/x$	$\frac{n}{\sum_{i=1}^n 1/x_i}$	M_{-1}	-
Power	$f(x) = x^p$	$f^{-1}(x) = \sqrt[p]{x}$	$\sqrt[p]{\frac{1}{n} \sum_{i=1}^n x_i^p}$	M_p	-

Table 2.2: Examples for the parametrization of generalized f-means and the Hölder-mean or Stolarsky-mean they represent.

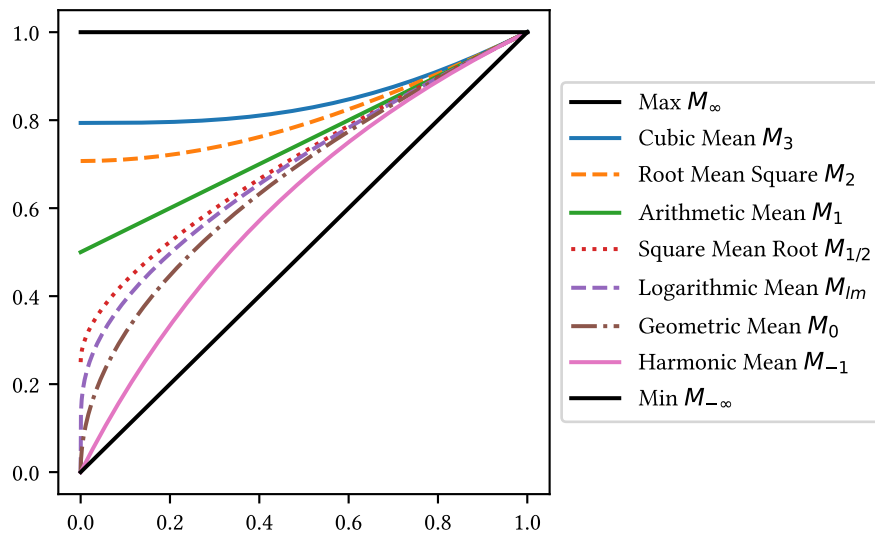


Figure 2.9: Examples for some variants of means for two values ranging from 0 to 1. This also showcases the sequence of inequalities: Means with smaller powers are always lower or equal than means with higher powers.

Related Work on Risk Assessment in Cyber-Security

If I have seen further, it is by standing upon the shoulders of giants.

Sir Isaac Newton, 1676

Summary: *This chapter describes the related work and state-of-the-art, including industry standards for risk analysis in cybersecurity. It shows an overview of the relevant topics in the field and details on the topics specifically relevant to this thesis. This includes cyber-security, risk management, expert judgment, metrology, as well as uncertainty propagation.*

◇◇◇

Risk analysis is an extensive topic covering many areas like business, life, technology, health, and society. In this chapter we give an overview over the related work in industry as well as academics, starting with general risk management and risk assessment, and diving into cyber-security later on. Moreover, the academic endeavours related to the topic and contribution of the thesis are described. We also describe and compare the differences to our developed method and framework, which is called RISKEE. RISKEE refines ideas from many existing methods. Specifically, the diamond model by Caltagirone et al. [57], the FAIR model by Freund and Jones [37], structured expert judgment by Cooke et al. [14], and the IDEA protocol by Hemming and Burgman [58]. In the following sections these related topics are described.

3.1 RISKEE and Graphical Models

RISKEE uses a graphical model for representing the attacks in a system. Graphical models define a system using nodes and edges which connect the nodes. The semantic meaning is defined differently in many systems, e.g., in fault trees [59, 60], nodes represent components that could fail and have some consequences on other components. In attack trees [61, 62], and Bayesian attack graphs (BAGs) [16], nodes also represent components or security barriers that have to be compromised or broken by an attacker to get to the subsequent components. Poolsappasit et al. [63] judge the probability of vulnerability exploitation and the respective loss or gain for every node and define security controls that lower the probability of exploitations. Their method allows for hypotheses and inference, given

some intrusion evidence or changes in the system. With the probabilities and loss/gain information on the nodes, they calculate the minimal set of mitigations, which result in the maximum reduction of loss.

In 2019, Wolthuis et al. [64] also published a graphical approach for a threat model based on a Bayesian belief network (BBN) with quantitative input values. Where data is not available, they use expert judgment to get the data. They also model the threats using an attack graph, which consists of input (or root) nodes, intermediate nodes, and result nodes. They also defined a process called “The Quantified Risk Methodology” to establish the attack graph and the values, consisting of nine steps and incorporating expert judgment. They use a three-point estimate using high, average, and low values for each identified threat actor. In such a way, they also include the uncertainties of expert judgments. While the impact values are assumed to be time-invariant, they recommend frequently updating the likelihood attribute. They use their approach for scenario analysis, sensitivity analysis, and root cause analysis. The whole approach was published at the same time as RISKEE was published and has some similarities, but also differences. The most important being that RISKEE uses structured expert judgment involving calibration of the experts and only concentrates on assessing the risks, not on root cause analysis, cyber-crime investigation or digital forensics. Furthermore, the method proposed by Wolthuis addresses a higher level of abstraction for policymakers and national security standards, while RISKEE addresses risks on an organization level.

Wu et al. [65] also describe a methodology for assessing risk in cybersecurity of cyber-physical systems. Their method is asset-based and works by evaluating the risk of attacks for individual nodes in a system. They assess the severity, probability, and consequence for each attack and node. These three attributes are broken down into several sub-factors derived from CVSS as the basis for their calculations. The drawback of their method is that CVSS only lists the known vulnerabilities, not the unknown ones, which is neglecting a huge part of the risks. RISKEE, in comparison, can also model the unknown risks or risks of attacks that have not yet been found. However, they use time series for their estimations, which allows for calculating a so-called “risk-change-curve” showing how the risks change over the years. This is something that would also be possible in RISKEE but is not implemented yet.

In RISKEE, we used the Diamond model by Sergio Caltagirone, Andrew Pendergast and Chris Betz [57] for the graphical model of a system. Here, nodes represent an attack or attack step, which is more versatile than being restricted to the network topology or structural view of a system because it models the behavioural aspect and sequence of attacks. The diamond model is intended to establish a conceptual foundation for a graphical model of attacks that other methods can refine and use. The model emphasizes the extensibility and modifiability in its definition – an ideal basis for advanced risk assessment models. We picked up the model and extended it by three specific attributes: *frequency*, *vulnerability* and *impact*. This is described in more detail in the included publication [P4]. The benefit is that these attributes allow concrete and mathematical definitions and calculations of the resulting risk instead of just having ordinal ratings. Here, the difference between attack trees and Bayesian attack graphs is recognizable: By incorporating the attack frequency and the impact, the result can be measured in comparable units of measurement, while BAGs only involve the probability of attacks, which could be deceiving. For example, when an attack is highly probable but the impact is neglectable. Furthermore, the diamond model describes many other attributes that help determine these three risk attributes, e.g., a description of the adversary and the victim, including the capabilities and infrastructure available to both. While these attributes help describe attacks, we did not elaborate on an exact ontology or mathematical equations to calculate the risk attributes – for this, we rely on structured expert judgment. The diamond model itself is described in Section 3.6a, and the extended risk attributes are described in Section 4.2.2.

3.2 **RISKEE and the FAIR Method**

The FAIR method by Jack Jones, Jack Freund and the Open Group [37, 66, 67] establishes an ontology for quantitatively assessing risks in cyber-security. Its definitions and mathematical models served as inspirations for the used attributes and equations used in *RISKEE*. More details about FAIR are described in Section 3.7.3. We refined and simplified the ontology and applied it to risk graphs. The benefit of applying it to risk graphs is to allow the definition of multi-step attacks in much more detail instead of combining all the probabilities into one event. The main difference here is that the vulnerability is distributed over multiple nodes in a risk graph instead of determined by a single comparison of the attacker and defender strength (in FAIR terminology: threat capability and resistance strength). Another change is the definition of the impact magnitude: In *RISKEE*, it is defined as a single distribution that includes the different sources of impact, while the FAIR model distinguishes between primary and secondary impact with separate frequencies and impact magnitudes. Furthermore, the attack frequency is stated directly in *RISKEE*, without splitting it into more detailed subfactors. Albeit these simplifications, *RISKEE* still allows splitting the impact into primary and secondary and the frequency into its subfactors; hence, it is still compatible with the FAIR model. Another aspect inherited from the FAIR model was to use the PERT-distribution [68] for modelling the uncertain estimations. This aspect especially affected the way we implemented the expert judgment. Nate Silver describes how important it is to model the uncertainty using statistics [69].

3.3 **RISKEE and Structured Expert Judgment**

Structured expert judgment (SEJ), also known as the classical model by Roger Cooke [14, 70, 11] is a method for assessing uncertain estimations by combining the estimations from multiple experts using a weighted linear opinion pool. The weights are based on the performance of the experts on specific calibration questions. We specialized this method for judging the three previously mentioned risk attributes: *frequency*, *vulnerability*, and *impact* in a risk graph. This allows us to apply it to the already established risk graph to determine the risk attributes. Furthermore, we adapted it to use the PERT distribution instead of uniformly distributed quantiles. This allows a more sophisticated definition of the underlying probability distributions. The PERT distribution allows for stating a minimum, maximum, the most-likely value, and the confidence in this most-likely value. The idea to incorporate the PERT distribution in the expert judgment came from the previously described FAIR model.

3.4 **RISKEE and the IDEA Protocol for Expert Elicitation**

The IDEA protocol by Victoria Hemming and Mark Burgman [58, 71] is a procedure for doing structured expert judgment in an efficient and useful way. It is based upon the Delphi method [72], consisting of multiple rounds of expert judgment with discussion and knowledge exchange in between. All this has the purpose of eliminating human bias and minimizing the estimation uncertainty [73, 74]. Ultimately, this elicitation will establish the information basis for decision-making, and therefore must be reliable and defensible [13, 75]. The IDEA protocol also incorporates a calibration phase, needed in structured expert judgment. We adapted this protocol to include *RISKEE* as the tool for modelling the estimations and visualizing the results. Furthermore, we specifically apply risk graphs in the protocol to assess the mentioned risk attributes. This specialization in risk scenarios makes it straightforward to use. This is elaborated in much more detail in the design chapter in sections 4.1, and 4.5, as well as in the second part of the evaluation (Section 6.2).

3.5 RISKEE and Qualitative Assessments

RISKEE exclusively uses quantitative assessments. This stands in strong contrast to the plethora of qualitative assessments used by many commonly used risk assessment methods, like e.g., NIST CSF [76], SP-800-30 [77], FMEA [78], HARA [79], or CVSS [80]. In the following sections of the related work, these standards and methods are described to give an impression of the scales and ratings they use. It can be seen that the typical qualitative ratings are very prevalent and established amongst these methods. The problem here is that qualitative ratings that apply ordinal scales or risk matrices have many problems. Maybe the biggest is the illusion of usefulness due to its seemingly simple application. Also, the neglect of uncertainty, quantization errors, the undefined arithmetic on ordinal scales, and arbitrary thresholds of risk matrices are only a few examples of the problems they introduce. This is elaborated in more detail in the included publication [P9]. RISKEE differs from such qualitative assessment by demanding assessments on ratio scales that can be used for calculations and can be compared to determine the effects of mitigations and define thresholds of acceptable risks based on real values instead of imaginary and coarse scales.

3.6 Uncertainty in Distributed Control Systems

Distributed control systems cover many areas like control systems, embedded systems, or Cyber-Physical Systems (CPS). Classic literature about this topic is, e.g., “Distributed Systems” by Tannenbaum et al. [81], or “Distributed Systems Architecture” by Puder et al. [82]. According to Amorim et al., “CPS usually operate in uncertain environments and are often safety-critical” [83]. This dangerous combination of uncertainty and safety-critical devices could lead to threatening situations causing harm, injuries, and the death of humans, as well as substantial damage to properties and financial losses. According to the HAZOP model, data errors in control systems are categorized as *Provision Errors*, *Timing Errors*, and *Value Errors* [84, 85, 86]. To handle these kinds of errors, contracts can be used that define the requirements [87]. For example, in the ConSerts M model [83, 88], every component provides guarantees to the consuming components based on assumptions and requirements to the serving components. As long as the assumptions hold, the guarantees must also hold. In such a way, it is possible to define safety contracts at design time which are evaluated at runtime and acted upon via safety mechanisms through self-adaptive systems [89]. The guarantees define some hard limits wherein a system can operate or define soft limits and continue working with some penalties and degraded functionality. Thus, a model of uncertainty is needed, which could be defined similarly to the RISKEE graphical model within this thesis. A graph of uncertain contracts could model a distributed system. Similar to the HAZOP fault propagation mentioned at the beginning of this section, the uncertainties could propagate throughout the system. While such resilient and self-adaptive systems inspired some ideas in this thesis, we went into another direction by modelling the propagation of cyber-security risks in such systems.

3.7 RISKEE and Risk Management Standards

While RISKEE is a method for assessing cyber-security risks, the following list of risk management standards goes beyond pure risk assessment and the domain of cyber-security. We begin with general standards for risks in business and organization, then dive deeper into standards and methods used in safety. This is interesting because many security standards and methods are derived from safety. In the end, we describe the related standards commonly used in cyber-security. For conciseness, we only

mention the most prevalent and well-known standards. However, there are many more since nearly every domain has specific risk standards tailored to the individual requirements in that domain. The following section lists and describes several standards and methods for assessing and managing risks. Figure 3.1 shows an overview of them, grouped by the general domains. These standard and methods are described in the following sections.

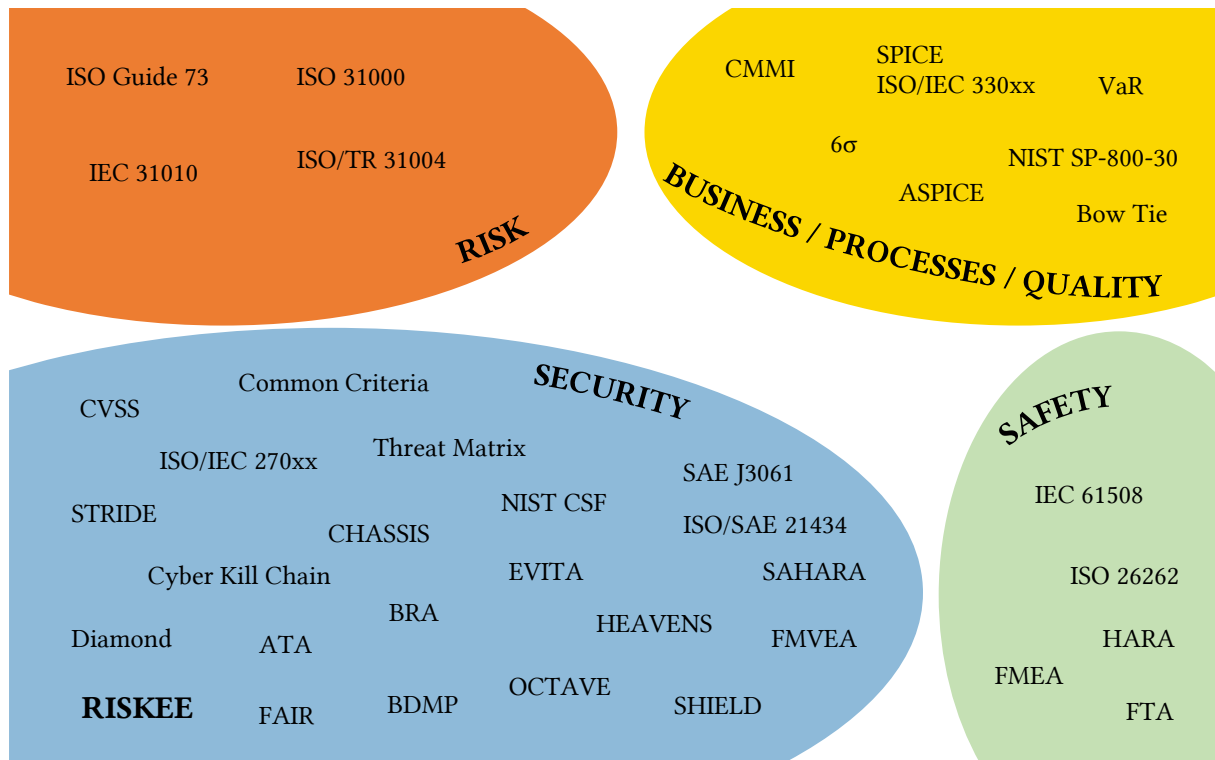


Figure 3.1: Overview over the relevant standards and methods for risk assessment, safety, and cyber-security.

Feature Overview Here, we compare the features of RISKEE to other risk assessment frameworks. These are compared based on different features like, e.g., the domain they are intended for, the metric they use, and the model they apply to map the system structure.

- **Domain:** The domain the framework comes from or is intended for (quality, safety, security, business).
- **Scale:** The type of scale used for evaluating the risk input values (nominal, ordinal, interval, ratio).
- **Structure:** How is the system reflected inside the risk model? As a plain list, attack paths, trees, or general graphs?
- **Speciality:** Here, specialities of the used framework are shortly mentioned. Further explanation is done in the respective section.

Product	Domain				Scale				Structure				Speciality
	Quality	Safety	Security	Business	Nominal	Ordinal	Interval	Ratio	List	Paths	Tree	Graph	
6σ [90]	●	●		●	●	●	●	●	●	●	●	●	Versatile
ISO 26262 [91]		●				●	●		●	●			Signal Path
HARA [79]		●				●			●	●			S, E, C
SAHARA [92]			●			●			●	●			R, K, T
FME(D)A [93]		●				●	●		●				Traceability
FMVEA [94]			●			●			●	●			Threat-Model
CVSS [80]			●			●			●				Factors
BRA [95]			●			●			●				Fast
CC [80]			●						●				Customized
HEAVENS [96]			●			●			●				Generality
OCTAVE [97]			●						●				Perspectives
NIST SP800-30 [77]			●			●	●		●				Widespread
ISO 31000 [23]				●					●				General
NIST CSF [76]			●						●				Widespread
ATA [62]			●					●			●		Attack Trees
FAIR [66]			●	●				●	●				Ontology
DIAMOND [57]			●									●	Graph
RISKEE [98]			●					●				●	Uncertainty

Table 3.1: Overview and comparison of related standards.

3.7.1 General Standards for Risk Management

ISO Guide 73 Risk management – Vocabulary The ISO Guide 73:2009 [22] is the standard for defining generic terms in risk management. It is relatively short, only having 15 pages. It contains the basic terminology to be used when talking about risks, e.g., risk, risk-management, risk-assessment, risk-matrix, events, hazard, likelihood, exposure, risk appetite/aversion/tolerance, or residual risk.

ISO 31000 Risk management – Guidelines The ISO 31000:2018 [23] is the standard for guidelines on risk management. On its 16 pages, it describes general principles and a framework, and a process on how to manage risks. Herein, risk assessment, risk identification, risk analysis, risk evaluation, and risk treatment and monitoring and review and reporting are described for general cases. Also belonging to this family of risk standards, the ISO/TR 31004 [99], and the IEC 31010 [100] refine the general principles of the ISO 31000. Especially the IEC 31010 defines risk assessment techniques in much more detail. It describes the usage of techniques for identifying risks, determining sources and causes, and dependencies, and how to evaluate and measure risk and contains many useful techniques and methods which can be applied, e.g., Ishikawa-diagrams, bow tie analysis, Bayesian networks, event-trees, fault-trees, Markov-diagrams, Value at Risk, ALARP, Pareto-charts, and risk-matrices.

NIST SP-800-30 Guide for Conducting Risk Assessments: The NIST SP 800-30 [77] is a guide which focusses on risk management and risk assessment. In the updated revision of the guide, the focus is shifted more towards conducting risk assessments than on managing risks. It builds upon the NIST SP 800-39 [101] by including three tiers of risk assessment in the process: Organizational tier, mission/business process tier, and information system tier. Furthermore, it specifies how to assess and categorize different aspects of risk: threat sources, threat events, vulnerabilities, predisposing conditions, security controls, and impact, determining the organizational risk. It proposes ordinal scales for all the aspects (e.g., the likelihood shown in Figure 3.2) and a risk matrix to combine them into a final risk value [102]. For example, the likelihood scale shown in Figure 3.2 states highly subjective definitions, e.g., moderate likelihood is when an adversary is somewhat likely to initiate a threat event. However, it does not define what *somewhat likely* means. The standard even includes cautionary notes that risk assessment is subjective and not a precise instrument of measurement [77].

Qualitative Values	Semi-Quantitative Values		Description
Very High	96-100	10	Adversary is almost certain to initiate the threat event.
High	80-95	8	Adversary is highly likely to initiate the threat event.
Moderate	21-79	5	Adversary is somewhat likely to initiate the threat event.
Low	5-20	2	Adversary is unlikely to initiate the threat event.
Very Low	0-4	0	Adversary is highly unlikely to initiate the threat event.

Figure 3.2: NIST SP 800-30 scale for the likelihood. It shows the coarse and uncertain definitions of the classes.

3.7.2 Risk Management in Safety

Risk management in safety has a different goal than for business: Instead of purely thriving for financial success, in safety, the goal is to protect humans from harm, danger, and possible injuries. Every domain has specific standards, e.g., the automotive industry has different standards than the industrial domain, aeronautics, critical infrastructure, or end-user products.

ISO 26262 Road vehicles – Functional safety The ISO 26262 is the standard for functional safety in the automotive domain. [91]. Here the safety level is called ASIL (Automotive Safety Integrity Level), which is QM, A, B, C, D, and builds upon the V-model as a process on three levels of abstraction: hardware, software, and system level. The ASIL level defines strict requirements for the lifetime of components measured in FIT (Failure In Time), based on the mean time between failures, which are only achievable using redundant signal paths, diverse components, and monitoring at runtime.

HARA The Hazard and Risk Analysis (HARA) is a method described in the ISO 26262 [79] and uses a threat-based approach to model risks. Based on assumed situations, the possible threats which could occur are collected and assessed for their severity, exposure, and controllability attributes based on ordinal scales. The scales and ratings are defined in the standard in more detail, with examples for each class. Table 3.3 shows the three tables for the classification of each parameter. The standard also defines that in border cases, the higher class should be chosen. The combination of these properties is the so-called Automotive Safety Integrity Level (ASIL), which has five levels: QM (quality management, lowest), ASIL A, ASIL B, ASIL C, ASIL D (highest). The determination matrix for the combination is shown in Table 3.2. Further on, the ISO 26262 defines measures and requirements, which have to be fulfilled for each of the ASILs. These requirements increase exponentially for each ASIL, e.g., the highest level, ASIL D, *requires* a failure rate lower than 10 Fit¹, while ASIL C requires the fit rate to be lower than 100 Fit and so forth. QM does not define a fit rate at all.

¹Fit (Failures in time) is a measure for the failure rate of a product and is defined with the base rate of one failure in a billion hours (failure rate of $10^{-9}h$) [79], which is approximately 1 failure in 114.000 years.

Severity	Exposure	Controllability		
		C1	C2	C3
S1	E1	QM	QM	QM
	E2	QM	QM	QM
	E3	QM	QM	A
	E4	QM	A	B
S2	E1	QM	QM	QM
	E2	QM	QM	A
	E3	QM	A	B
	E4	A	B	C
S3	E1	QM	QM	A
	E2	QM	A	B
	E3	A	B	C
	E4	B	C	D

Table 3.2: HARA risk matrix for the ASIL according to ISO 26262 [79].

Lvl	Severity
S0	No injuries
S1	light and moderate injuries
S2	severe injuries
S3	life-threatening or fatal injuries
Lvl	Exposure
E0	incredible
E1	very low probability
E2	low probability
E3	medium probability
E4	high probability
Lvl	Controllability
C0	controllable in general
C1	simply controllable
C2	normally controllable
C3	difficult to control or uncontrollable

Table 3.3: HARA assessment parameters severity, exposure, and controllability according to ISO 26262 [79].

FMEA / FMEDA The Failure Mode, Effects and Diagnostics Analysis (FMEDA) is one of the most popular techniques in safety to guarantee the fault-tolerance on component level, defined in the IEC 60812 [93]. Here, the analysis revolves around the intended functionality of a component and is a six-step process [78]: Scope Definition, Structure Analysis, Function Analysis, Failure Analysis, Risk Analysis, and finally Optimization. All thinkable failure modes of functionality are considered during these steps, and the reasons and detection and correction mechanisms are elaborated. To compare the risks of the different failure modes, each is assessed according to the following three attributes: the severity level, the occurrence frequency, and the detection mechanisms. While the severity and occurrence increase the risk, the detection attribute lowers it. All three attributes are assessed on a scale from 1 to 10, where the classes are defined in different ways: The severity is defined arbitrarily using semi-quantitative guidelines like, e.g., level 4 is when a failure is noticed by more than 75% of the customers but does not have further consequences, level 5 is when the comfort of a function is somewhat reduced, and the customer is slightly dissatisfied. The occurrence is defined using a logarithmic scale where the failure rate is approximately doubled or tripled between each level, e.g., level 5 specifies a failure rate of 2 out of 1000, level 6 means 5 out of 1000, and level 7 specifies 10 out of 1000 items. Detection is defined using an inverse Gaussian scale to divide the extreme values into more detailed classes, e.g., levels 1, 2, and 3 are about the highest detection chances, levels 4, 5, and 6 about mediocre chances and levels 7, 8, 9, and 10 subdivide the very low detection chances. They are combined by multiplying the attributes, and the resulting number is called the RPN (risk priority number).

$$RPN = S \times O \times D \tag{3.1}$$

where: RPN ... is the risk priority number between 1 (lowest) and 1000 (highest).

S ... is severity, between 1 (no effect) and 10 (hazardous).

O ... is the occurrence level between 1 (very low occurrence) and 10 (very high occurrence).

D ... is the detection level between 1 (always detectable) and 10 (absolute uncertainty, never detectable).

Even the standard itself states that this RPN is just for prioritizing the risks, and there is no threshold of the acceptable remaining risk. Newer versions fix this by using the so-called action priority [103]: a more complicated rule set, similar to a three-dimensional risk matrix, which defines fixed combinations of the three attributes (S, O, D) instead of simply multiplying them. The result is not an RPN anymore but a directive for further actions to mitigate the risk. This is divided into three levels: High means that appropriate actions must be taken to manage the risk, medium means such actions should be taken, low means they can be taken, but there is no obligation to [103].

FTA Fault Tree Analysis (FTA) was developed in the 1960s by H. A. Watson from Bell Laboratories for the U.S. Air Force to model possible faults and failures in their systems (aerospace) [104]. Since then, it was applied in many other domains and areas, e.g., chemistry, engineering, physics, computer science, or information science [59]. It is defined in detail in the Fault Tree Handbook [105, 106] and has its own standard, the IEC 61025 [107]. The idea is to model events that could lead to the failure of a system in a hierarchical tree. Events can be divided into sub-events, and conditions or combinations can be modelled using logic gates (used in electrical engineering), e.g., AND, OR, or XOR.

3.7.3 Risk Assessment for Cyber-Security

Many concepts and ideas for risk assessment can be carried over to cyber-security [108, 109]. However, here additional challenges arise that are not entirely solved yet: Attacks are non-deterministic due to the human nature of attacks and biased system evaluations. Cyber-Security incidents are not like failures of components due to wearing and ageing components or technical failures. In reliability engineering, technical failures are modelled using the famously known bathtub curve (based on the combination of three Weibull distributions for early failures, random failures, and wear-out failures at the end of the lifetime). However, for cyber-security, such a model does not exist (yet). Hacker attacks and cyber-security incidents have high uncertainty in the models and estimations, which must be considered. Furthermore, they can change over time and switch rapidly between extremes. For example, a newly found highly severe attack, potentially affecting millions of devices, could be mitigated entirely by a software update.

SAHARA The Security-Aware Hazard and Risk Analysis (SAHARA) is an adaption of the HARA method [79] that uses similar techniques and enhances it also to consider security issues [92, 110]. It uses ideas from STRIDE [111] to find security threats on the system design level. These threats are then assessed on three categories: the needed resources (R), the required know-how (K), and the threat-criticality (T). The combination of these assessments results in the so-called security level (SecL). If the SecL exceeds a certain threshold ($\text{SecL} > 2$), the underlying security threat must be included in the HARA, similar to a safety hazard, with a safety goal (security goal in this case) and measures to guarantee this goal. Table 3.5 shows the assessed parameters for each threat, and Table 3.4 shows the risk matrix used to combine these parameters into the Security Level.

Required Resources 'R'	Required Know-How 'K'	Threat Level 'T'			
		0	1	2	3
0	0	0	3	4	4
	1	0	2	3	4
	2	0	1	2	3
1	0	0	2	3	4
	1	0	1	2	3
	2	0	0	1	2
2	0	0	1	2	3
	1	0	0	1	2
	2	0	0	0	1
3	0	0	0	1	2
	1	0	0	0	1
	2	0	0	0	1

Table 3.4: SAHARA risk matrix for calculation of the security level, by Macher et al. [92].

Lvl	Required Resources 'R'
0	no or commodity tools
1	standard tools
2	simple tools
3	advanced tools
Lvl	Required Know-How 'K'
0	no prior knowledge needed
1	technical knowledge needed
2	domain knowledge needed
Lvl	Threat Level 'T'
0	no security impact
1	moderate security relevance
2	high security relevance
3	high security and possible safety relevance

Table 3.5: SAHARA assessment parameters by Macher et al. [92].

FMVEA The Failure Mode, Vulnerabilities, and Effect Analysis (FMVEA) is a method for analysing the security aspects of a system as an extension to an already existing safety analysis. FMVEA [112], is based on FMEA [113] and extends it by using STRIDE [111] to focus on the security domain and find additional failure modes that are caused by security attacks. It works by assessing the system susceptibility (attacker motivation and capabilities) and the thread properties (reachability, unusualness) on an ordinal scale ranging from 1-3. The resulting criticality is determined by combining these properties with the severity and failure probability (see Figure 3.3 for the risk matrix). Equations 3.2, 3.3, and 3.4 show the calculation of the subfactors. The resulting criticality is used for the prioritization of the threat modes.

System Susceptibility	Threat properties				
6	8	9	10	11	12
5	7	8	9	10	11
4	6	7	8	9	10
3	5	6	7	8	9
2	4	5	6	7	8
	2	3	4	5	6

Figure 3.3: The used risk matrix in FMVEA to evaluate the resulting risk.

$$\text{System Susceptibility} = \text{Motivation} + \text{Capabilities} \tag{3.2}$$

$$\text{Threat Properties} = \text{Reachability} + \text{Unusualness} \tag{3.3}$$

$$\text{Attack Probability} = \text{System Susceptibility} + \text{Threat Properties} \tag{3.4}$$

HEAVENS Security Model evaluates threats based on a multitude of input parameters and calculates a resulting security level (SL), which resembles the risk level [96], similar to the NIST SP800-30, FMEA, CVSS, or SAHARA. Its speciality lies in the fact that the standard tries to connect and be compatible with many other standards like OCTAVE, ISO 26262, Common Criteria, HARA, TARA, FMEA, and others. Many of the classifications were derived from other standards to make it easy to adopt the HEAVENS Security model. It is based upon estimating the threat level and the impact level based on several aspects like, e.g., the

Security Level (SL)	Impact Level (IL)					
	0	1	2	3	4	
Threat Level (TL)	0	QM	QM	QM	QM	Low
	1	QM	Low	Low	Low	Medium
	2	QM	Low	Medium	Medium	High
	3	QM	Low	Medium	High	High
	4	Low	Medium	High	High	Critical

Figure 3.4: HEAVENS security level risk matrix [96].

required expertise, the needed knowledge, the window of opportunity, the required equipment, and different dimensions of impact (safety, financial, operational, legislation). By estimating these aspects on ordinal scales and combining them using risk matrices, the HEAVENS security model comes up with a security level ranging from quality management (lowest) to critical (highest) to judge the criticality of cyber-security risks. The scales are defined in different ways: the attributes determining the threat have four linear levels; the attributes for the impact have four logarithmically scaled levels, and both are combined into the general aspect by adding up and applying to a risk scale. Both aspects are then applied to a risk matrix to figure out the final security level. Figure 3.4 shows the risk matrix for the security level.

Cyber Kill Chain The Cyber Kill Chain was developed by Lockheed Martin and is a method for modelling cyberattacks and finding possible mitigations [115, 114]. The model splits up attacks into seven different phases which are needed to execute a cyberattack: Reconnaissance, Weaponization, Delivery, Exploitation, Installation, Command and Control, and Actions on Objectives². Each of the steps is needed to execute a cyber-security attack successfully. The idea is to prevent the attack in the earliest phase possible to mitigate (to “kill”) the whole attack. This also makes it possible to compare different mitigation actions, which intervene in different phases of attacks, to decide for the most efficient. Figure 3.5 shows this with an example. It shows three different mitigation campaigns and the phases in which they can/could/should block a specific attack. The Cyber Kill Chain is part of a bigger method called *Intelligence-Driven Defense* [116], which tackles cyber-security in its entire organizational scope.

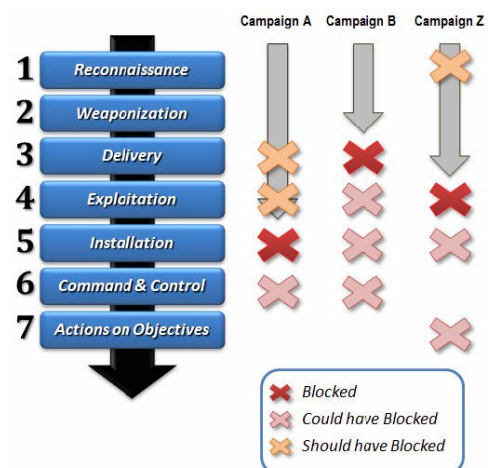


Figure 3.5: The Cyber-Kill-Chain by Lockheed Martin [114]. The figure shows the seven phases, and mitigation actions for three different campaigns.

ATA The Attack Tree Analysis (ATA) [117, 118] is a method using a graphical model based on attack trees by Bruce Schneier [62]. It has its origins in safety, especially fault-trees and fault-tree analysis (FTA) [60, 119, 120]. In ATA, the events are not a simple list, but they are arranged in a tree structure where the root node is the attacker’s goal, and the leaves are the steps needed to reach this goal. With every layer of the tree, the steps get more detailed and refined. The nodes in the attack tree can have specific attributes needed to analyse the tree for, e.g., the most feasible or dangerous attack paths. Using an attack tree provides an effective way to model the attack sceneries and quantify risk values in a Cyber-Physical System [121]. Liu used enhanced attack trees to assess the risk in Cyber-Physical Systems [122].

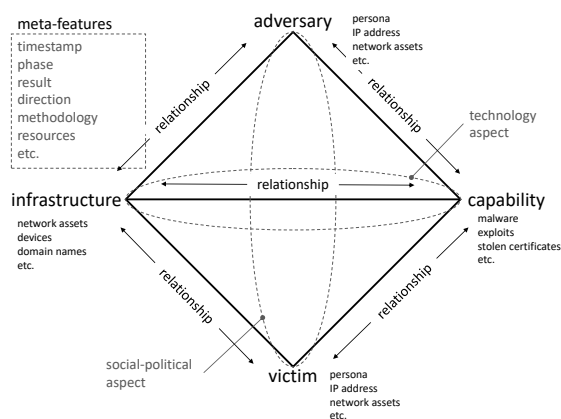
Diamond Model The *Diamond Model of Intrusion Analysis* [57] is a formal model to analyse cyber-security incidents. It is defined in very general terms to allow adaption and extension. The Diamond model describes core and meta-features for every attack event and how the events themselves are connected. In that regard, it resembles an attack tree [62], but with much more detailed definitions of the respective attributes. Going beyond attack trees, it is not limited to a tree structure but can consist of a whole attack graph, where the nodes can be aligned to several attack phases, e.g., as the kill

²<https://www.lockheedmartin.com/en-us/capabilities/cyber/cyber-kill-chain.html>

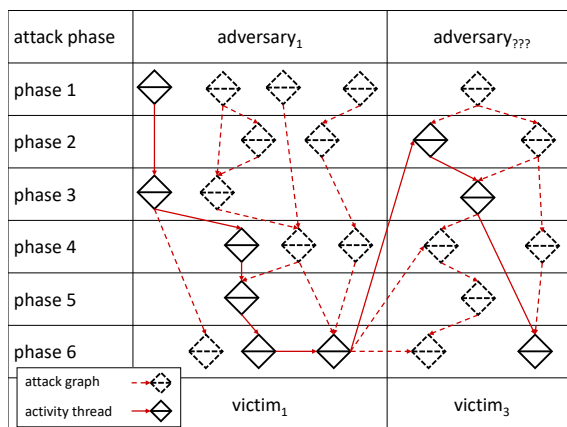
chain phases for intrusion [123]. In Diamond, the basic building blocks are attack events (represented by nodes), which consists of four core features, which are connected in a diamond shape (hence the name), and some meta-features [57]. Figure 3.6a shows an overview of the diamond features. The four core features of a Diamond attack event are:

- *Adversary* describes the set of adversaries that seek to compromise computer systems or networks to further and satisfy their intent and needs.
- *Capability* describes the tools and techniques an adversary used in the event. It consists of the capability capacity, which are the possible vulnerabilities that could be used, and the capability arsenal, which is the actual set of the adversary’s capabilities, resources, and knowledge.
- *Infrastructure* describes the physical and logical communication structures used by the adversary to deliver a capability, maintain control of capabilities, and effective results from the victim. It should be considered that in multi-stage attacks, a victim asset can be the end target in one event and then leveraged as the infrastructure in other events. Thus, one must always beware that the target of an activity may not necessarily be the victim.
- *Victim* describes the adversary’s target and against whom vulnerabilities and exposures are exploited.

The diamond model supports arbitrary graph structures and attributes. It is extensible and already includes confidence in the respective judgments. The last part is especially important since each of our judgments comes from historical data or expert judgment and, therefore, is subject to uncertainty. In accordance to the FAIR method [66] and the book series “How to measure anything” by Douglas Hubbard [124, 13], we use the confidence to express the respective uncertainty in the judgements. In RISKEE, we define the features in the form of value ranges and their respective confidences to establish probability distributions.



(a) Illustration of the diamond model of intrusion analysis (adapted from [57]).



(b) Illustration of activity threats and attack graphs (adapted from [57]).

Figure 3.6: Illustration of the diamond modelling capabilities. Figures reused from the included publication [P4].

Common Criteria (CC) is defined in the ISO/IEC 15408 [125] and is used for the certification of security in computer systems. The newest version is 3.1 revision 5 (April 2017). Common Criteria works by testing the devices against specific requirements defined in the common criteria catalogue consisting of several documents that are continuously expanded on demand. In January 2021, it consisted of 227

general protection profiles, 1548 specific security targets. The result of a risk assessment using Common Criteria is the certification with a specific **Evaluation Assurance Level (EAL)**. The EAL is a numerical rating, going from 1 (most basic) to 7 (most secure) and describes how well a component fulfils the **Security Assurance Requirements (SARs)**. SARs are a publicly available catalogue of requirements that a component must fulfil to get a specific EAL level. They define measures that assure the security properties for a specific **Target Of Evaluation (TOE)**. The description of a TOE consists of three parts:

1. **Protection Profiles (PP)** define general security requirements for whole classes of devices (e.g., smart cards)
2. **Security Targets (ST)**, refine these requirements for a more concrete type of product. These requirements are the security goals and properties which a product should fulfil. They could comply with multiple PPs.
3. Finally, the **Security Functional Requirements (SFRs)** define specific functions to ensure the goals of the security target. A product has to provide these functions to get certified for the respective EAL.

To get certified for a specific EAL level, components are tested by independent certification labs, whether they correctly provide all the functionality defined in the *functional security requirements* to assure the respective *security targets*. The EAL certification gives us the following guarantees: A system is guaranteed to be secure regarding the security assurance requirements defined in the EAL document for a specific target of evaluation (security target and protection profiles).

CVSS The Common Vulnerability Scoring System [80] is a standard to assess and prioritize software vulnerabilities. The newest version is 3.1 from 2019. It assesses vulnerabilities on a scale from 0 to 10 and is based on three scores consisting of several metrics: The mandatory base score is the foundation and can be refined by the temporal score and the environmental score, which are optional. Each score consists of several metrics, e.g., the base score consists of the metrics: Attack Vector, Attack Complexity, Required Privileges, User Interaction, Confidentiality, Integrity, Availability, and Scope. Each metric has explicitly defined values on an ordinal scale, e.g., confidentiality has the values None (0.0), Partial (0.275), and Complete (0.66); Attack Complexity has the values Low (0.77) and High (0.44). The standard defines precise formulas for how all these metrics and scores are combined mathematically [80].

FAIR The Factor Analysis of Information Risk (FAIR) is a method and ontology for quantified risk analysis and was one of the most influential methods for this thesis. The goal of FAIR is to define a unified terminology on how to speak about risks and their factors and provide a mathematical framework to calculate risk quantitatively by judging the contributing factors (which is easier and more accurate) [126, 38]. The Open Group and RiskLens standardize FAIR under the lead of Jack Freund and Jack Jones [37]. It works by splitting risk into several sub-factors which can be assessed more easily to evaluate IT-security and operational risk [37, 66, 67]. The whole ontology and factor hierarchy is shown in Figure 3.7. On the top level, FAIR splits risk into two factors: Loss Magnitude (LM) and Loss Event Frequency (LEF). Loss Magnitude consists of Primary Loss and Secondary Risk. Loss Event Frequency consists of the Threat Event Frequency (TEF) and the Vulnerability (Vuln). These individual factors are assessed with a three-point estimate (min, mode, max) and modelled as PERT distribution which originates in project management and was first used by the US Navy to estimate time plans for missions [36]. More specifically, the modified PERT-beta distribution [39, 68] is used, which includes the confidence as optional fourth parameter. For calculating the risk, Monte-Carlo simulation [127] is used. The result is a probability distribution over the magnitude of loss that may occur. By analysing

this distribution, it is possible to understand the risks on a much higher level of comprehension than just having a single point estimate or a coarse categorization like a risk matrix. The result is called the Loss Exceedance Curve (LEC), which depicts the outcomes and respective probabilities. Such a LEC can be compared to the risk appetite curve to decide if the risk is tolerable or not. To show its general applicability to risk management processes, the FAIR ISO/IEC 27005 Cookbook [38], describes how the FAIR process is conforming to the risk management process defined by ISO/IEC 27005:2018 [128] and ISO/IEC 27001:2013 [129].



Figure 3.7: The FAIR ontology showing the sub-factor decomposition [126, 37]. Graphic by FAIR institute [130].

Threat Event Frequency (TEF) defines the frequency of events that target the analysed system. It can be calculated as the product of the *Contact frequency CF*, stating how often the event occurs in general, and the *Probability of Action PoA*, defining the portion of how often the analysed system is the target of such an event.

Loss Magnitude can be broken down into additive subterms. The total loss magnitude of an event consists of two terms that have to be added up: *Primary Loss* and *Secondary Risk*. Primary losses result directly from the cyber-security incident, e.g., immediate response actions, replacement costs, and production losses. Secondary risk occurs in the aftermath as a subsequent event to the original incident. It has its own frequency, e.g., when stolen data is used for bribery when the company has to pay fines due to lawsuits or pay other financial penalties. Secondary risk events depend on primary risk events since they only become active when a primary event has occurred. The loss magnitude represents the impact of a successful attack, which is measured in financial terms. The financial magnitude is the loss of money that is caused by the attack. This can be due to direct loss (e.g., property damage, service outage, debits from bank accounts), called the primary magnitude, or indirect loss (e.g., reputation damage,

customer withdrawal, defamation, recovery costs) called the secondary magnitude. Harm is the other type of impact an attack can have. It revolves around harm and danger to human life and is considered separately. Financial magnitude could be estimated like: The loss is between 1000\$ and 2000\$. The FAIR ontology describes several types of impact types, which represent different sources for possible loss or damage:

- *Productivity Loss* is the loss due to the disruption of production and supply chain. This concerns the absence of value that could have been produced during the outage time of an incident.
- *Response Loss* are the costs produced as a direct answer to an incident. These are actions that have to be done to overcome and reflect on the incident.
- *Replacement Loss* are the costs of replacing damaged assets, resources, or personal compensation.
- *Fines and Judgments* are the costs caused by trials, courts, and penalties due to an incident.
- *Competitive Advantage* are losses that occur indirectly by being blocked with an incident, while competitors continue to research, develop, and advertise to increase their market share and acquire more customers.
- *Reputation Damage* is the loss of trustworthiness and brand equity due to an incident. Brand and company value could decrease, and the stock market could react badly to this.

3.8 Other Related Scientific Work

3.8.1 Uncertainty in Probabilistic Programming

Probabilistic programming, and probabilistic computing in general concerns tools, models, and devices which are capable of working with uncertain values [131, 132], fuzzy logic [133, 134], or heuristic computation and compilation [135, 136]. In probabilistic programming, every measured value can be represented as a random variable. Such random variables have continuous or discrete probability distributions used for inference, arithmetic, and conditionals in a program. Besides measuring and specifying the uncertainty, propagation, and evaluation play a significant part in probabilistic programming. Some use-cases for the applications of probabilistic programming and uncertainty propagation were described in the included publication [P2]. Andy Gordon wrote a survey about the current state of probabilistic programming [137]. James Bornholt et al. published several papers about their implementation approach of such mechanisms [138, 139]. Another aspect of this is approximate computing, which creates software with just enough precision as needed. Eva Darulova et al. investigated many aspects of approximate computing and created a framework for compiling programs with uncertainties to be faster and use fewer memory [135, 136, 140, 141].

We derived software design patterns based on the existing implementations in probabilistic programming: *PROBABILISTIC MODEL*, *PROPAGATION RULES*, and *INFERENCE TECHNIQUE*. These design patterns were used in *RISKEE* to model and propagate uncertain values and are explained in great detail in the included publication [P3]. Design patterns are abstract descriptions of proven solutions, including the benefits, drawbacks, and liabilities, which can be used as templates for solving a specific problem. Especially for the design and implementation of an application, design patterns are highly beneficial since they represent already proven solutions, which incorporate the experience of multiple professionals to create more flexible and maintainable solutions.

3.8.2 Metrology and Measurement Theory

Uncertainty in metrology consists of epistemic uncertainty (also called incertitude or systematic uncertainty) and aleatoric uncertainty (variability or statistical uncertainty). The epistemic component is

the uncertainty in the correctness of the model. In contrast, the aleatoric component accounts for the randomness part inside the model. According to the ISO GUM Standard "*Error is an idealized concept and errors cannot be known exactly.*" [142]. Measuring reality is a task that always involves some measurement error, hence has some uncertainty. The ISO GUM (JCGM 100:2008) standard and all additions define how to express uncertainty in measurements [142, 143, 144, 145, 146]. Here two ways of error propagation are used: The first applies Gaussian error propagation (see Section 2.4), and the second uses Monte-Carlo simulation. In RISKEE, we *assume* our experts to be similar to sensors and expert judgment to correspond to a measurement. Hence, we tried to apply the methods from metrology to our systems. Unfortunately, Gaussian error propagation was not applicable since we had to use different probability distributions than the normal distribution. Therefore, Monte-Carlo simulation was the way to go, but with some considerations to get deterministic and reliable risk assessments. See Section 4.4 for more information about our sampling and computation methods.

Design of the RISKEE Method

*All experts are equal,
but some experts are more equal.*

adapted from Animal Farm
George Orwell, 1945

Summary: *In this chapter, the design of the RISKEE method is described. It introduces risk graphs, describes the structure and attributes of risk graphs, and elaborates on the propagation of risk attributes over such graphs. Finally, it discusses the needed process steps to apply the RISKEE method and assess the values using expert judgment.*

◇◇◇

RISKEE is a method for assessing the risk of cyber-security threats in distributed networked systems. Such systems can range from single components to whole distributed networked infrastructures spanning over multiple continents. To model the risk in arbitrary detail, we use so-called risk graphs. Risk graphs model the steps of an attack in a system. This is based on the network infrastructure but goes beyond that by including network topology, software, side-channels, and human interaction. Any type of influence and communication (including side channels and social factors) can be included in a risk graph. Section 4.2 and Section 4.2.1 discuss risk graphs in greater detail. Risk itself is the range of possible outcomes of cyber-security incidents. By outcome, we mean the possible damage of an incident. Some outcomes are more likely than others, and the prediction and estimation of these situations involve high uncertainty. This means that single-point estimates cannot represent risk appropriately, but a probability distribution, which includes and models the uncertainty of the estimations, can. Furthermore, risk is an intangible, abstract value, which makes it hard to estimate. To tackle this problem, we use specific cyber-security attributes that are easier to estimate and can be used to calculate the risk. These attributes are frequency, vulnerability, and impact. Frequency defines the frequency of attacks, vulnerability describes their success rate, and impact defines the damage of a successful attack. Section 4.2.2 describes the attributes in more detail, and Section 4.3 describes how these uncertain attributes are propagated over a risk graph.

Finally, the RISKEE method also describes how to use expert elicitation to assess those risk attributes. It consists of multiple rounds of individual expert judgment. The experts discuss the previous round results, exchange information, and reconcile with each other without having to entirely agree since it is always an individual judgment. The final result is the combination of the weighted expert judgments. The weights are based on the prediction performance assessed by an initial calibration test every expert has to go through.

4.1 The RISKEE Method

The RISKEE method is a method for assessing risks in cyber-security by asking several experts for estimation of threats and value of assets. It is based on the IDEA protocol by Abigail Colson [147, 148]. We used this protocol as a basis since it is already established in expert elicitation. It is based on theories and methods practised in the field for decades already [149, 150]. The IDEA protocol uses structured expert judgment and defines several steps on how to do expert elicitation. We refined these steps and incorporated RISKEE into this process. Figure 4.1 shows an overview of the steps, consisting of calibration, the first round of expert judgment, discussion, the second round of expert judgment, and calculation of the final results.

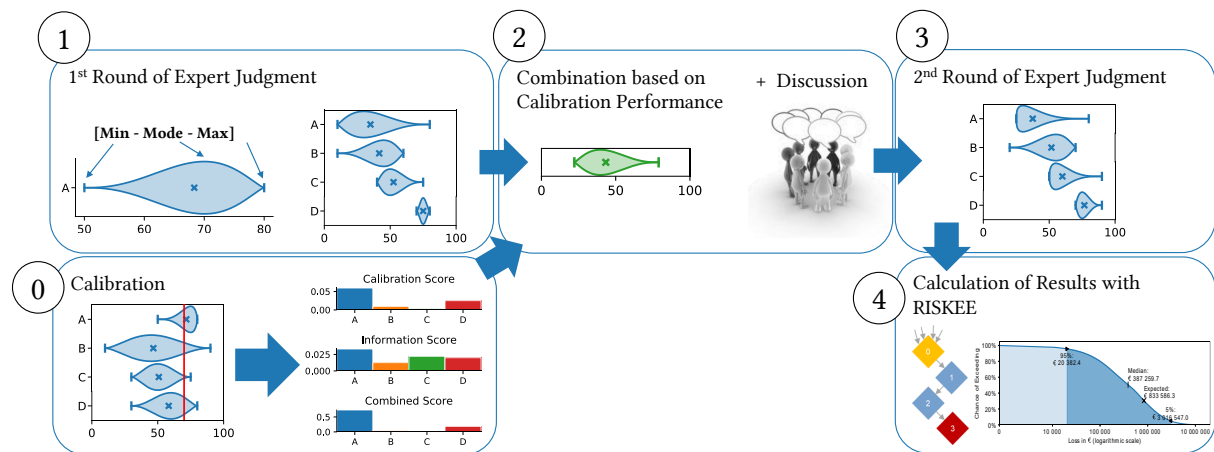


Figure 4.1: The RISKEE method for expert elicitation.

This process is described in more detail in the attached publications [P7] and [P8]. This process is an expert judgment that incorporates social consent, group dynamics, mathematic aggregation methods and considers individual judgment performance. Here is a short overview of the steps (as illustrated in Figure 4.1):

- 0. Calibration:** In this initial step, the experts are tested for their judgment quality by asking some calibration questions. This is needed to calculate the weights later on for the combination of real judgments.
- 1. First round of individual expert judgment:** In this step, each expert anonymously gives an initial judgment for the needed value to get an intermediate result that is not biased by any social or group dynamics.
- 2. Discussion and reflection on intermediate results:** In this step, the first results are discussed in the group, and during this discussion, knowledge exchange can happen, and experts can revise their judgment.
- 3. Second round of individual expert judgment:** Here again, the experts state their judgments anonymously.
- 4. Reflection on results:** Finally, the results are shown again, and the experts are asked if they want to accept them or continue with another iteration. If the result is accepted, the elicitation is over, and the results can be used for further analysis.

The details on each of the phases are described later in Section 4.5. Before we discuss how we get the data, we first describe what data we need and how risk graphs are structured.

4.2 Risk Graphs

Risk graphs are graphical models consisting of nodes and edges that have specific risk attributes. The nodes represent individual steps of an attack, and the edges represent the sequence of these steps. This notion comes from the well-known model of attack trees [62]. We refined this idea of an attack tree by distinguishing between different types of nodes (entry nodes, intermediate nodes, and impact nodes) and using specific risk attributes (frequency, vulnerability, and impact). The details about the structure and the attributes are explained in Sections 4.2.1 and 4.2.2. Figure 4.2 shows a visual example of an elementary risk graph.

The first design was intended to have only one impact node, similar to an attack tree with risk attributes. This is also the origin for the name *RISKEE*, as an acronym for risk tree. However, later, we enhanced the model to support multiple impact nodes and an arbitrary directed acyclic graph structure, making it more similar to Bayesian attack graphs and allowing for much higher expressiveness. The name *RISKEE* was kept, although now it uses risk graphs instead of risk trees. Aside from attack trees, several other ideas from the literature were woven into *RISKEE*. Specifically, the Diamond Model by Caltagirone et al. [57], the FAIR method by Freund and Jones [37]. Furthermore, the classical model and the IDEA protocol were used as the basis for expert judgment to assess the risk attributes. More about this is described in Section 4.5. An attack can be modelled without knowing all the details of its execution. However, the more detailed an attack is modelled, the more accurate assessments can be made. By keeping this abstract view of attacks, a risk assessment can already be done during system architecture and system design and does not directly depend upon a detailed and finished system description. This view was adapted from Caltagirone's Diamond Model [57]. As a tribute to this basis, we used diamond-shaped symbols to depict the nodes.

In the following sections, we describe the structure and the attributes of risk graphs. Furthermore, we explain how to create a risk graph and show and discuss a more extensive example in the end.

4.2.1 Structure of Risk Graphs

Risk graphs consist of nodes and edges. Each node represents a single attack (or attack step), and the edges between them represent the sequential order of these attacks along an attack path. These attack paths are the sequence of attacks that have to be done to reach an attack goal. The attack goals are the primordial reason why attacks are made, and they often inflict the most significant damage or impact. Attack goals are often stealing data, manipulating data, damaging a company's or person's reputation, bribery, or influencing systems' behaviour, e.g., by executing a denial of service attack or inflicting some malfunction.

Nodes in a risk graph can be distinguished into three different types, based on the semantic role along the attack path: entry nodes, intermediate nodes, and impact nodes. Figure 4.3 shows a simple risk graph (consisting of a single attack path). This form of depiction will be used throughout the remaining thesis.

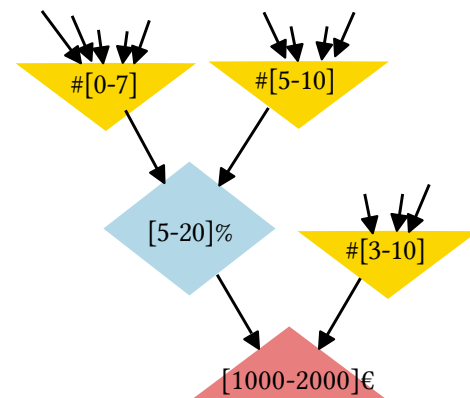


Figure 4.2: This risk graph shows three different attack paths which attack the same impact node.

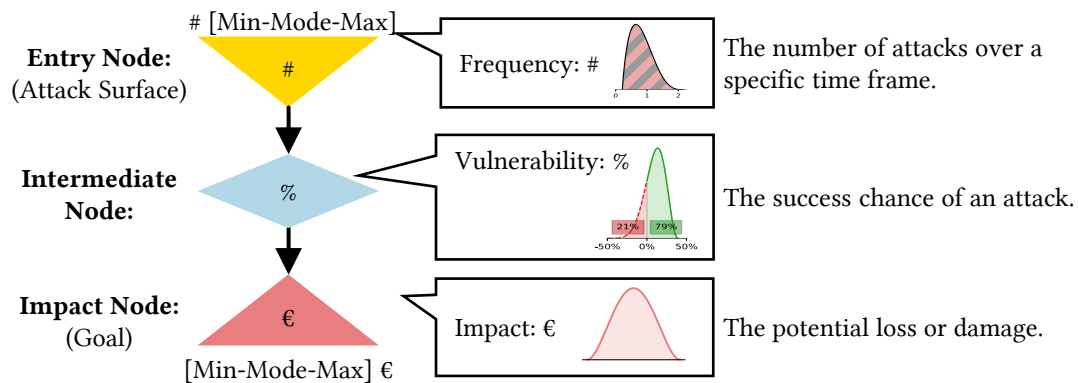


Figure 4.3: The structure of a single attack path, having an entry node, an intermediate node, and an impact node. Nodes are depicted as diamond-shaped figures, and edges are directed arrows connecting them. The frequency is generally denoted using the # character, vulnerability, or probability is using %, and impact is connotated with either the € or \$ character. Ranges are stated using [Min-Mode-Max] notation.

Types of Nodes in a Risk Graph

Nodes in a risk graph fulfil three semantic roles: Entry nodes are the nodes on the attack surface, representing the first step needed to enter the system. Impact nodes are nodes at the end of the attack paths, representing the goal of an attacker and inflicting the actual impact in the form of damage or losses. All other nodes in between are the intermediate nodes, representing single steps of the attacks. Each type of node has different attributes, which are only shortly mentioned here in the descriptions of the types but are described in more detail in Section 4.2.2.

Entry Nodes define the attack surface and represent the first step an attacker has to do to enter the system. Therefore, Entry Nodes are specified using the frequency attribute, which propagates throughout the attack path. It defines the risk attribute of **frequency**: The rate of attacks over a specific time frame (mostly a year). This only accounts for the attacks which are directed towards the system. All other attacks do not have to be considered. In RISKEE, Entry Nodes are depicted as triangles pointed towards the system as shown in Figure 4.4.



Figure 4.4: An entry node in RISKEE specifying the frequency.

Intermediate Nodes represent single steps that attackers have to undertake successfully to reach their respective attack goals. They only define vulnerability, which is the chance that the attack will be successful. If damage is done during the attempt, an impact node must be used right after the intermediate node. The path defined by the intermediate nodes ultimately leads to impact nodes. It defines the attribute **vulnerability**: How likely is the success of an attack attempt? Intermediate nodes are depicted as diamonds, as shown in Figure 4.5.

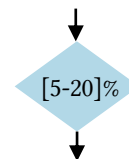


Figure 4.5: An intermediate node in RISKEE specifying the vulnerability.

Impact Nodes are the reason the attack was made in the first place – they represent the attack goals. Their attribute is the potential impact (either in monetary terms or other forms of damage). It defines the **impact** attribute

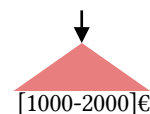


Figure 4.6: An impact node in RISKEE specifying the impact.

resembling the damage and losses due to the cyber-security incident, e.g., damage to hardware, production outage, data leaks, bribery, fines and judges, reputation losses. This step induces the most damage to the system, which was the original motivation for the attacker. Impact nodes are depicted in RISKEE as triangles pointed towards the system, opposing the entry nodes, as shown in Figure 4.6.

Structural Considerations

Here, we discuss different structures of risk graphs. This ranges from simple independent attack paths to a complicated graph consisting of many nodes and connections in-between.

Simple Risk Graph with Non-Intersecting Paths The simplest forms of risk graphs only consist of direct paths between an entry node and an impact node. When used in this way, the method corresponds to traditional risk event models, where every event is independent of each other and is analysed separately. This also depicts how such flat models can be converted to risk graphs: by creating three nodes for each incident and assigning the frequency to the entry node, the probability to the intermediate node, and the impact to the impact node. As the next refinement, this intermediate node could be split up into more nodes. Figure 4.7b shows how such a simple risk graph looks, and Figure 4.7a shows an even easier graph, which consists of an entry and impact node. Here the vulnerability is 100%, which is why the intermediate node can be left out. While such a simple risk graph is straightforward to construct, the benefits come with multiple nodes and more complex models with intersecting paths.

Advanced Risk Graphs with Intersecting Paths Figure 4.7c shows a risk graph with multiple connected nodes and intersecting paths. Such graphs already allow more sophisticated modelling of attack scenarios by splitting attacks up into multiple steps: Attacks that only concern the attack surface, attacks that are just intermediate steps, and attacks that directly target the ultimate attack goal. This decomposition makes it easier to judge the frequency, vulnerability, and impact for each node. Figure 4.7c also shows merging and branching paths:

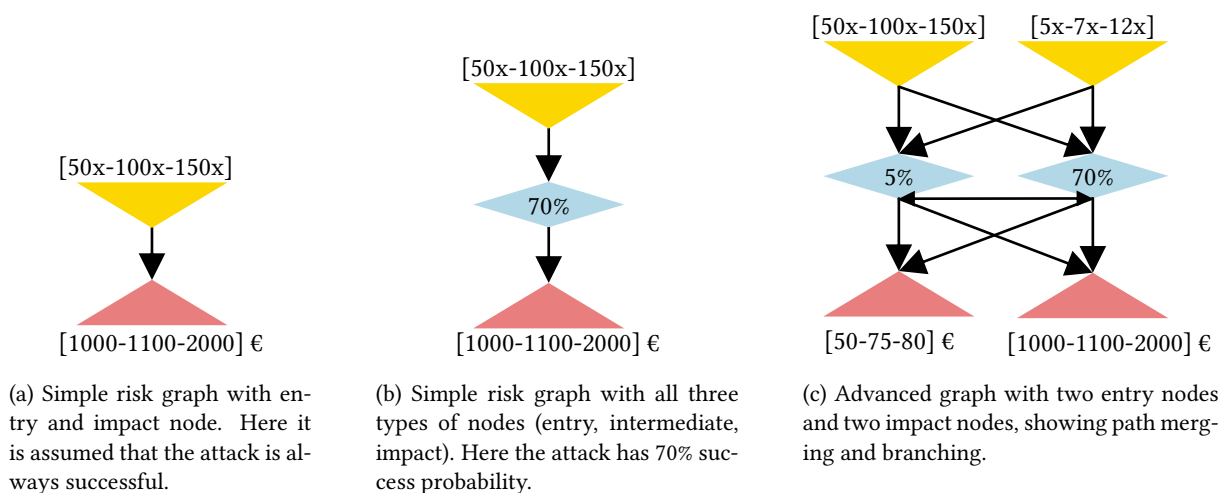


Figure 4.7: On the left: Two examples for simple risk graphs; On the right: An example for a risk graph with intersecting paths.

Merging Paths Paths that merge in a node combine the incoming frequencies. The total amount of incoming attacks stays the same. It does not matter how an attacker reached a specific node but that it was reached. However, the vulnerability could differ for each incoming path due to differences in knowledge or resources. This can be modelled by splitting up the paths and adding a duplicate node for the same attack but with a different vulnerability. Applying this transformation makes it possible to convert classical Bayesian attack graphs to risk graphs in linear time, as it was successfully demonstrated in the included publications [P10] and [P11].

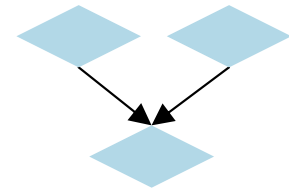


Figure 4.8: A risk graph where two attack paths merge into one.

Branching Paths When paths are branching, the frequency attribute is split up between the outgoing edges of a node. Without further attribution, this is done by equally distributing the frequency among the edges (Figure 4.9a), but this could also be specified in more detail by assigning a probability to the outgoing edges (Figure 4.9b). Outgoing attack paths that are more or less likely can be weighted by this means. In the investigated risk graphs, we always used equal weights for all outgoing edges. However, it would be a simple change to use specific weights for the outgoing edges to model the likelihood of an attacker taking a particular path. The only requirement is that the weights must sum up or be normalized to represent 100%. In total, the incoming frequencies have to equal the outgoing frequencies reduced by the vulnerability factor.

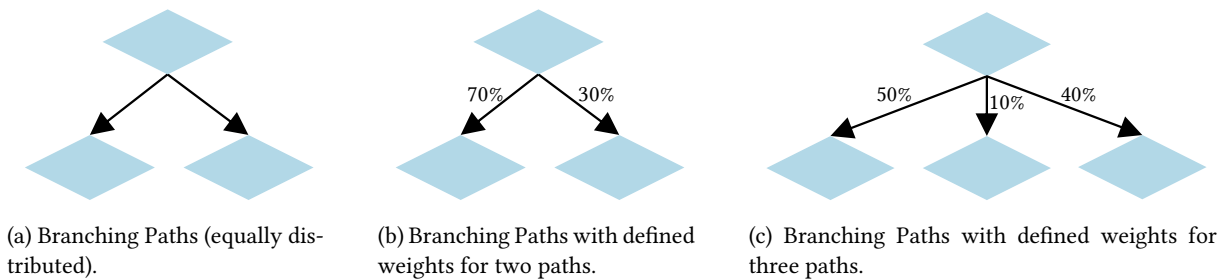


Figure 4.9: Figures of risk graphs with branching paths.

4.2.2 Attributes of Risk Graphs

The nodes in a risk graph have different attributes, which are used to calculate the risk according to the following simplified formula:

$$Risk = Frequency \cdot Vulnerability \cdot Impact \tag{4.1}$$

Each of the attributes in Equation 4.1 is described in the following paragraphs. This equation should only depict the basic idea – the actually used formula (Equation 4.2) is more complicated since the values are distributed over the nodes in an attack graph, and also the calculation is complex since we have to deal with distribution arithmetic (see Section 4.4).

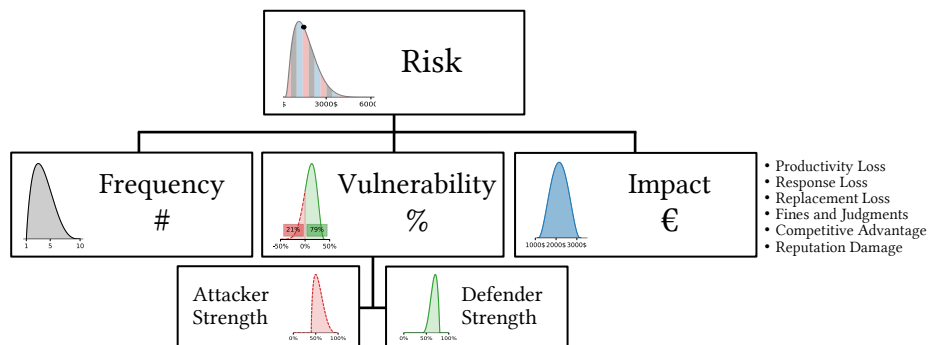


Figure 4.10: The adapted risk model with its sub-factors and illustrations for probability distributions. This is derived and adapted from the FAIR ontology [67].

Notation

Attributes in the risk graph are stated using the following notations.

Three-Point-Estimate: This notation allows stating the interval of values and defines the most likely value (the mode). In this thesis, we use square brackets to enclose the values: [min-mode-max]. Examples:

- [5–45–50] specifies a value range starting from 5 until 50, but most likely having a value of 45.
- [0%–50%–99%] specifies a range of percentages starting at 0, having most likely the value 50, and going up until 99%.
- [1000–2000–10000] € specifies an interval going from € 1000 to € 10000, with the most likely being € 2000.

In addition to the range also the confidence in the most likely value could be stated, which is used for parametrizing the PERT distribution. The confidence is stated in round braces after the range: [min-mode-max] (confidence). The values for the confidence are described in Section 2.3.4, and range from no confidence ($\lambda = 0$, uniform distribution), to very high confidence ($\lambda = 14$), and even higher if needed. The middle number could also be interpreted as the median or mean of the distribution, but in this thesis we assume it always to be the mode.

Interval-Estimate: The interval estimate specifies the limits of an interval, e.g., that an estimated value is between 20 and 80, but nothing more. The used notation in square brackets with two values, separated by a dash: [min–max]. Examples:

- [10–15] specifies a value range going from 10 to 15.
- [25%–75%] specifies the inter-quartile-range, going from 25% of values, to 75% of values.
- [1000–1500] € specifies an interval going from € 1000 to € 1500.

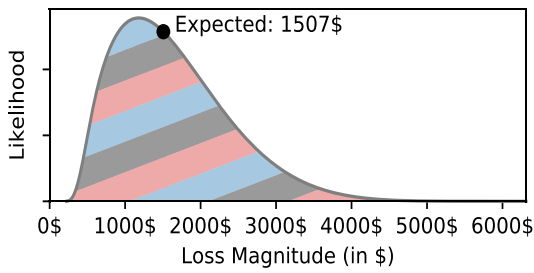
Here the most general assumption is that the values inside the range are distributed uniformly. However, other assumptions would also be reasonable based on the context, e.g., in the example shown in Section 4.3, we depict the estimated risk values as intervals, but assume a PERT distribution, having its mode in the middle of the interval. Another possibility would be to assume that the interval states the 90% confidence interval of a normal distribution.

Risk Attributes

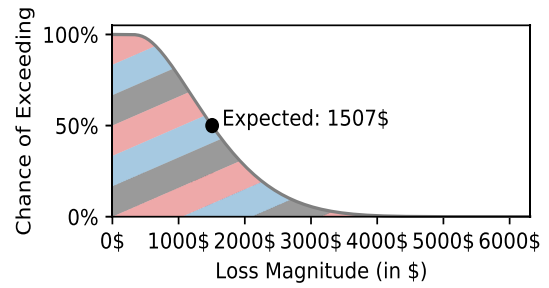
In the following paragraphs, the attributes in risk graphs are described.

Attribute: Risk The attribute risk is an abstract value that allows comparing the danger or harm of events. This is the result of the risk assessment, which combines all risk attributes. This value is a probability distribution covering all possible outcomes: Starting from situations where no or only minor damage happens to situations where everything goes terribly wrong. Usually, these extremes are rare, having the typical realization somewhere in between, but this depends upon the risk attributes. In very pessimistic estimations, the probability of the worst-case could be pretty high. In the end, one can decide if the risk is tolerable or not by looking at the risk distributions. Figure 4.11a shows an example of a probability density curve, and Figure 4.11b shows the respective loss exceedance curve (the equivalent to the survival function in statistical terms). These two depictions contain the same information, but the probability density curve encodes the risk in the area under the curve, while the loss exceedance curve directly shows it on the curve along the axes, which is easier to read. This already shows how important it is how risk is depicted and communicated. More about the visualization of risks is described in Section 5.2.2.

Risk is the range of possible outcomes based on the attack *frequencies*, the node *vulnerabilities*, and the *impact*. Equation 4.2 shows how risk is calculated in RISKEE.



(a) Example risk curve showing the probability distribution of risk.



(b) Example loss exceedance curve showing the chance of exceeding a certain amount of losses.

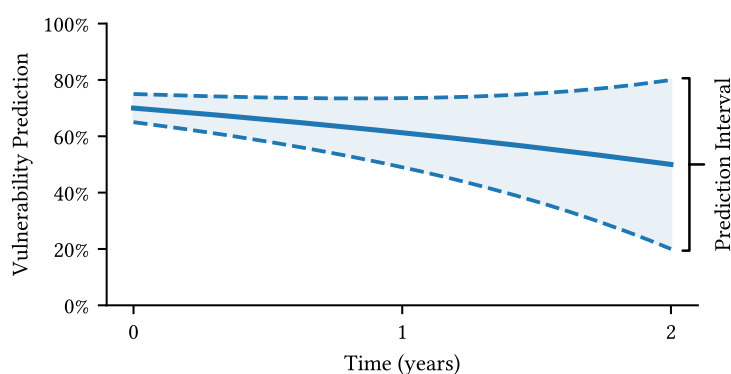
Figure 4.11: The same risk scenario is visualized twice: On the left with a risk distribution curve and on the right using a loss exceedance curve. While both show the same truth, the loss exceedance curve is easier interpretable for humans.

$$Risk = \sum_{\substack{P \in Paths(e,i) \\ e \in Entry\ Nodes \\ i \in Impact\ Nodes}} \left(Frequency_e \cdot \prod_{k \in P} Vulnerability_k \cdot Impact_i \right) \quad (4.2)$$

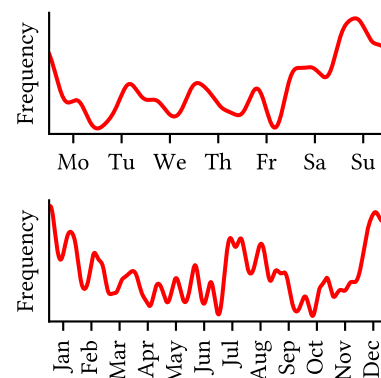
where: *Paths(x, y)* ... All paths between the nodes *x* and *y*.
Entry Nodes ... The set of all entry nodes.
Impact Nodes ... The set of all impact nodes.
Frequency_x ... The frequency attribute of node *x*.
Impact_x ... The impact attribute of node *x*.
Vulnerability_x ... The vulnerability attribute of node *x*.

Attribute: Frequency The *frequency* specifies the number of attacks over a specific time frame (most commonly estimated over a year). Other time frames would also be possible, but the estimations should all have the same frame; otherwise, extrapolations could introduce errors. Some attacks may occur more often in specific months of the year than in others. Moreover, uncertainty increases the longer the estimated period gets. This attack frequency is influenced by, e.g., the motivation of an attacker, the attractiveness of a target, or the accessibility of the system. Only Entry Nodes define the attack frequency. An important aspect is that the frequency of attacks is not constant but could vary over time. Two modelling aspects have to be considered:

1. The further a prediction goes into the future, the higher the uncertainty is (Figure 4.12a). There is no rule of thumb by how much the uncertainty increases, but this must be kept in mind when modelling the frequency attribute. Furthermore, this is true for the frequency and the vulnerability, impact, and every other estimated prediction. It is the most obvious for the frequency since it is an explicit judgment of the number of attacks over time. Vulnerability and impact also suffer from this increase in uncertainty the further they are predicted into the future.
2. The attack frequency could be distributed unevenly (see Figure 4.12b). There could be times when the attack frequency is higher or lower than on average. Attack frequencies vary over different times scales like times of the day or seasons of the year, e.g., during Christmas holidays or a crisis like the Corona pandemic in 2020, as stated in the INTERPOL cybercrime analysis report (August 2020) [151]. The report found that especially phishing attacks increased during the pandemic.



(a) Increasing uncertainty for predictions over longer time frames. The uncertainty (prediction interval) gets larger the further a prediction is extrapolated into the future.



(b) Changing frequencies for specific time frames (e.g., on weekends, in the summer/winter months).

Figure 4.12: Some aspects of frequencies: Increasing uncertainty for future predictions and changing frequencies depending on specific time frames.

Attribute: Vulnerability The attribute *vulnerability* defines the probability that an attack is successful. This probability is specified for each step along the attack path and is different for each attacker. Since this is a very vague value and difficult to estimate, it can be split up into subfactors that may be easier to judge: the *attacker strength* and the *defender strength*. The *attacker strength* describes the

capabilities, knowledge, and resources of the attacker. The *defender strength* describes the complexity, difficulty, and countermeasures and mitigations for an attack. These sub-factors were directly derived from the threat capability and resistance strength in the FAIR method [67].

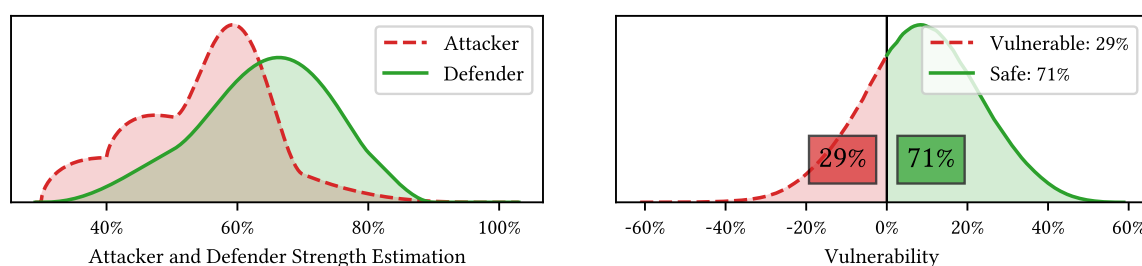
The *attacker strength* and the *defender strength* are always relating to each other and measured compared to the strongest possible attacker. If new vulnerabilities are found, the *attacker strength* increases, while the *defender strength* decreases relating to that. The *defender strength* increases only when structural changes in the system are done via updates, upgrades, or organizational changes. The *attacker strength* automatically grows over time since the attackers get continuously stronger by always keeping their capabilities and knowledge up-to-date. To keep up with the attackers, a system also has to actively and continuously evolve.

The range of estimation for both values always spans the whole support range (from 0% to 100%), and the expert can decide the most likely value and the confidence. The result of this is that the calculated vulnerability can never reach 0% or 100% due to the estimations' probabilistic properties. An attack is never guaranteed to succeed or fail in all cases. There is always a tiny remaining probability for the other case.

Here the two sub-factors for the vulnerability are described in more detail:

- Defender strength is the rating for which portion of all attacks the analysed system is protected. For example, if we have a highly resistant system, it may be protected against everyone except for the highest skilled attackers; therefore, its evaluation would be somewhere in the 90% range. In comparison to that, a very weakly protected system may have the resistance to protect against the most basic attacks and, therefore, could resist, e.g., just 30% of them. A rating of 0% for the resistance does not mean that every attack will automatically be successful. Moreover, a rating of 100% does not strictly mean that a system is protected against every attack – just that it has a very high protection level. There is still a tiny chance that attackers may come through in unknown or currently unimaginable ways. Side-channel attacks are an example of this. The mathematical model for this is a PERT distribution with minimum 0%, maximum 100%, and a mode of the respective rating. Additionally, a certainty could be given as the fourth parameter for the PERT-distribution: The certainty defines the distribution's spread and depends upon the stated judgment's precision.
- Attacker Strength is the rating of the capabilities, skills, and resources of an attacker, compared to the whole population of attackers. The most potent attackers with the highest skills and most resources qualify for 100%. This still does not mean that every attack will be successful – just a very high portion of it. The weakest attacker would have a rating of 0%, but still, there is a tiny possibility for successful attacks. Note that the attackers are steadily changing and getting stronger automatically over time. According to Oracle's chairman Larry Ellison [152] they are ahead of state of the art, and the defenders are constantly just trying to catch up. A good overview for the attacker groups, or cyber threat actors (CTA), as they are often called in literature, is given in [153, 154, 155, 156].

Figure 4.13a shows two example judgments for the attacker and defender strength. These came from three different experts. The experts agreed on the defender strength but not on the attacker strength, which is why the “humps” in the curve of the attacker strength. This shows the strength of the kernel density estimation since such a shape would not be possible to model using only a simple probability distribution. Subtracting the distributions from each other results in a distribution that spans over negative and positive values. The area on the negative side of values represents all cases where the attack was successful. This portion on the negative side of the graph is the resulting vulnerability, which can be used in the risk calculation. Figure 4.13b illustrates these results.



(a) The rating of the attacker strength and the defender strength via the combination of multiple expert judgments.

(b) The resulting vulnerability: values above zero depict cases where the attack is prevented, below zero means that the attack succeeded.

Figure 4.13: Vulnerability calculation via the subtraction of defender strength and attacker strength.

$$Vulnerability = \frac{1}{2} \left(1 - \frac{Defender - Attacker}{\|Defender - Attacker\|} \right) \quad (4.3)$$

where: *Defender* . . . Defender Strength: An estimation of the nodes defensive strength on the scale of all possible attackers.

attacker . . . Attacker Strength: The range of plausible attacker strengths for the analysed attack on a node on the scale of all possible attackers.

Attribute: Impact The *impact* defines a comparable value of damages or losses. Most commonly, this value is defined in monetary terms. It could also use any other ratio scales, as long as it supports linear algebraic operations like multiplication or addition. This means that ordinal scales and semi-quantitative scales are not usable here. Furthermore, the impact is hard to estimate as a single value because it often consists of many additive subterms, e.g., response costs, replacement costs, production outage, fines and judgments, and even damage to reputation and loss of future customers. Therefore, estimating it as a single value could lead to inaccuracies. The individual terms should be estimated and summed up to get the final impact. We refer to the loss magnitudes defined in the FAIR method [67]. They defined primary and secondary losses, which can be further split into several types of loss and damage.

4.2.3 Creating Risk Graphs

Here we describe how to create risk graphs. There are two ways to create them. Ideally, both should be considered to get a complete picture: risk modelling and threat modelling. Risk modelling begins with defining the valuable assets and identifies and evaluates the possibilities of damage to them. Threat modelling also starts with the assets but then goes another route by looking at specific vulnerabilities and threats to exploit them. Risk modelling is more abstract and technology-independent, while threat modelling relies more on the applied technologies, exploits, and vulnerability databases. Furthermore, risk modelling is more about the assets and the defender side of attacks, while threat modelling is more about the attacker perspective. We elaborated on threat modelling and integrating quantifiable security risk in the included publication [P4].

Before the risk of a system can be assessed, it is essential to define the valuable assets and how they could be attacked. This can be done using a risk graph. Below, the concrete steps to create a risk graph are explained:

1. **Define the valuable assets.** The first step in risk analysis is to specify the valuable assets. Valuable means that it would inflict damage or losses to the organization or individuals if these assets were damaged, stolen, lost, or modified. Assets include physical objects but also immaterial things like data, information, or intellectual property. The act of compromising these assets is the attacker's goal, and the damage is represented by impact nodes in the risk graph. Therefore, with defining the assets also their possible impact of damage and losses should be defined. The value of the impact can be specified independently of the possible attacks.
2. **Define the Attack Surface.** The attack surface represents the borders of the security domains in our system. These are the entry doors and first barriers an attacker has to overcome to enter the system. This also includes attacks that do not need Internet connectivity, e.g., a compromised USB stick dropped at the car park could be brought into the company by employees, and unwittingly a virus could be injected by just plugging it into the computer. Here, the attack surface is the publicly accessible car park. Generally, the attack surface is the border of influence between controllable and uncontrollable environments, situations or other aspects. This notion also includes the uncontrollability of personnel and humans, given that many attacks are initiated by insiders [6]. This is not a plea to not trust the employees anymore – it should just raise awareness that not all attacks come from outside attackers. The nodes on the attack surface define the frequency of attacks. Different types of attackers could be modelled by having multiple entry nodes for each attacker profile, having different frequencies.
3. **Find the Attack Paths.** In the next step, we model the *possible* ways of reaching these valuable assets from the entry nodes on the attack surface. These possible ways are the so-called attack paths and describe the steps to reach the impact nodes. The attack paths can be modelled without knowing the exact techniques or tools needed for the attack, but just the possibility that an attacker could take this path. Of course, knowing more details about an attack would help because the more one knows about an attack, the smaller is the uncertainty when judging the attacks' success probability – the so-called vulnerability. To create the attack path, one has to think of individual steps and judge the vulnerability of each of these steps. The vulnerability compares an assumed attacker's strength and the system resistance – it answers the questions: How vulnerable is the system against a specific type of attack and attacker capabilities? With the vulnerability judgment, assumptions about the attackers' strength are made. Different attacker profiles should be judged separately since attackers may have different attack frequencies and vulnerabilities based on their resources, knowledge, and intents.

4.3 The RISKEE Propagation Algorithm

After the risk graph is established and the attributes are assessed, the risk can be calculated. For this, we established the **RISKEE Propagation Algorithm** (see Algorithm 1). It works by multiplying the frequencies with the respective vulnerabilities on each attack paths until the path ends at an impact node. Here the remaining frequency is multiplied with the impact, which results in the respective risk emerging on this path. This risk is then propagated back to all nodes along the path by adding and accumulating it on the risk attribute for each node.

Algorithm 1 RISKEE PROPAGATION ALGORITHM v2. The algorithm for propagating risk attributes forward and backward over a risk graph. Refined version from the included paper [P5].

Input: G ▷ $G \dots$ Directed Acyclic Risk Graph
Input: $frequency_i$ ▷ Frequency for each entry node i .
Input: $vulnerability_i$ ▷ Vulnerability for each intermediate node i .
Input: $impact_i$ ▷ Impact for each impact node i .

- 1: **procedure** PROPAGATE(G) ▷ Initialize the risks.
- 2: $risk_{total} \leftarrow 0, [risk]_{nodes} \leftarrow 0, [risk]_{edges} \leftarrow 0$
- 3: **for all** $path \in Paths(G)$ **do**
- 4: $frequency \leftarrow frequency_{entry}$
- 5: $risk \leftarrow 0$
- 6: **for all** $node \in Nodes(path)$ **do** ▷ Propagate attacks forward on the path
- 7: $frequency \leftarrow frequency * vulnerability_{node}$
- 8: $risk \leftarrow risk + impact_{node} * frequency$ ▷ Compute the risk each path.
- 9: **end for**
- 10: $risk_{total} \leftarrow risk_{total} + risk$ ▷ Aggregate the total risk.
- 11: **for all** $edge \in Edges(path)$ **do** ▷ Propagate risk backwards to edges on the path
- 12: $risk_{edge} \leftarrow risk_{edge} + risk$
- 13: **end for**
- 14: **for all** $node \in Nodes(path)$ **do** ▷ Propagate risk backwards to nodes on the path.
- 15: $risk_{node} \leftarrow risk_{node} + risk$
- 16: **end for**
- 17: **end for**
- 18: **end procedure**

Output: $risk_{total}$ ▷ Total aggregated risk for the system.
Output: $[risk]_{nodes}$ ▷ List of risk values for all nodes.
Output: $[risk]_{edges}$ ▷ List of risk values for all edges.

In Figure 4.14 a simple risk graph is shown to illustrate the calculation principle of the RISKEE propagation algorithm. Each node represents a single step in a cyber-security attack. The risk graph has three entry nodes with attack frequencies as the min-max-intervals: 0-7, 5-10, and 3-10. The first two are connected to an intermediate node with has a vulnerability between 5% and 20%. The third is directly connected to the impact node (and with that, it implicitly has a 100% vulnerability). In the end, they all lead to an impact node inflicting damage between € 1000 and € 2000. All these values are modelled using a PERT distribution with default confidence. For simplicity, we assume that the most likely value is the mean of the stated ranges, so we did not explicitly state it in the figure. The RISKEE propagation algorithm splits the risk graph into its attack paths, calculates the risk for each path and aggregates all these risks together in the end. For illustration, let us look at a specific path, drawn in Figure 4.14. The path begins at the second entry node, goes to the intermediate node, and ends in the single impact node. Figure 4.15 shows this path in isolation with the estimated risk values as probability distributions. Furthermore, it

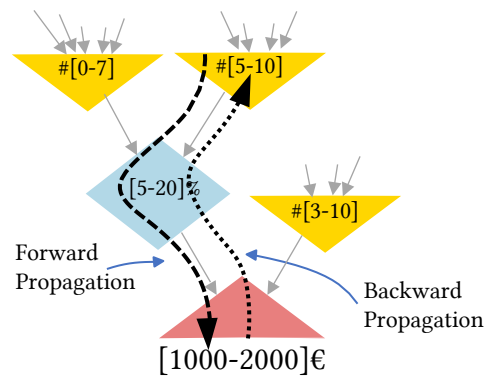


Figure 4.14: Example risk graph showing risk propagation.

shows the calculation of risk, which works as follows: The distribution of the attack frequency [5–10] is multiplied with the vulnerability of [5–20]%, resulting in [0.25–2] attacks per year. This is then multiplied with the impact of [1000–2000]€ which results in a risk distribution of [250–4000] €, emerging on this impact node. Mathematically the distribution is sufficient to calculate all interesting properties, e.g., statistical moments like the mean, median, or standard deviation – also quantiles can be calculated like IQR, the inter-quartile range spanning from 25%-75% of the values, or the 90%-interval going from 5%-quantile to the 95%-quantile. However, for humans, a probability density is tough to interpret. Therefore, a more intuitive view is the depiction of the loss exceedance curve drawn on the right of Figure 4.15.

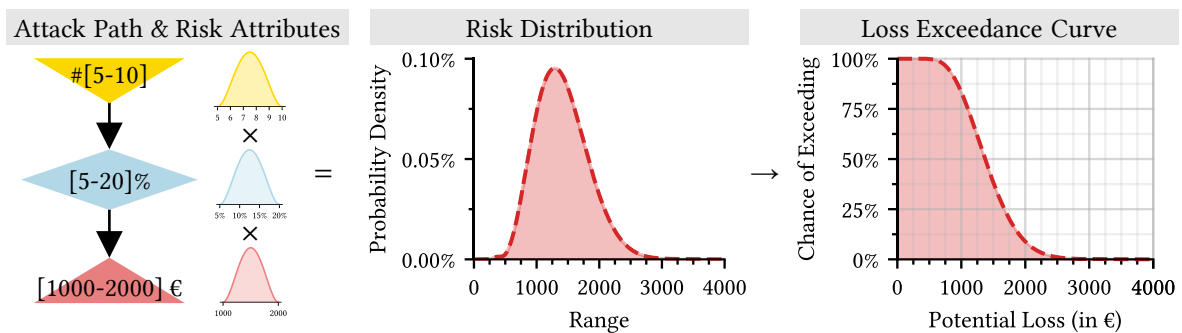


Figure 4.15: An example for calculating the risk for a single attack path (the second attack path in this case).

This emerging risk distribution is propagated back over the path where the attack came from. This works by adding it to the risk of each node along the attack path. In this case, those are the impact node, the intermediate node, and the second entry node where the attack path began. If a node already has an assigned risk value from a different path, the new risk is added to it. For example, the intermediate node is part of two risk paths, and the risk that emerged in this second risk path is added to the already calculated risk from the first path.

This is done for all possible paths in the risk graph illustrated in Figure 4.16. This figure shows an overview of the calculation steps: First, the risk graph is split up into individual attack paths. Then the risk for each attack paths is calculated as was described before. The resulting risk distribution is propagated back over the path and added up in the respective nodes, and for the total risk, all risk paths are summed up. In this example, the risk of the first attack path is [0–536–2800] €, the risk for the second path is [250–1289–4000] €, and for the third path, it is [3000–9305–20000] €. Here, we used the notation of the pert distribution: [min–mode–max]. While the numbers already gave it away that the third path inflicts the most risk, Figure 4.16 demonstrates the extent graphically: By looking and comparing the different loss exceedance curves, it can easily be recognized that the third path inflicts the most risk – even a magnitude higher than the other paths. Therefore, this would be the first spot for mitigation strategies, e.g., installing a firewall, using stronger encryption, or introducing a stricter authentication policy.

Remark: This illustrative example is very simplified, having only a few nodes and short attack paths – in reality, the model would consist of more attack paths having more nodes representing multiple steps of attacks. Figure 4.17 shows a more complex example of a risk graph. Furthermore, we used simplified estimations for the risk attributes. In reality, these estimations would come from structured expert judgment.

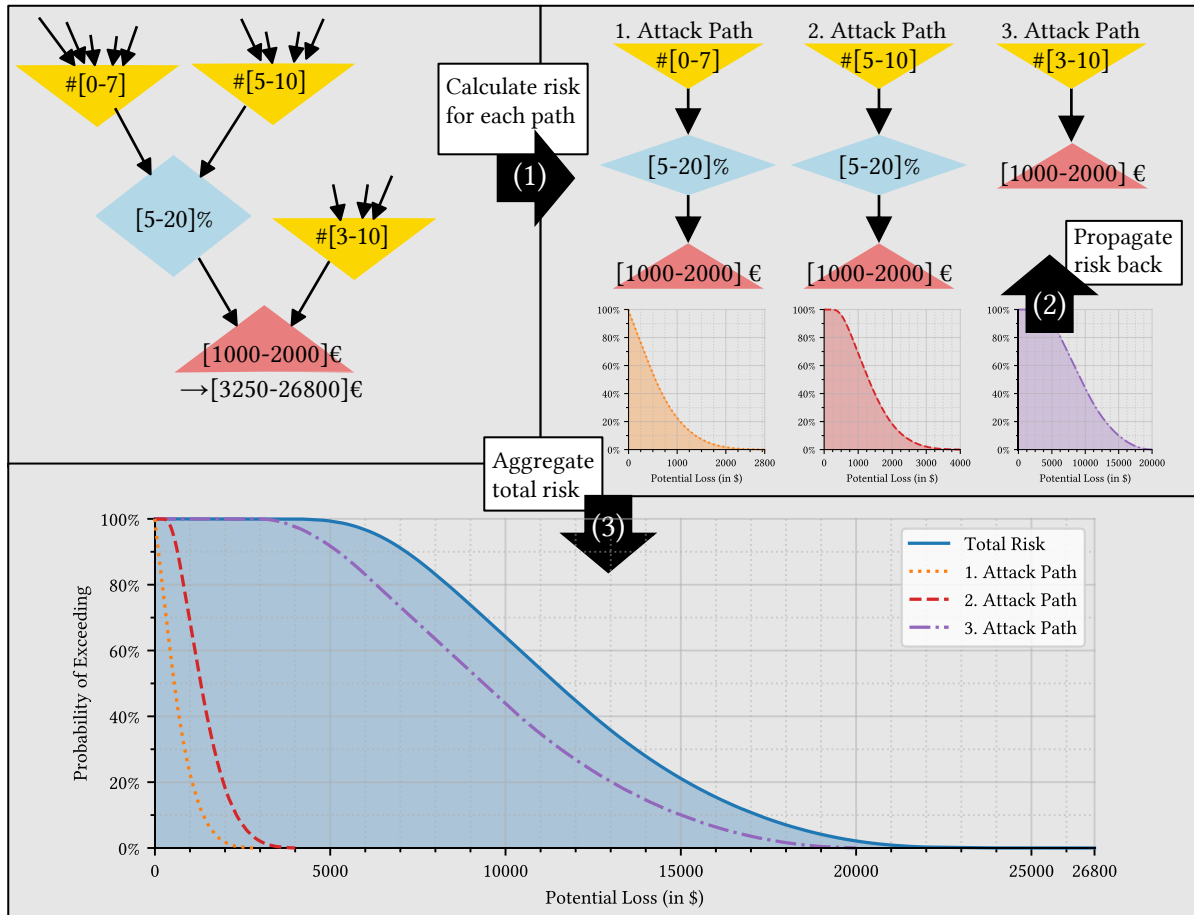


Figure 4.16: An example for calculating the risks over multiple attack paths in RISKEE.

4.3.1 Risk Graph Example

Here an illustrative example for a risk graph is shown and discussed (Figure 4.17). This example was derived from Caltagirone et al. [57] and brings together the structure and attributes of a risk graph. The numbers are not estimated by real judgments but are arbitrarily defined for demonstration purposes. This example is included here to show a more extensive example with more nodes and paths than previous examples.

Figure 4.17 shows a risk tree with 33 nodes. The yellow triangle-shaped nodes on top of the graph ((1), (2), (3), (4), (6), (7), and (8)) represent the entry nodes. They define the frequency of attacks. At the bottom of the graph, the nodes (31), (32), and (33) represent the impact nodes, being the ones that define actual damage values (impact). They are depicted as red triangles pointing upwards (towards the system and contrary to the entry nodes). All three impact nodes have 2000\$ as the impact value but result in different values for the actual risk since they are not equally reachable by the attacks. The different paths result in different risk results, being 151\$ for node (32), 340\$ for node (31), and 181\$ for node (33). All other nodes in between are diamond-shaped intermediate nodes which define the vulnerability (always 50% in this example).

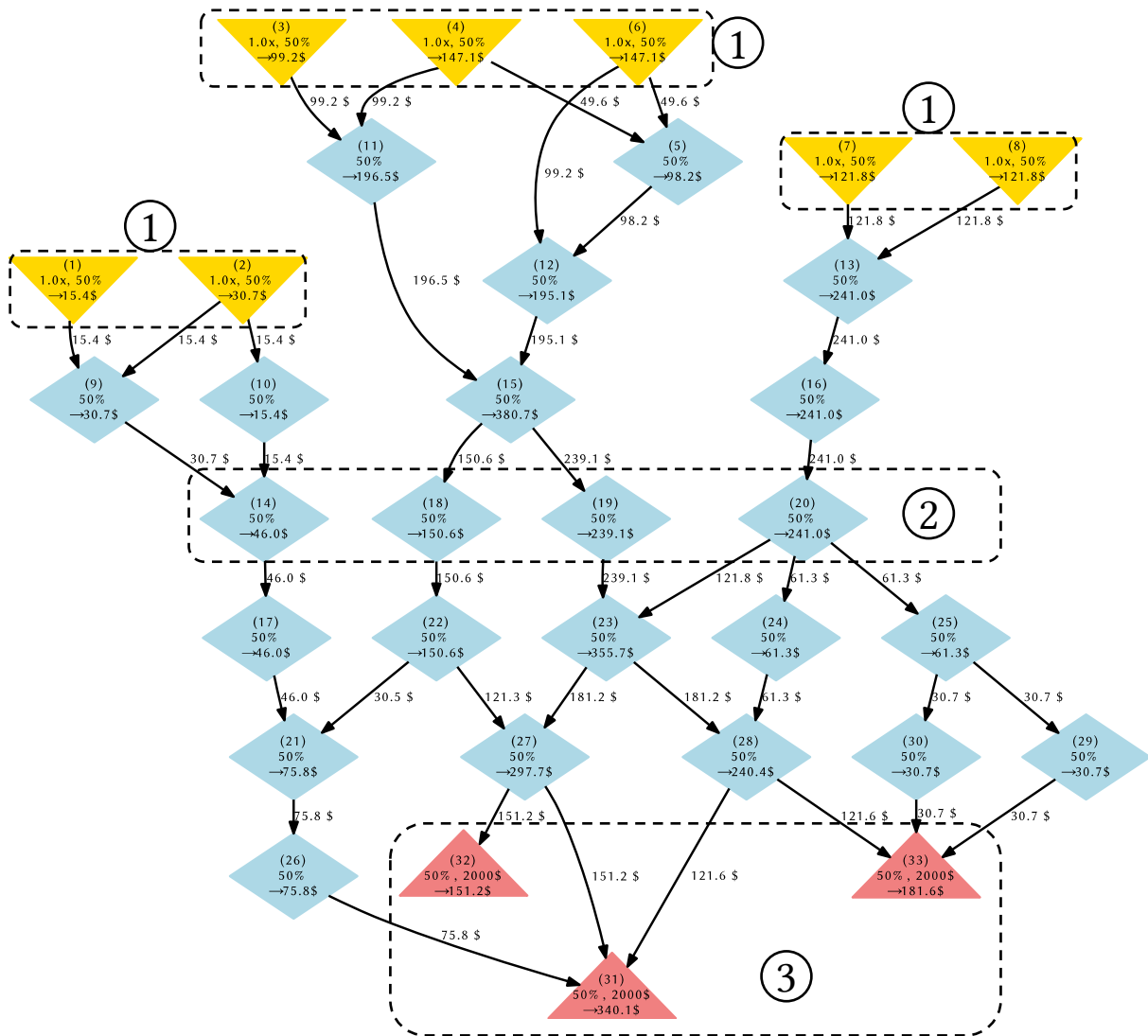


Figure 4.17: An example risk graph, containing 33 nodes in a directed acyclic graph structure. The entry nodes are the sources on top, the goal nodes are the sinks on the bottom, and the intermediate nodes are the connectors in between.

Interpretation of the Graph The example in Figure 4.17 shows some interesting features which can be discovered using a risk graph. We describe the annotations from the bottom up:

1. Annotation 3 in the figure shows that the impact nodes get different risk values assigned, although they have the same estimated impact value (of 2000 \$). This is because they are reachable by different paths with different attack frequencies and vulnerabilities; hence the RISKEE propagation algorithm calculated different resulting risks. For example, Node (31) has the highest risk evaluation of 340.1 \$, which is a good indicator that it should be treated first.
2. Annotation 2 marks some of the different risk values for the paths and intermediate nodes. This reflects the individual contribution to the total risks emerging in the impact nodes. Hence, the different risk values show the different influences of the nodes on the total risk. For example, Nodes (14), (18), (19), and (20) contribute 46 \$, 150.6 \$, 239.1 \$ and 241 \$ of average risk to the total.

Here, again, the paths and intermediate nodes with the highest risks would be good candidates for mitigation.

3. Annotation 1 shows that looking at the risk evaluation of the entry nodes also reveals an interesting insight: Which entry node is responsible for the highest risk contribution? Maybe this is also an excellent spot to look if it can be mitigated or if the frequency can be reduced somehow by, e.g., making it less attractive for an attacker.

4.4 Distribution Arithmetic

Here, we describe how arithmetic operations like multiplication and addition can be done with probability distributions. We call this *distribution arithmetic*. The used risk attributes, *frequency*, *vulnerability*, and *impact*, contain uncertainty. This uncertainty has to be modelled and considered when doing calculations like multiplication and addition. As described in Section 2.3, we can use probability distributions for modelling this uncertainty. For example, the normal distribution is used very commonly for measurements. However, the normal distribution is not suitable for risk estimations since it is symmetric around the mean, which risk estimations, for most cases, are not. They often have a long tail, meaning that the maximum value could be much larger than the minimum or the mode, spreading the distribution towards the larger side. For that reason, a log-normal distribution is often used. The log-normal distribution is asymmetric and allows the maximum value (or the 95% percentile in our case) to be magnitudes higher than the minimum or mode value. However, the reverse is not true: The log-normal distribution does not support so-called left-skewed or positive-skewed distributions. Such distributions have increasing density towards the larger side, while the log-normal only allows a higher density towards the lower portion of the distribution. Unfortunately, risk estimations could take arbitrary shapes and could even be multimodal.

Actual estimations for risk attributes are always finite. Therefore, it is not reasonable to model them as an unlimited probability distribution. This means that infinite distributions have to be trimmed at some point to correspond to an actual estimation. Of course, there are truncated versions of unlimited probability distributions, e.g., the truncated normal distribution. However, truncation can lead to erroneous distortions of the distribution shape if done without extra considerations like reflection. The alternative to this is to use finitely bounded distributions. Only a few continuous distributions have bounded support on both ends of the support domain: Uniform, Triangle, Beta (and derivations: Kumaraswamy, PERT, Arcsine, Wigner Semicircle), Logit-normal, Reciprocal, Bates, Metalog, Trapezoidal. RISKEE uses the PERT distribution to model the individual expert judgments because it is a unimodal, finitely bounded distribution that is very versatile (regarding the shape, which furthermore determines the probability density). It can be parameterized to model symmetric and asymmetric distribution, which can be left-skewed and right-skewed. Even the kurtosis can be adjusted to model the certainty in the mode value. More details about the PERT distribution were described in Section 2.3.4. Multiple expert judgments are combined using a weighted linear opinion pool. The resulting distribution is then approximated using a kernel density estimation truncated at the percentiles corresponding to the 6σ -intervals from the mean. This new distribution is used in the computations using stratified sampling over the trimmed support range.

4.4.1 Computation using Probability Distributions

Computing arithmetic operations like addition, subtraction, multiplication, and division with probability distributions require consideration of their inherent stochastic characteristics. There are two options for calculating with probability distributions: Either analytically or numerically. Analytical

solutions are possible when the distribution is always the same (for example, the Gaussian error propagation assumes normal distributions), but this is not applicable for arbitrary distributions. In such cases, numerical solutions solve this by using approximation, interval arithmetic, or sampling/fitting a new model. Approximations transform the distribution to a normal distribution and applies Gaussian error propagation. Of course, this only works when the transformation is reasonable because otherwise, it could induce huge approximation errors. With interval arithmetic, the underlying distribution is dismissed, and only the boundaries are used in the calculations. This is useful for quickly calculating the minimum and maximum limits but does not provide any information about the distribution of the values inside. The third option, sampling, is the one we chose. This is commonly called Monte-Carlo simulation and works as follows: Random samples of the input distributions are drawn, and these values are used in the calculations. If done often enough, the distribution of the results approximates the exact analytical solution. The challenge here is to find a sampling strategy that quickly converges to the analytical solution. The problems with inefficient sampling are that to get accurate results, the number of samples has to be very high (e.g., in the millions), and the border cases could be hard to overcome (multimodal distributions and areas where the probability density is very low). To cope with these problems, we refrained from using random sampling at all but used stratified sampling. Stratified sampling is a deterministic way of drawing samples based on a specific strategy, e.g., drawing 100 samples at every percentile of the distribution or calculation of the probability space between equidistant bins on the support domain of the distribution. In such a way, we get deterministic samples that consider multimodal distributions and low-density areas.

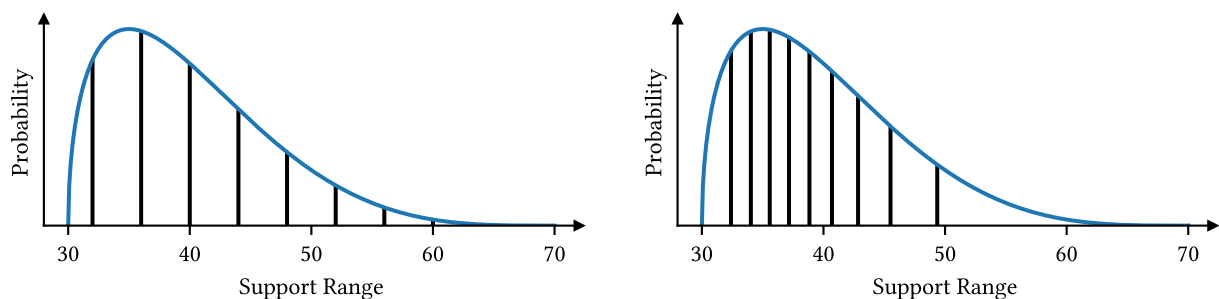
Calculations are done with all combinations for the drawn samples. For example, if we want to add to distributions $A + B$, we first have to draw samples from these distributions and then calculate the addition for every combination of samples. In the end, we approximate a new distribution based on all the results from the addition using kernel-density estimation.

Monte-Carlo Simulation is an approach that takes random samples out of the probability distribution and does the calculations with these samples. This works with arbitrary probability distributions and supports arbitrary arithmetic operations at the cost of precision and performance overhead. Taking too little or accidentally the wrong samples would lead to bad results, but taking very many samples results in expensive computationally simulations.

In early versions of RISKEE, we used randomized sampling instead of stratified sampling. However, due to the non-determinism, minor differences in the first samples are amplified repeatedly to result in entirely different outcomes. Furthermore, the extremes were not always considered since they only occur very rarely. Both effects are not acceptable for risk calculation since we want to have deterministic results, and we want to cover the black swan events. Taking many samples would probably solve this most of the time, but not consistently due to the random effects, and it would get expensive computationally. There are many well-known strategies to speed up the sampling process for random sampling, like Metropolis-Hastings Sampling [157] or Gibbs-Sampling [158]. These methods try to sample the highest density areas of distributions to accumulate the results with the highest probability quickly. The challenge, which makes this problem difficult is the arbitrary shape of distributions and that they could be multimodal. A good sampling algorithm has to cover the dense areas of distributions first but should also consider the areas with a low probability to cover the whole support range. To avoid these random effects altogether, we switched from nondeterministic sampling to stratified sampling using a fixed number of quantiles in the calculation and kernel-density estimation for smoothing the result to minimize digitization errors of the resulting histogram. This way, the results were deterministic, the black swan events were covered, the errors due to approximation were kept small, and the performance was acceptable. In stratified sampling, the samples are drawn out of a distribution by using

some specific strategy, e.g., by clustering the results and drawing a proportional amount of random samples out of each group based on the group size. Another strategy is systematic sampling, which takes every n th value as a sample, but this only works when the samples are ordered and equidistant over the whole sampling range (equiprobability property). Another rule, which we used in RISKEE, is quantile- or percentile-sampling. Here, we get samples at equidistant quantiles, e.g., every 1%. We need the inverse cumulative distribution function (or survival function) to look up the quantiles quickly for this to work. We chose this method because it is deterministic, and it also considers the low probability areas when the quantiles are fine-grained enough. However, for infinite distributions, we have the problem that the 0%-quantile and the 100%-quantile are at negative and positive infinity. We encountered such corner cases by not using the 0% and 100% percentile, but truncating the distribution at the 6σ -interval from the mean, which results in coverage of 99.999998% of the probability space. During development and trial sessions, we decreased this to a 3σ -interval, which still covers 99.7% of values and only uses 100 samples, reducing the range and calculation effort. However, this performance optimization may dismiss black-swan events, which is not acceptable. However, during exploration and experimentation, it was more important to have fast results than accurate ones. Higher resolution and broader distribution intervals can be used to get more accurate results in the end.

First, we used a fine-grained histogram with linear or b-spline interpolation between the bins. Later, we switched to a bounded kernel density estimation to get smoother and less error-prone results. The first interpolation approaches had a severe problem of overfitting and extreme extrapolations when the input distributions contained 0 as a value. Especially, b-spline interpolation additionally had the problem of producing *negative probabilities* that had to be truncated due to the cubic curve-fitting between nearly equal support points. This led to high distortions and erroneous behaviour in the distribution. KDE avoids such a problem by smoothing it with Gaussian kernels, which are added up and normalized to preserve the attributes of a probability distribution (more about that in section 2.3.5).



(a) Equidistant bins over the **support dimension**. They do not consider the probability density, but cover the supported range dimension equally. Low-density regions get the same amount of samples than high probable regions, but cover different probability spaces.

(b) Equivalent bins over the **probability density**, containing the same probability space in each bin, but only sparsely sample the support dimension on areas with low probability. In this example, the range 49–70 is only covered by one single bin due to its low probability.

Figure 4.18: Two examples of stratified sampling: The left is based on equidistant bins on the support; the right has equidistant bins based on the probability area.

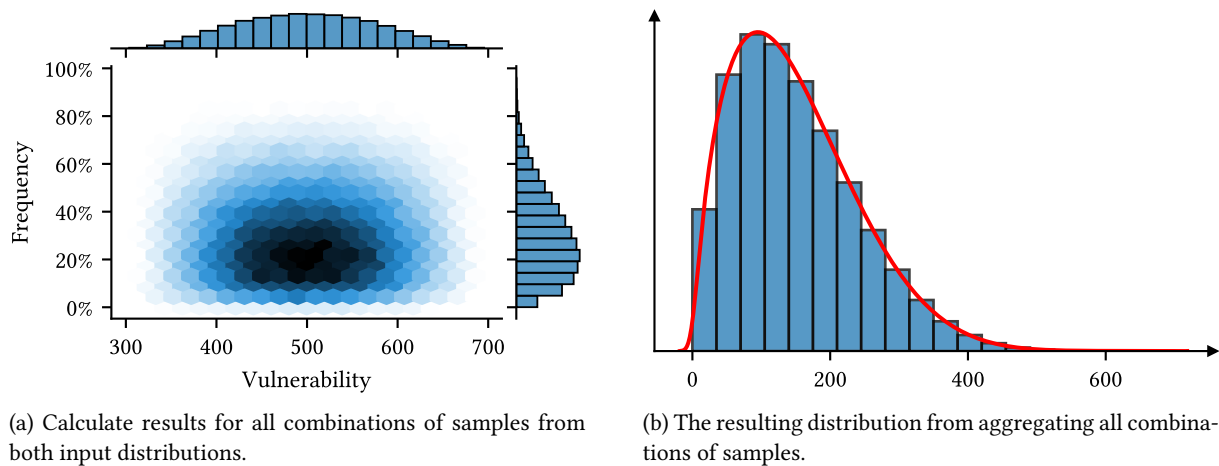


Figure 4.19: Simulation of arithmetic operations on probability distributions by sampling and calculating the results for each combination, which produces the resulting distribution.

4.4.2 Limiting Infinite Probability Distributions

In our model, the estimated values are often bounded by an upper and a lower limit, e.g., probabilities can only be between 0% and 100%. Financial loss, as well as frequency, cannot be negative. However, probability distributions often do not have upper or lower limits and go on until infinity. The normal distribution and the log-normal distribution are examples for this, as well as kernel-density estimation using Gaussian kernels. To appropriately model limited support ranges, such infinite distributions have to be limited that they still fulfil the requirements for probability distributions (the function must be non-negative, and the probability space sums up to 1). The question is now where this limit should be set. The industry standard 6σ [90] defines this threshold as sixfold the standard deviation from the mean as adequate coverage (hence the name of the standard). Figure 4.20 shows the normal distribution and the probabilities until the 6σ -distance from the mean. This covers 99.999 999 8 % of the probability space. Only 0.000 000 2 % of values are outside this range, which corresponds to less than two outliers out of a billion ($2 \cdot 10^{-9}$). Hence, we had to limit them, and we did this by trying two approaches: Firstly, truncation and consequent normalization, and secondly, adding a reflection around the limits before doing the actual truncation and normalization.

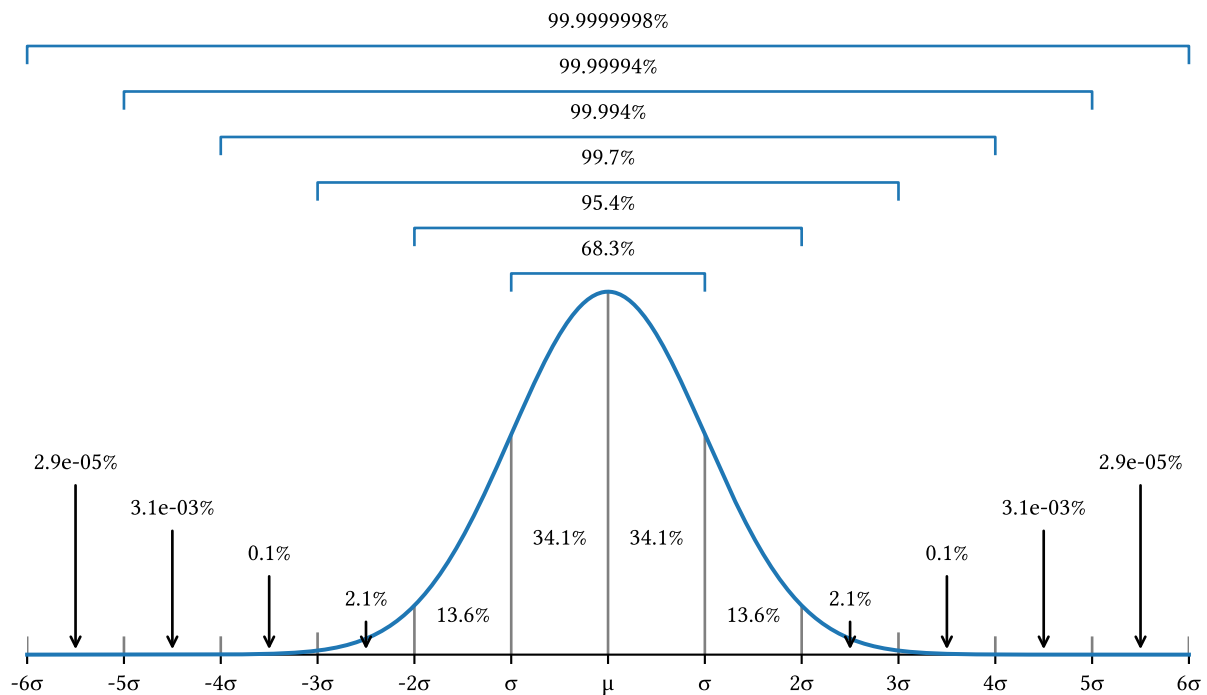


Figure 4.20: Normal distribution, and the probability intervals for the respective standard deviations.

Truncation The first (naive) approach was just to cut the excess areas at the domain boundaries and normalize the remaining area to 100%. While this works well for the central (inner) parts of the distribution, it underestimates the probabilities at the boundaries, especially in skewed distributions or when the boundaries are at high regions with high probability. Such situations happen when the estimated most likely values are near the boundaries or when the experts disagreed or were uncertain in their given judgments. This was bad because, especially in risk estimations, the values at the borders model the extreme scenarios (the so-called *black-swan events*), which could severely impact the total risk. Just truncating the distribution would result in inaccurate approximations for these edge cases. As a consequence of that, another approach had to be found: reflection.

Reflection The second approach also used truncation at the domain boundaries, but here, the distribution was reflected at the boundaries first. In other words, we added mirror images of the distribution at the lower and upper boundaries, and afterwards truncated it at the boundaries and normalized the remaining area to 100% again to get a probability metric. The reflection increases the probability density of the boundaries, which better reflects the probabilities of edge-cases and black-swan events that otherwise would have just been cut off. An illustration of this is shown in Figure 4.21a. The reflection approach worked out better, especially in the regions around the boundaries. Figure 4.21b shows an example where the likelihood at the boundaries more than doubled due to the reflection. Of course, this additional probability must come from somewhere, as can be seen in Figure 4.21b: The likelihoods around the maximum of the curve were slightly decreased.

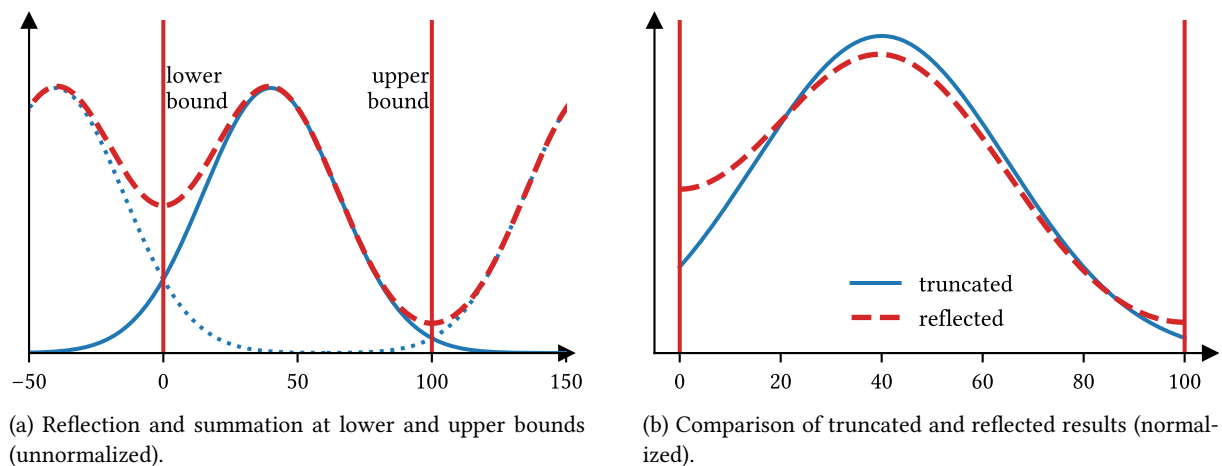


Figure 4.21: Limiting the support range of an infinite probability distribution at lower and upper boundaries.

We only reflect the distributions once at the boundaries, although theoretically, the reflected part of the distributions could hit the other boundary again. This double-reflection is not an issue in our case since we truncate the distributions at the 6σ -distance from the mean, where the probability is already very low. This means that the probability space reaching the other boundary again would already be $6\sigma + 12\sigma = 18\sigma$ away from the mean, which is neglectable. Furthermore, the impact and frequency attributes have a natural lower bound at 0. This boundary is independent of the 6σ -distance, therefore higher double-reflection effects could occur. However, we can safely assume that experts will not estimate negative values for impacts and frequencies. Therefore, the probability space on the support domain below 0 only emerges due to our used kernel-density-estimation method. This reasoning also holds for the vulnerability, which has two relatively narrow bounds between 0% and 100%, but again: Experts will refrain from estimating outside these bounds, and the kernel-density-estimation will only produce tiny spaces outside which are reflected, but are not large enough to hit the other boundary. Therefore, it is enough to reflect the distribution once.

4.5 Expert Judgment and Consolidation

We have described the structure and attributes of risk graphs, explained how the attributes are being propagated over the graph, and which mathematic concepts are involved in accomplishing this. Here, we cover the last part of the method design, which is about estimating the risk attributes.

In the following sections, we describe the method for expert elicitation and a combination of expert judgments to get reliable and defensible data of unknown or uncertain values. This is used for determining the risk attributes *frequency*, *vulnerability*, and *impact* in a risk graph. The method is based on structured expert judgment (Cooke's classical model [14]) and the IDEA Protocol by Hemming et al. [58], which we specialized for the assessing risks of cyber-security threats.

The idea of the *wisdom of the crowd* is to ask many experts to get a representative and significant sample size, which increases the confidence and minimizes the sampling error. However, we seldom can ask more than three to five experts for an assessment in the industry. These experts have to judge many values (depending on the risk graph's size), which would take a long time and therefore be very costly. This requires us to ask only a few experts and still get significant and correct results. Since the sample size is so small, weighing them equal could result in high uncertainty or imprecise results already if

there are just a few outliers who are erroneous or uninformative. Therefore, we have to weigh them, but the problem is how to define the weights: Cooke and Colson showed that weighting by their expertise, rank, or character traits could even be worse than equal weighting in some cases [159]. Therefore, either we train the expert to give better judgments or apply weighting factors on the experts to prefer those who give accurate judgments and degrade those who have performed poorly.

4.5.1 The Elicitation Process

As already mentioned in Section 4.1, we acquire the risk attributes by asking experts for the estimations. This is done in several rounds: First, the calibration, followed by two rounds of expert judgments. In between, the experts can discuss the results of the previous round and exchange information, insights, and experience. While the discussion fosters social effects for consent, experts are always asked individually and anonymously to state their estimations. This removes social pressure and allows the experts to express their honest individual opinions.

The results are combined using a weighted linear opinion pool, where the weights are determined based on the judgment performance of the experts during the calibration questions.

The general topology of the risk graph should already be established before the judgment. Otherwise, the discussion would be too extensive because the experts would have to come up with all the possible attack paths in addition to the judgment, which is already very tedious. Anyhow, the risk graph could be refined during elicitation if the experts deem it necessary. For each node, the respective value is judged under the assumption of a specific attacker type, e.g., industrial espionage, script-kiddie, or cyber-terrorist.

4.5.2 Estimation of Uncertain Risk Attributes

To determine the risks in a system, we have to acquire reliable estimations for the risk attributes *frequency*, *vulnerability* and *impact*. There are different ways to estimate these attributes. One easy but naive way is to state them using qualitative values, e.g., low, medium, or high. Qualitative values seem easy to judge, but they are highly subjective, have many flaws, and cannot be used in calculations. This is elaborated in much greater detail in the included publication [P9]. As a consequence, qualitative estimations should be avoided altogether – instead, quantitative estimations should be preferred. However, even when quantitative estimations are used, it is not enough to state a single-point estimate because of the considerable uncertainty in the judgments. Expert judgments are estimations that involve high uncertainty which cannot be neglected. Hence, experts should not only judge a single-point value but should model a complete probability distribution for their judgment which includes their uncertainty [11]. For practical reasons, this is done using descriptive statistics by stating, e.g., the mean and the standard deviation, or by stating the 90% confidence interval. This is simplifying the judgment while still allowing us to model it rigorously as a probability distribution.

For example, in the classical model, experts are asked to give prediction intervals by stating estimations for defined quantiles of their judgment, e.g., by stating the 5%, the 50%, and the 95% quantile. Based on these quantiles, a histogram is modelled using uniform intervals [14]. One problem here is that humans are imperfect in estimating probabilities or confidence intervals [74], which could lead to quite some errors [160]. This is why we expanded the judgment intervals to the minimum and maximum values because this does not involve probabilities or confidence intervals. Furthermore, we ask for another value that experts are already familiar with: the most likely value or the mode in statistical terms. With these three values – minimum, maximum, mode – a PERT distribution can be parametrized. Additionally, we include the confidence value λ of the modified-PERT distribution (see 2.3.4 for the

exact definition). This parameter influences the kurtosis of the distribution. High confidence causes the probability density to accumulate around the most likely value, making the curve narrower, while low confidence causes the distribution to spread out, making the curve broader.

4.5.3 Calibration

Before we can use the expert judgments as input values for assessing our system's risk, we have to define the judgments' quality by doing a calibration. The goal is to get quantitative weights representing the quality of the experts judging performance. By emphasizing the good experts and downgrading bad ones, the results will be more accurate.

The calibration is done using a questionnaire with calibration questions, which require similar knowledge and judging skills as the actual values we want to estimate. The only difference is that we know the true answers to these questions and can test how well the judgment fits. It is important to note that the experts should NOT know the true answers before the judgment; otherwise, the calibration does not test their actual judgment skills. For the context of cyber-security, judgment skills for the following types of estimations are needed:

- **Frequency:** How often will an event happen, e.g., over the next year?
- **Vulnerability (Probability):** What is the chance that an attacker can successfully execute an attack?
- **Impact:** How much is something worth? What are the direct and indirect costs if an attacker damages, compromises, or steals assets and data?

For each of the types, at least 5-10 calibration questions should be asked to get reasonable results [14, 13]. Experts can be trained to get better at giving judgments and ratings. This means that an expert should go through rigorous judgment training where they practice giving judgments that consistently cover the true value (e.g., in 90% of cases), while the uncertainty range is narrow enough to be still informative. Douglas Hubbard developed a training program for this and found out that it does not need long to become better – already, after a few hours of training, the accuracy, and precision can be increased [124, 13]. However, this applies to a specific domain only. Someone good at judging lengths or weights is not automatically good at judging percentages, or money, although our brain tricks us into believing that judgment skill can be transferred to all domains [74, 161]. This means that the calibration training should also be done in the same domain that will be assessed later.

4.5.4 Weighting the Experts' Performance

After we got all the estimations for the calibration questions, we can assess the experts' performance scores, consisting of two factors: **calibration score** and **information score**. The calibration score tells us how well the expert hits the true values, and the information score tells us how valuable the judgments are. Equation 4.5 describes the calibration score, and equation 4.4 explains the information score, which are used to calculate these scores. In short, the calibration score compares the probability distribution for the correctness of an experts' judgment to a normal distribution: the more often the expert hits the vicinity of the true value, the better the calibration score. The information score is the average entropy of the judgments compared with a uniform distribution over the range of judgments for all other experts in the same question: the smaller the range compared to the other experts, the higher is the information content.

The total score is then calculated by multiplying the information score and calibration score and normalized in comparison to the other experts. This defines the individual weight for an expert compared

to the other experts. After being normalized, this total score is used to weigh the actual judgments of each expert. The assumption here is that if an expert did well during the calibration questions, he or she would also do equally well during the actual judgments.

The calibration can be done independently of the actual elicitation. Only the same experts have to be included since the weights could change if the expert composition changes. This is due to the information score, which relies on the maximum range of judgments, and the normalization of the total score, depending on the other experts' scores.

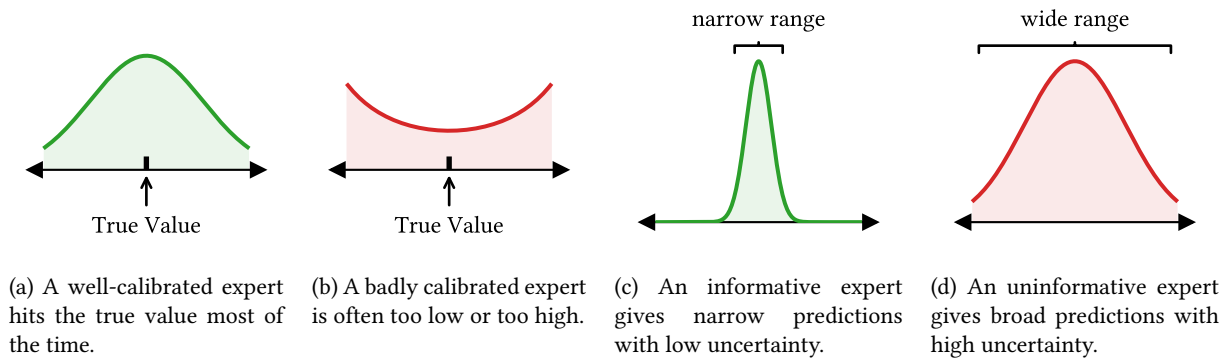


Figure 4.22: A good expert (Figures (a) and (c)) versus a bad expert (Figures (b) and (d)) regarding the calibration score and the information score to assess the quality of estimations.

Calculation The actual calculation of the weights is derived from the classical model by Roger Cooke [14, 70]. However, since we used fine-grained probability distributions instead of coarse uniform quantiles, we had to adapt the equations. The details are described in the included publication [P7], but an excerpt is replicated here for completeness. Afterwards, the equations for the calculations are also repeated here.

A log-likelihood χ^2 -test (also known as G-test) is used to compare this to an expert's actual realizations. The result is the calibration value and signifies how well the expert fits the assumed reference expert. For our method, however, we had to make some changes, summarized as follows: Instead of uniform areas, we decided to take the PERT probability distribution for the three-point estimate because it is the distribution recommended and used in the FAIR method, and this distribution is already established and trialled for its appropriateness to predict uncertain time schedules in project management [39]. Albeit our initial choice for PERT, RISKEE is not exclusively limited to it and works with other distributions. We use the distribution to get the probability with which the true value was predicted. Combining all realizations of the calibration questions, the resulting histogram H should correlate to a normal distribution \mathcal{N} . This is again tested via the log-likelihood χ^2 -test to calculate the calibration score, similar to how it was done in the classical model for expert elicitation by Roger Cooke [14]. Ultimately, this means calculating the Kullback-Leibler distance and applying it to the χ^2 -distribution to get the model fit. [162]

Information Score The information score is computed based on the entropy of a given estimation compared to a uniform background distribution over a reasonably large range $[\mathcal{U}_{min}, \mathcal{U}_{max}]$ (see Equation 4.4). It measures the relative information gain of a judgment compared to the uniform distribution over the same range. The range must span at least the minimum and maximum of all the experts' ranges for this question. The classical method even recommends adding a 10% margin around these values.

$$Information = \frac{1}{N} \sum_{Questions} \frac{-\sum_i^n \mathcal{P}_i \log\left(\frac{\mathcal{P}_i}{\mathcal{U}_i}\right)}{\log(\mathcal{U}_{max} - \mathcal{U}_{min})} \quad (4.4)$$

where: *Information* ... The information score for a single expert.

- \mathcal{P}_i ... The probability of bin i on the distribution of the expert.
- \mathcal{U}_i ... The probability of bin i on a uniform distribution.
- n ... The total number of bins over the support domain.
- \mathcal{U}_{min} ... The minimum value on the support domain.
- \mathcal{U}_{max} ... The maximum value on the support domain.
- N ... The number of questions in the survey.

Calibration Score The calibration score compares the distribution of actual probability results for an expert to an assumed ideal distribution which models a good expert. The probability results represent the probabilities the true values would get in the judgments. An ideal expert would give the true value the highest probability most of the time. The lower the probability for the true value is estimated, the lower is the calibration of an expert. The comparison itself is made via a G-test (similar to a χ^2 -test), which calculates the log-likelihood of frequencies in the supplied bins using the Kullback-Leibler distance. This is part of the general family of divergence tests called power-divergences for the goodness of fit tests.

$$Calibration = 1 - \chi^2 \left(2 \cdot \sum_i^n H_i \log\left(\frac{H_i}{\mathcal{N}_i}\right) \right) \quad (4.5)$$

where: *Calibration* ... The calibration score for a single expert.

- H_i ... The probability space of the judgment results for bin i .
- \mathcal{N}_i ... The probability space of the assumed normal distribution for bin i .
- n ... The total number of bins over the support domain.

Combined Score Calibration and information scores are then multiplied to get the final combined score (Equation 4.6). This combination allows comparing and ranking the experts. Experts with higher calibration and information scores get a higher combined score, and experts with bad judgment performance get a lower score, which is what we wanted. This combined score is normalized afterwards to get the weight w_i for the linear opinion pool of the experts (Equation 4.7).

$$Combined\ Score_i = Calibration_i \cdot Information_i \quad (4.6)$$

where: *Calibration_i* ... The calibration score for expert i .

Information_i ... The information score for expert i .

Combined Score_i ... The combined score for expert i .

$$w_i = \frac{\text{Combined Score}_i}{\sum_j^n \text{Combined Score}_j} \quad (4.7)$$

where: w_i ... The normalized weight for expert i .
 Combined Score_i ... The combined score for expert i .

Performance-based Weighted Decision Maker (PWDM) The expert weight w_i is then used to emphasize or reduce the influence of the respective judgment f_i given by an expert. Since the weights are already normalized, this reflects a weighted linear opinion pool (weighted arithmetic average). The resulting distribution represents the combined expert judgment used for decision-making (hence the name).

$$PWDM(x) = \sum_i^n w_i f_i(x) \quad (4.8)$$

where: $PWDM(x)$... The combined probability distribution function for the judgment.
 x ... Values taken from the support domain of the judgments.
 w_i ... The weight for expert i .
 f_i ... The probability distribution function for the judgment of expert i .

Implementation of the RISKEE Framework

Le bon Dieu est dans le détail.

Gustave Flaubert, 1821-1880

Summary: *This chapter describes the implementation of RISKEE as a risk propagation framework realized in Python. It first discusses the requirements for the implementation, then the framework architecture and design, and then shows some excerpts from the actual implementation, including examples of the source code. The implementation combines many of the methods and techniques discussed in the previous chapters to demonstrate the feasibility of RISKEE.*

◇◇◇

Here the RISKEE framework is elaborated. This framework was developed according to the RISKEE method to serve as a software prototype that implements the needed mathematical routines to calculate and propagate uncertain values for risk assessment. This prototype was used to demonstrate the feasibility and to serve as a platform for evaluations and experiments in our research. This chapter discusses the general requirements and shows some architectural considerations like the data flow, the processing steps, and beneficial design patterns.

5.1 Requirements

The reason for implementing RISKEE was to produce a scientific prototype that serves as a platform for experiments and demonstrations. For that, the requirements were as follows:

- *Requirement 1: Serve as an explorative and scientific prototype.* The most important reason we implemented RISKEE was to use it as a scientific prototype and demonstration platform. Hence, the quality does not have to be industrial grade but good enough to reliably perform experiments and repeatedly calculate use cases. The typical use cases of RISKEE are first to serve as a platform for frequent and fast trial runs and exploration. Secondly, to produce high-quality results and visualizations for scientific publications that are correct and repeatable.
- *Requirement 2: Distribution Arithmetic.* The framework should support mathematic operations like addition and multiplication based on uncertain values (probability distributions). It should

support arbitrary kinds of probability distributions to model any estimation and uncertainty. This includes finite and infinite, as well as continuous and discrete distributions.

- *Requirement 3: Fast development and exploration.* Since RISKEE is intended to be an explorative prototype. It should be possible to flexibly change the used internal design, mechanisms, and algorithms. A loosely coupled and flexible architecture was needed, which allowed quick changes and high development speed.
- *Requirement 4: Good library integration.* Since there are many sophisticated and robust statistics libraries, it should be possible to reuse them and integrate them into RISKEE – especially mathematical libraries to compute probability distributions for sampling, kernel-density-estimation, numerical optimization, and graph exploration are needed.
- *Requirement 5: Performance and Memory usage.* RISKEE is intended to run on current off-the-shelf desktop PCs, without real-time requirements, and without immediate responses. Typically, RISKEE is not used in time-critical situations. Memory usage should be kept under 8 GB of main memory to run on standard desktop PCs. The response times should be lower than a minute for a risk graph with approx. 50 nodes. For small graphs with less than ten nodes, the application should respond immediately (less than a second response time).
- *Requirement 6: Appropriate and pleasing visualization and reporting.* The results of RISKEE should be reported in a common data format for further processing and should be visualized in the form of risk graphs and distributions to be used in publications and reports.

5.2 Framework Architecture and Design

In this section, the RISKEE framework is described. First, the data flow is depicted to show which information is needed, where the data comes from and how it is processed, and then the structure and design of the framework itself is explained.

5.2.1 Dataflow: Input and Output

The data flow in RISKEE can be described using two distinct phases: The combination of expert judgments and the application of the RISKEE propagation algorithm. These two steps are performed in direct succession in the framework, but this could also be split up.

Phase 1: Combination of Expert Judgments

In this phase, the estimations coming from the experts are aggregated into a single combined probability distribution for each attribute. This is done using individual weights for the experts determined by a prior calibration survey. More details about how this calibration works are described in Section 4.5.

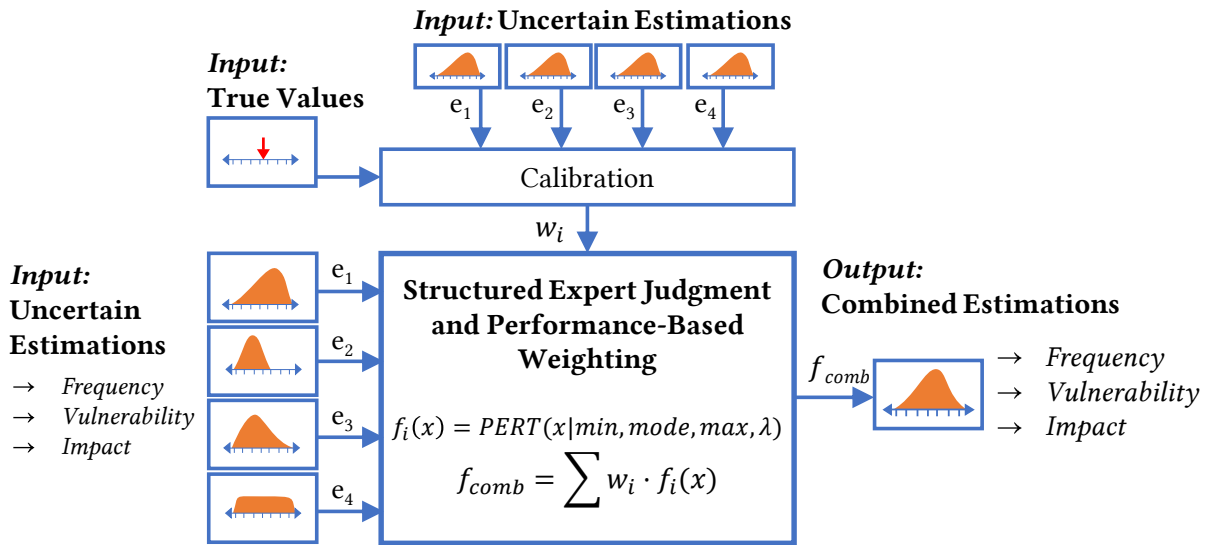


Figure 5.1: Dataflow for the expert judgment.

Input The experts need some information about the system. They use this information to make informed judgments about specific risk attributes. The minimal information needed are the components, the communication paths, valuable assets, and possible threats. This could already be provided as a preliminary risk graph, which could be expanded during the elicitation. The experts judge the attributes of frequency, vulnerability, and impact for the nodes in the graph. These attributes are modelled as probability distributions also to consider the respective uncertainty of the judgment.

Output After the judgments of the risk attributes are made, these values are combined with a weighted linear opinion pool, where the weights are determined using the performance metrics of information and calibration score. See Section 4.5.4 for more information. The results are the combined probability distributions for each of the attributes and nodes in the risk graph. This can then be used for the next step: the RISKEE propagation algorithm.

Calculation In this phase, the actual calculation steps are sampling, weighting the respective quantiles, and finally, aggregating the results and approximating a distribution using kernel density estimation.

- Step 1: Sampling. The distribution of every judgment is sampled using stratified sampling of a specific number of samples.
- Step 2: Weighting. The respective probabilities of each sample are multiplied with the respective expert-weight.
- Step 3: Aggregation. All weighted samples are combined and summed up into one huge list of samples.
- Step 4: Kernel-Density-Estimation. A kernel density estimation is made with Gaussian kernels and Fast-Fourier-Transformation to speed things up. The resulting distribution is reflected and truncated to the domain boundaries:
 - Frequency is limited between zero and the 6σ distance from the mean: $[0, \mu + 6\sigma]$

- Vulnerabilities are probabilities and therefore bounded between 0% and 100%: $[0, 1]$
- Impact is limited between zero and the 6σ distance from the mean: $[0, \mu + 6\sigma]$

The resulting probability distributions for the attributes are then used for the risk calculations on the risk graph.

Phase 2: RISKEE propagation algorithm

After the risk attributes are determined, the RISKEE propagation algorithm can be applied (see Algorithm 1). The results of this algorithm are risk distributions: A risk distribution for the whole system and individual distributions for all nodes and path segments in the risk graph. These can be visualized using loss-exceedance curves or can be used for further processing.

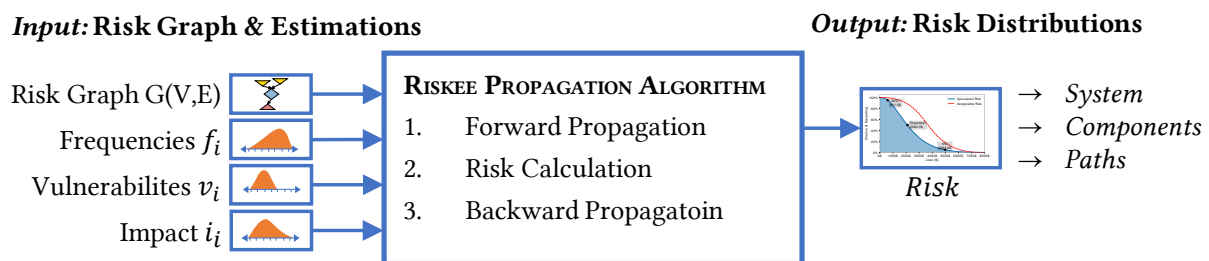


Figure 5.2: Dataflow for the RISKEE propagation algorithm.

Input As input, RISKEE needs the risk graph including the required attributes for each of the nodes: For entry nodes, the frequency attribute is required; for intermediate nodes, the vulnerability is required, and for impact nodes, the impact estimation is required. These values should be defined as probability distributions, but RISKEE also accepts single-point estimates. The graph is a directed acyclic graph (DAG) that can either be defined as a node-list or an adjacency-matrix. In summary, the needed input data and attributes are:

- Risk graph (directed, acyclic):
 - Nodes (Vertices)
 - Paths (Edges)
- Risk Attributes:
 - Attack frequency, specified as a probability distribution for each entry node.
 - Vulnerability, specified as a probability distribution for each intermediate node.
 - Impact, specified as a probability distribution for each impact node.
 - (Optional) Weights for branching paths.

The type of the nodes is determined automatically: All sources in the graph are entry nodes, all sinks are impact nodes, and everything in between is an intermediate node. By default, branching paths are weighted equally, but this can be adjusted by stating explicit weights.

Output After RISKEE calculated and propagated all the risk values, it returns the given risk graph with the same structure but enriched with probability distributions for the risk contribution of each node and edge. The total system risk is also calculated and returned separately. From the total risk distribution, descriptive statistics can be calculated like, e.g., the expected risk and standard deviation, or the 90% confidence interval. The distribution can also be visualized in the form of loss exceedance curves which can be compared to the risk appetite curve or any other threshold to decide if the risk is acceptable.

5.2.2 Communicating the Results

Communicating uncertain values and risk requires some considerations that were examined and published in the form of design patterns in one of our publications [P6]. Usually, uncertain values are communicated using the mean and the standard deviation, but this simplification could be highly problematic. Firstly, it implies that the distribution follows the normal distribution, being unimodal and symmetric around the mean. However, especially expert judgments are often asymmetric and multimodal, disqualifying the normal distribution because it may give a wrong picture [163, 164].

A more informative view can be given by stating an n-point quantile estimation of the distribution, e.g., by expressing a three-point estimate like the 5%, 50%, and 95% quantile, or as a five-point estimate by additionally stating the 25% and 75% quantiles. Even better would be a histogram or a complete visual depiction of the probability distribution showing, e.g., the probability density function (PDF), the cumulative distribution function (CDF), or the survival function (SF). However, a PDF is not very intuitive because the actual probability density is encoded in the area of the curve (which is hard to interpret). Therefore, in RISKEE, we decided to use the loss exceedance curve (LEC) [126], as a more intuitive visualization of risk by using the distribution's survival function. It shows the probability that a specific value will be exceeded over the whole range of possible values. The start is always at € 0 on the impact dimension and 100% on the probability dimension, and as the curve progresses on the impact dimension it continuously goes downwards until it reaches 0% on the probability dimension.

Since risk distributions are often long-tailed distributions, the endpoint could have a very high value or even be infinite. To cope with this, we elaborated on visualizations using logarithmic scales, which can be very misleading. While logarithmic scales cope well with huge differences in magnitudes, they are very misleading because of the high lie-factor. The lie factor is the ratio of the numeric difference and visual distance of values in a graph. According to Tufte [165], a good lie-factor should be between 0.95 and 1.05. Since a visualization on a 2D surface in Euclidean space is always linear, a logarithmically scaled depiction cannot achieve this. Therefore, in situations where the upper limit of the risk distribution is huge, the solution is to clip the visualization at some point from which the probability becomes neglectable.

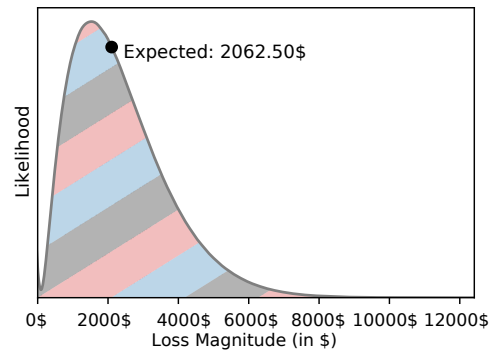


Figure 5.3: A risk distribution, showing the probability density over the domain of possible loss values.

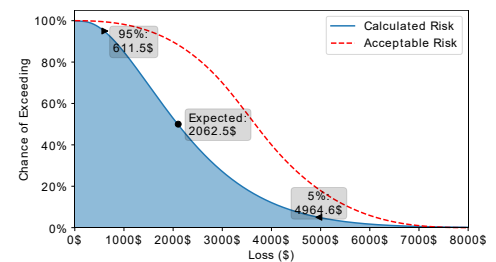


Figure 5.4: The two figures show the same risk distribution, once as a probability density curve, and once as the loss-exceedance curve (survival function).

Visualizing Expert Judgments Aside from the visualization of the results, often also visualizing the input data is needed. This enables analysing and comprehending the combined estimations better and gives a general image of the experts’ estimations. In RISKEE, we used so-called “guppy-plots” which is a term coined in paper [P7]¹. Guppy plots are a specialization of violin plots to visualize the range and distribution of multiple expert judgments. Figures 5.5 and 5.6 show examples for this. Both figures depict the same judgments, but Figure 5.5 does it on a linear scale and Figure 5.6 on a logarithmic scale. A linear scale is easier to comprehend since values can intuitively be interpolated by looking at it graphically. However, if the scale is huge, small differences are difficult to distinguish. Logarithmic scaling discriminates small as well as large differences visually but is much harder to comprehend and interpolate. This is due to the high lie-factor inherent in logarithmic scales (see Tufte et al. [165]). Here, the solution would be to use two visualizations: one showing the big picture and another showing only the areas of interest by clipping the oversized areas.

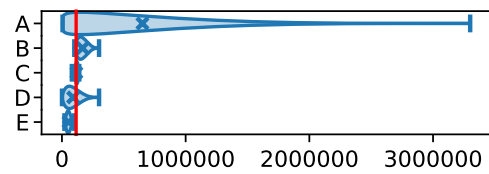


Figure 5.5: Linear scaling of expert judgments.

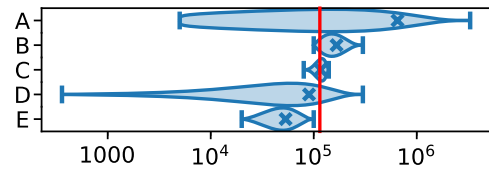


Figure 5.6: Logarithmic scaling of expert judgments.

5.3 Implementation

Here we describe the specifics of the implemented prototype. This is focused on the implementation in Python.

5.3.1 Programming Language and Libraries

The implementation goal was to develop a tool that enables us to elaborate and demonstrate the feasibility of this thesis’s theories. See the requirements in Section 5.1 for more information about the guiding principles and forces. We used Python as the programming language because it allows fast prototyping and includes many valuable libraries for scientific mathematics, e.g., statistics, working with probability distributions, or numerical optimization algorithms. Since we did not have commercialization of the tool in mind, we developed it as an academic prototype to show the algorithms’ feasibility and see the concrete problems discovered during actual implementation.

We used these languages and libraries for developing RISKEE:

- **Python 3.7:** We used the programming language Python because it allows fast prototyping and supports many scientific libraries. Initially, we used Python 2 because some of the libraries were not ported to Python 3 at the time, but in 2018 we switched to Python 3, although this meant a complete rewrite in some parts, since newer versions of the libraries changed their API (networkX and graphviz).
- **numpy:** A library for handling numeric math in Python. It is especially useful for working with numbers and matrices because it has highly optimized computation routines for that. It includes many useful functions that simplify working with numbers and mathematical operations.

¹Actually, the term “guppy-plot” was coined by my wife, Melanie Krisper, who called them “guppies” when she saw the visualizations the first time because they look like guppy-fish in a fishtank.

- **scipy**: Another library building upon numpy and extending it by many convenience functions and algorithms, e.g., for statistics. We used it heavily for statistical calculations and the implementation of the probability distributions.
- **statsmodel**: This library implements a fast kernel-density-estimation algorithm using Fast-Fourier-Transformation, which we used for smoothing the results of our calculations in *RISKEE*. The KDE in numpy also worked but was too slow for our purpose.
- **networkX**: Implements graphical models and many algorithms on graphs. It was used for representing the risk graph and finding and navigating all paths on the graph.
- **graphviz**: A library for visualizing the graphs using the dot-language. This was used for the visualization of the risk graphs in *RISKEE*.
- **matplotlib**: A library for visualizing data in diagrams like line charts or scatter plots. This was used for visualizing the probability distributions in *RISKEE* and this thesis.

We developed everything using Jupyter Notebook and Google Colab², which allowed us to work on the project regardless of the installed IDE and Python environment. With Python as the language and Jupyter Notebook with Google Colaboratory as the platform, fast development cycles and access from everywhere were the benefits.

5.3.2 Implementation of Probability Distributions

Probability distributions can be modelled either analytically using a specific probability density function (or a combination thereof) or numerically by storing discrete frequencies for bins with either fixed or dynamic widths like, e.g., Histograms, Dempster-Shafer-structures, or discrete probability distributions. While the analytic representation is the most precise, it cannot be used in arbitrary calculations since the combination of arbitrary probability distributions is not easily solvable analytically, especially if they have different support ranges. This is why we used the fallback to numerical representations whenever we had to do distribution arithmetic. This means that we sampled equidistant probability areas and calculated all combinations. Based on these results, we did a kernel-density estimation to model the resulting distribution. The benefit here is that we can use arbitrary probability distributions as input for each processing step and get a continuous distribution as output. We used two different modes of precision regarding the sampling: For exploration and trial runs, we used 100 bins distributed uniformly over the probability space. Every bin represents a 1% probability range, which induces a digitization error of 0.5% at maximum due to the binning. For calculating the final results, we used 5000 bins, resulting in every bin representing 0.02% probability space with a digitization error of 0.01% at maximum. The problem here was that using 5000 bins enormously slowed down the calculation, which was unacceptable during trial runs. Using 100 bins still gave an approximate result without impacting the performance. After approving that the approximate results were reasonable, we reran the calculations with higher precision to get more exact results in the end.

5.3.3 Class Design and Design Patterns

We heavily used existing libraries and classes already defined in numpy and scipy and StatsModel and FastKDE. So, the biggest problem regarding the system architecture was to make these libraries compatible with each other. For that, we used the design pattern ADAPTER [166]. We wrote an interface that incorporates the minimum set of features we needed for the calculations and implemented several adapters which transform the distribution classes of, e.g., scipy, or StatsModel to our minimal interface.

²Google Colaboratory: <https://colab.research.google.com/>

Figure 5.7 shows the base class `ProbabilityDistribution` which defines the minimal set of functions needed for RISKEE to work with.

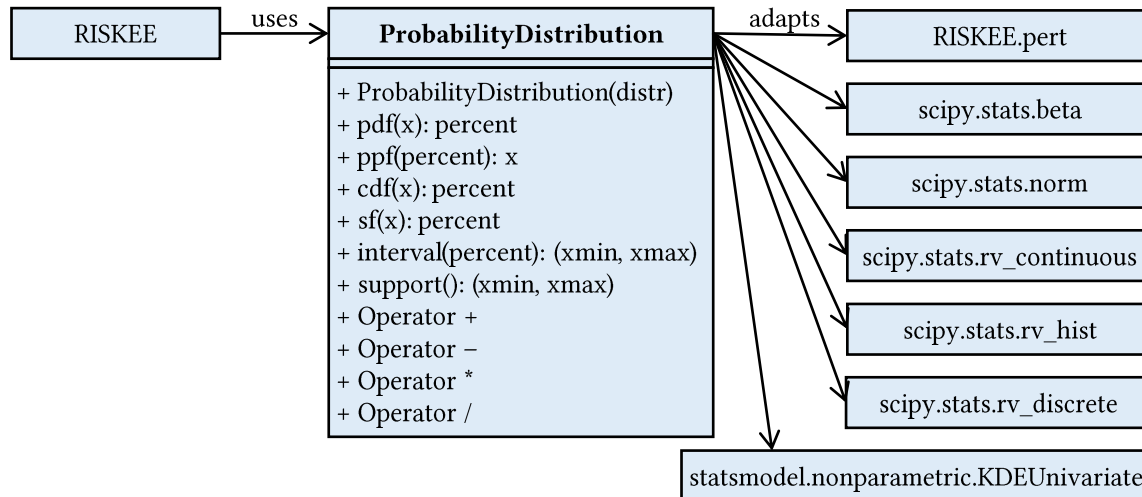


Figure 5.7: The class `ProbabilityDistribution` with its minimal set of functions for RISKEE.

`ProbabilityDistribution` contains three types of methods:

- Firstly, a **constructor** which accepts many other types of probability distributions that should be adapted.
- Secondly, a bunch of methods for **accessing the probability distribution**, e.g., the probability density function (PDF), the cumulative distribution function (CDF), or the percent point function (PPF).
- Thirdly, it supports **arithmetic operators** for addition, subtraction, multiplication, and division, which can be used in RISKEE for arbitrary operations.

One speciality of the `ProbabilityDistribution` is that it transforms different distributions into a continuous form to make them compatible. In our implementation, we even considered combinations of distributions with single-point estimates. For example `scipy.stats.rv_discrete`, and `scipy.stats.rv_hist` will automatically be converted into a continuous probability distribution using a kernel-density-estimation with the help of `KDEUnivariate`. By doing this transformation step, all distributions will be made compatible with each other, regardless of whether they are discrete or continuous. Furthermore, it limits unbounded distributions to a specific support range: If the support for the underlying probability distribution is infinite, it will be trimmed to the 6σ distance from the mean. By doing this, it is much easier to work with the distribution. The 6σ -distance was chosen because this covers almost the whole significant probability space of 99.999998%. Only $2 \cdot 10^{-6}\%$ of the probability space is cut off, which is neglectable according to the industry standard 6σ [90].

To summarize, the class `ProbabilityDistribution` supports many kinds of probability distributions to work with: finite and infinite, continuous and discrete, unimodal and multimodal, single and mixed. This feature makes it very versatile and comfortable to use in any framework. This is also described as a pattern in publication [P3] by the author (see publication [167]). The arithmetic operations also work with single values instead of distributions. The kernel density estimation is much faster in such cases since there are not so many values to consider. Another idea was to model single values with a uniform

distribution that is very narrow (a few thousand floating-point epsilons wide). The narrower, the more it would correspond to a single point value with the probability of 1. Due to floating-point limitations of the programming language, a too narrow distribution resulted in floating-point errors, while a too-wide distribution introduces unwanted uncertainty. But even a few thousand machine epsilons would still be much smaller than the 6σ range, which is the truncation boundary for unbounded distributions (a machine epsilon on modern computers is around $1.1 \cdot 10^{-16}$, compared to the 6σ cut-off of $2 \cdot 10^{-8}$).

Evaluation

One accurate measurement is worth a thousand expert opinions.

Rear Admiral Grace Murray Hopper.

Summary: *In this chapter, we evaluate RISKEE. We do this in three ways: First, we compare the results from RISKEE to the established classical model using 45 public studies from the DELFT expert judgment database. Second, we show the results of our own study in the form of a use case to judge cyber-security risks using RISKEE. Third, we discuss the application of RISKEE to a design space exploration scenario to determine secure design solutions.*

◇◇◇

The evaluation of RISKEE in this thesis consists of three parts which show different aspects of the method and the framework: First, we compare it to the classical model by Cooke et al. [14, 147, 11] using 45 publicly available studies based on structured expert judgment. The comparison shows that the results of RISKEE are consistent with the established and proven classical model, which indicates the correctness of the used model and mathematical routines.

The second part of this evaluation shows the feasibility of the process for expert elicitation in our own use case of structured expert judgments we did in a conference workshop in 2019 [162, 168]. Here, 17 experts were surveyed to judge the risks of a cyber-security scenario. The use case showed that the RISKEE method of calibration and expert judgment is feasible and applicable in a real-life scenario.

The third part discusses the application of RISKEE for design space exploration. This shows the applicability to a real-world scenario and compares the results to Bayesian attack graphs (BAGs). Here, the findings are that RISKEE returns better results than the commonly used BAGs but has the drawback of slower execution times and a higher demand for data.

6.1 Evaluation Part 1: Comparison to the Classical Model

In this part of the evaluation, we show the correctness of our method by comparing and correlating the results to the established classical model by Cooke et al. [14, 58, 148]. Over the last 30 years, the classical model (also called Cooke's method for structured expert judgment) has proven to be a reliable method for combining expert judgments based on uncertain estimates. To demonstrate the correctness

of RISKEE, we compare it to the classical model by correlating the resulting scores to each other. But first, we compare the applied methods and discuss the differences in the calculation methods.

6.1.1 Estimations and Distributions: Three-Point Estimates, Uniform, PERT

Both methods (the classical model and RISKEE) allow estimations of uncertainty ranges. The classical model uses uniformly distributed bins based on the quantiles for a 90% confidence interval. Mainly a three-point estimate is used, but sometimes also a five-point estimate. For example, a three-point estimate consists of estimations for the 5% quantile, the 50% quantile, and the 95% quantile. Based on this definition, the following four bins can be determined to create a histogram: [$<5\%$], [5%-50%], [50%-95%], and [$>95\%$]. The bins represent the respective probability spaces of 5%, 45%, 45%, and 5%. Figure 6.1a shows an illustration of the quantiles and the bins. It shows that this histogram is very coarse, and the individual bins are rectangular-shaped because the used inner distribution is uniform. RISKEE uses a three-point estimate, with the additional possibility of defining the confidence into the most likely value. For representation, it does not use a histogram with coarse uniformly distributed bins, but a modified PERT-Beta-distribution, as can be seen in Figure 6.1b. While the classical model uses a 90% confidence interval, RISKEE uses the full possibility range (100%). This is because, in RISKEE, the confidence is parametrized differently than in the classical model. While in the classical model, the confidence is implicitly determined by the 90%-confidence interval, RISKEE uses the confidence parameter λ . Therefore, experts can estimate the full possibility range and adjust the confidence for their specified most-likely value by changing the λ -parameter. The lower this confidence parameter is, the broader the probability density gets, and vice versa. In the extreme case of no confidence at all ($\lambda = 0$), the PERT distribution takes on the form of a uniform distribution over the whole range. In contrast, for very high values of λ (e.g., over 100), the distribution approaches a single-point estimate around the most likely value. For more information, see Section 2.3.4 in the Background chapter.

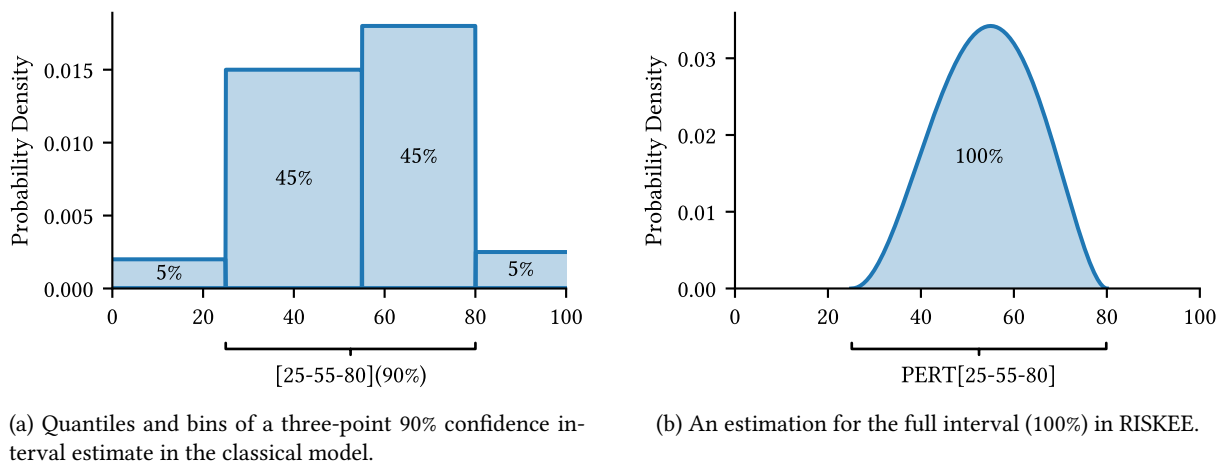


Figure 6.1: Models for estimations in the classical model and in RISKEE.

A final remark has to be made on the types of estimations in both methods: The previous sections only mentioned the default estimation types for both methods. In reality, both also allow more sophisticated definitions of the estimations. For example, while the classical model by default uses the 90% confidence interval with a three-point estimate, it also supports, e.g., a five-point estimate, stating the 5%, 25%, 50%, 75%, and 95% quantiles. Even a ten-point estimate based on arbitrarily distributed bins would be

possible (albeit not very intuitive). RISKEE also allows using different kinds of estimations, as long as they somehow can be represented as a probability distribution.

6.1.2 Structured Expert Judgment Studies Database

As the basis for our comparison, we used 45 studies done with structured expert judgment, which were made available online by Roger Cooke¹. These studies were also the basis for a cross-validation meta-study by Colson and Cooke [159], and are analysed and described in more detail in the online supplementary material of this meta-study [169].

They all used structured expert judgment as a method, asking experts for estimations of the probability quantiles or confidence intervals and weighting the experts based on their performance measured by calibration questions. In this thesis, only the calibration questions are of interest since they included the correct answers, allowing analysis and comparison of the classical model and RISKEE.

The studies we analyse in this evaluation are done by many researchers in different domains over the years 2006 to 2020 for topics, like, e.g., environmental research, health issues, air quality, disease control, volcanic and environmental hazards. They were done for several contracting parties, e.g., Universities of Wisconsin, Cambridge, Ottawa, Bristol, Maryland, and Medical Center Utrecht, several governmental organizations from the UK, Japan, Italy, and other institutes in the health and environmental sector. The majority of studies were performed by Willy Aspinall, TU Delft, and the UK Center for Disease Dynamics, Economics & Policy and others like, e.g., the University of Notre Dame, University of Maryland, University Medical Center Utrecht, Vicki Bier, Matthew Gerstenberger, and Benjamin Goodheart. More information is available in [159] and in the respective supplementary online material where the studies are described, analysed, and referenced in greater detail [169].

One significant difference between the methods is that the classical model defines a threshold for the calibration score to filter out badly calibrated experts. In RISKEE, they are still included but would not influence the final results since they would receive a low weight due to the diminishing calibration score. Another difference is how the estimations are represented: In the classical model, the probability for values inside the same bin is uniformly distributed. The resulting distribution has jumps in probability between the bins, which are rectangular-shaped. When the expert judgments are combined, these jumps become more fine-grained but are still noticeable as discontinuous rectangular jumps in the combined probability distribution. In RISKEE, an estimation is represented by the smooth and continuous PERT distribution. When these are combined, the resulting curve is smooth and does not contain discontinuous jumps in probability, although it could contain sharp corners if one distribution begins in the middle of another, which is often the case. However, these corners do not result in discontinuous jumps like in the classical model.

6.1.3 Comparison of Information and Calibration Scores

Both mentioned methods use information score and calibration score as performance metrics but calculate them slightly differently due to the different models underneath. The information score is a metric for the informativeness of estimations. It measures the width of a stated range (determined by the uncertainty) and compares it to the distribution with the lowest possible information content: the uniform distribution. Narrow estimation ranges provide more information and have lower uncertainty than broader estimation ranges. The second metric, the so-called calibration score, is a benchmark that measures how well the experts' estimations conform to an assumed ideal expert. To determine this, an ideal distribution for the estimation errors is assumed and compared to the actual realizations of an

¹http://rogermcooke.net/rogermcooke_files/45%20studies%20Cross%20Validation.zip

expert by using statistical tests. An ideal expert should have normally distributed accuracy, meaning that most estimations should model the true value quite well, but sometimes the estimations are too high or too low, which is ok as long as these situations do not occur too often.

Information Score Both methods calculate the information score, in the same way, using a χ^2 -statistical test which uses the Kullback-Leibler divergence to compare the entropy of the estimated distribution to the entropy of a uniform distribution with the same support range. Since the uniform distribution has the highest possible entropy, others with the same range can be compared to it, and the resulting relative entropy can be used as a factor for calculating the weights for the experts. The exact formula is stated in Section 4.5.4.

Comparing the information score for RISKEE and the classical model shows that the information score is strongly correlated between them, having a Pearson correlation coefficient of $r^2 = 0.67$. Values above 0.6 can be interpreted as very strong correlations [170, 171, 172, 173]. Although the distributions underneath are different for both methods, they both assume the uniform distribution as the least informative distribution. The strong correlation shows that both methods are consistent with each other in this regard. Figure 6.2 shows the information score for all 521 experts from the included studies. It can be seen that, especially in the regions of high information, the two methods sometimes differ by small amounts, but in the regions, with lower information score, they coincide quite well.

Calibration Score When it comes to the calibration score, the calculation differs between the methods. However, the basic idea is the same: To compare the distribution of the realized true values to an assumed ideal distribution for a perfectly calibrated expert. For the classical model, this means that frequencies of the true values should be similar to the probabilities defined by the bins of the estimation (e.g., for a three-point estimate, the true values should fall into bins 1, 2, 3, 4 with probabilities 5%, 45%, 45%, and 5% respectively). Here, again a χ^2 test is done to determine the statistical similarity of the distributions. In RISKEE, however, the model does not define such discrete bins. Therefore, the comparison of distributions has to be done differently: Here, the distribution of percentiles for the realized value in each of the estimates is determined. The distribution of these percentiles is compared to a reference distribution of an assumed ideal expert. Here we assume that he or she has a “normally distributed error behaviour”. In other words: The errors of a well-calibrated expert are assumed to be normally distributed. On the contrary,

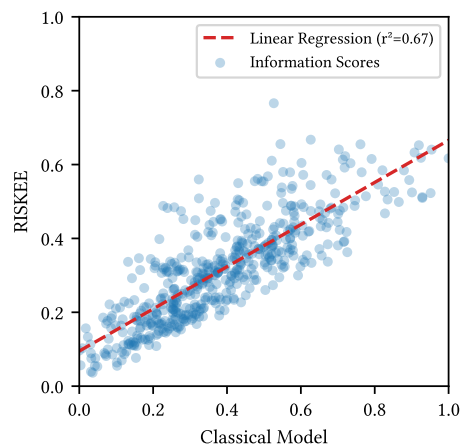


Figure 6.2: Comparison of the information score shows a strong positive correlation ($r^2=0.67$) between the methods.

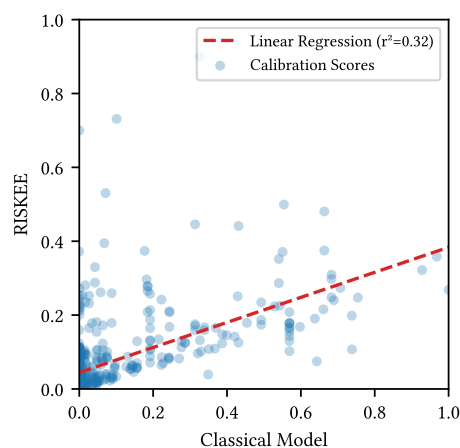


Figure 6.3: Comparison of the calibration score shows a moderate positive correlation between the methods ($r^2=0.32$).

if the distribution of percentiles for the true values is consistently too high, too low, or both, we spotted a badly calibrated expert. The exact calculation of the calibration score was described in Section 4.5.4. Again, the actual results differ due to the coarse definition of bins in the classical model compared to the fine-grained sampling in RISKEE. Additionally, the definition of the ideal reference distribution has also quite an impact on the results. A too narrow reference distribution only considers the well-calibrated experts, resulting in a situation where no expert would remain. To refrain from such arbitrary decisions and to get similar results like the classical model, we optimized the parameters of the reference distribution in RISKEE to resemble the best fit to the distribution in the classical model. Despite this adjustment, the correlation of the resulting information scores between the two methods was not as good as for the calibration score but still resulted in a moderate correlation ($r^2 = 0.32$).

6.1.4 Comparison of the Resulting Expert Weights

The actual weights for combining the given expert judgments are calculated similarly in both methods: The information score and calibration score are multiplied and then normalized in relation to all other experts. When the performance of information and calibration is bad, the relative weight should be low compared to the other experts. When both scores are high, the relative weight compared to the other experts should also be high. Figure 6.4 shows a comparison of the resulting weights calculated by the classical model and by RISKEE. Every point corresponds to the weight of a single expert in the 45 studies from the DELFT expert judgment database. In total, 521 experts were included in this comparison. The figure shows that both scores correlate moderately (with an r^2 -value of 0.41). The linear regression line is curved since the figure uses logarithmic scales. Here, logarithmic scaling was chosen because most of the resulting weights are very small compared to the whole value range. In this way, we can show the correlation better. On a linear scale, the small weights would have clustered near 0. This effect can be seen in the visualization of the calibration score in the previous section. This would have been even more extreme due to the multiplication of the scores; therefore, we used logarithmic scaling in the visualization.

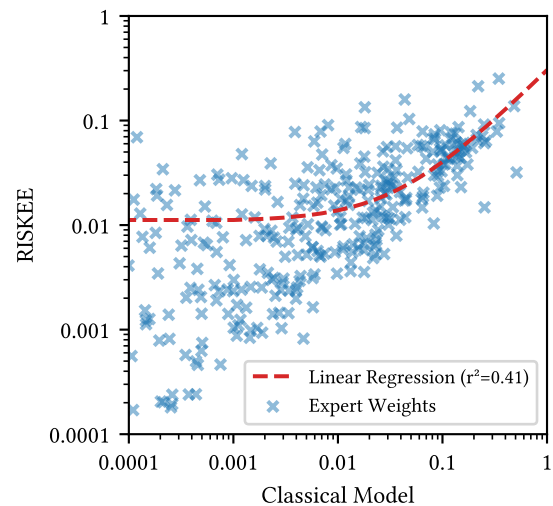


Figure 6.4: The comparison of the resulting weights for the experts shows a moderate positive correlation between the methods ($r^2=0.41$). The figure is using logarithmic scales, which is why the linear regression line is curved.

6.1.5 Summary for Evaluation Part 1

In summary, this part of the evaluation showed that RISKEE is consistent with the classical model, an established and proven method. We showed this consistency by comparing the results for the calculated information and calibration scores between the two methods based on 45 studies. While the correlation for the information score is very strong ($r^2=0.67$), the correlation for the calibration score is only moderate ($r^2=0.32$). This may be because the studies assumed a 90% confidence interval, which we extended to a 100% interval. In combination, the resulting weights still correlate moderately with $r^2=0.41$. This

shows that RISKEE delivers consistent results to the established classical model.

To use the data provided by the studies, we had to clean up the data files, which involved some changes and corrections. For example, several studies used the value of -999.5 to indicate an invalid or missing number, which we had to remove because it would distort the results. Furthermore, while most studies used a three-point estimate, some used a five-point estimate. We had to dismiss two of the values because the PERT distribution can only be parametrized using three attributes. One threat to validity could be that the estimations originally were stated assuming a 90% confidence interval. However, for the parametrization of the PERT distributions, we assumed them to be a 100% confidence interval, as has been shown in Figure 6.1. This mainly concerns the calibration score and is probably also why the correlation for the calibration score is lower than for the information score.

6.2 Evaluation Part 2: Use Case: FAST, FURIOUS and INSECURE

The following example describes a use case taken from the included publications [P7] and [P8]. This use case consisted of a survey among experts in a conference workshop to assess the cyber-security risks of a given scenario. Multiple experts were asked to judge the risk attributes for a specific cyber-security attack on Tesla cars. These estimations were used as input for RISKEE to calculate the total risk. The attack was taken from a publication by the COSIC group from KU Leuven [174, 4]. The scenario was derived from a well-known white-hat attack *FAST, FURIOUS AND INSECURE* on Tesla-S cars, exploiting some implementation issues in the passive keyless entry system (PKES) and imitating the paired key-fob using a rainbow table² containing pre-computed intermediate keys. With this hack, a car can be unlocked and started with minimal effort. Just physical vicinity to the car and the driver for a few seconds, but not necessarily at the same time. It consisted of four phases, which were represented as four nodes in our risk tree:

- **Phase 1:** Adversary acquires the car identifier wirelessly
- **Phase 2:** Adversary performs two constructed challenge-response requests with the key fob.
- **Phase 3:** The adversary recovers the key with the help of a pre-computed rainbow table containing intermediate keys.
- **Phase 4:** With the recovered key, the adversary unlocks the car, starts it, and drives away.

The experts had to assess the cyber-security risk for the following scenario: A car rental company in Leuven (the city where the mentioned attack was developed) has 100 Tesla-S cars and should be assessed for the risk of these cars being stolen. The experts had to judge the risk attributes frequency, vulnerabilities, and impact. First, we did an initial calibration to assess the performance for each of the attributes. Then we did two rounds of expert elicitation, discussing and presenting intermediate results in between. After the first round, the experts were allowed to discuss their opinions and estimations. The result of this exchange was that nearly all experts increased their estimations slightly, which resulted in twice the risk evaluation in the end. These slight changes amplified due to multiplication over

Figure 6.5: The questionnaire for the expert judgment, depicting the attack path and fields for the risk attributes.

²A rainbow table is a pre-computed lookup-table which is used for reverse hash lookups.

multiple steps and resulted in this tremendous increase in risk. We discuss this in more detail in the included publication [P7] and [P8].

Calibration Results The calibration results showed that the experts, unfortunately, had rather bad judgment performance. Figure 6.6 shows the calibration results. For comparison, we added the reference curve for a well-calibrated expert, which shows that none of the participants came close to this. A well-calibrated expert would have given estimations that almost always estimate the true value near the centre of the estimation. This calibration result is consistent with Colson et al. [159] who showed that more than half of the experts would not pass a statistical significance test due to having a p-value of lower than 0.05. Colson also found out that in 20% of the analysed studies, not a single well-calibrated expert was present – similar to the situation in our survey. We would have never expected these results, and this alone already shows the significance of the calibration. Without the calibration, we would not know how to weigh and combine the expert so that the results will be as good as they can be. At least we can compare the experts relatively to each other: In this survey, we see that some experts were at least a little better than the rest. We also see that one expert (V) was significantly worse than the others. Figure 6.6 shows the distributions of the calibration scores and Table 6.1 shows the final weights.

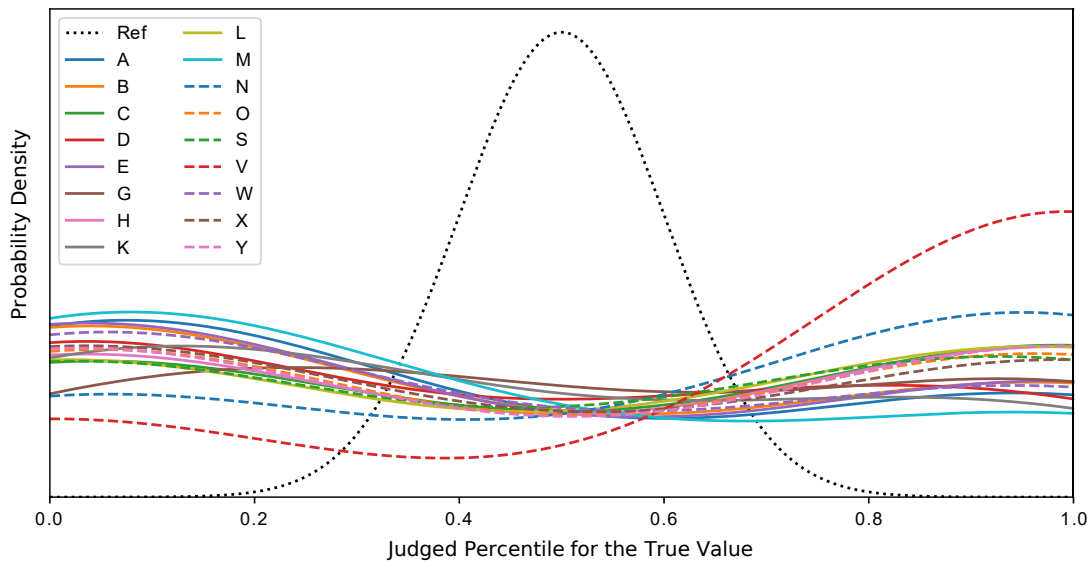


Figure 6.6: The calibration results shown as distributions for the percentile of the true values. The x-axis represents the percentile of a probability distribution, going from 0 to 1. The y-axis shows the respective density. The black-dotted line represents the reference curve for a well-calibrated expert.

K	G	D	M	W	A	E	O	C	S	B	X	H	Y	L	N	V
0.10	0.09	0.08	0.08	0.08	0.06	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.04	0.04	0.02

Table 6.1: The weights for the participants in decreasing order. These weights were calculated based on the performance on the calibration questions.

Survey Results After combining the estimations using the calibration weights (shown in Table 6.1) and calculating the risk for the stated attack path in RISKEE, the results show a tremendous increase

in risk between the first and the second expert judgment round. In the first round, the 90% confidence range for the risk was [9 980–1 560 000] €, while in the second round, it was [24 100–3 090 000]. Moreover, also the average risk doubled, with 400 000 € in the first round and 869 000 € in the second. The discussions revealed some interesting insights to the experts, which generally increased the risk awareness. Each estimated parameter increased after the group discussion. In the end, the uncertainty was increased, and the total expected risk more than doubled.

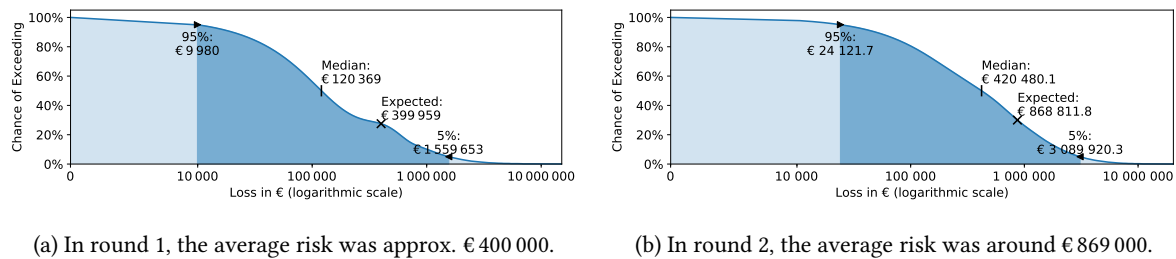


Figure 6.7: Comparison of Round 1 and Round 2, showing that the resulting risk more than doubled.

This part of the evaluation showed the feasibility of the expert elicitation process. We conducted an expert judgment workshop using two rounds of expert judgment with an initial calibration phase. From the calibration, we were able to determine the judgment quality of the experts, which was used for weighting the subsequent judgment rounds. The discussion in between gave the experts the opportunity for knowledge exchange, and the visualization using the loss exceedance curve gave intuitive insight into the results.

6.3 Evaluation Part 3: Providing a Cyber-Security Metric for Design Space Exploration

The second use case for RISKEE was the application and integration of the framework to calculate a comparable metric for cyber-security in a design space exploration. This was done in collaboration with Lukas Gressl and was published in the included papers [P10] and [P11]. The goal was to find optimal solutions for a chip design out of a plethora of design variants. Optimal solutions are secure, have the lowest power consumption, provide high enough performance and are still affordable regarding the costs. For the security metric, RISKEE was used and compared to Bayesian attack graphs (BAG). The relevant finding for this thesis is that RISKEE found better solutions than the BAG method because it also included the attack frequency and the impact attributes. However, this improved quality came with a cost because RISKEE was far slower in execution. This was mainly caused by the facts that it uses probability distribution arithmetic, and the python libraries were not optimized for performance. In addition to that, each method called from C to python introduced a slight overhead, which summed up over time because millions of different design variants were tested, each requiring a separate function call. Additionally, the python source code itself was not optimized for performance, although it used numpy routines underneath, but still needing the python runtime as glue code. This also made it difficult to parallelize the application because, for every instance, an own Python interpreter had to be started, which impacted the memory footprint. Figure 6.8 shows an example result for a risk evaluation in a design space. Each point represents a design variant: a specific combination of components for a chip design. The x-axis represents the power consumption of the design variants, and the y-axis shows

the respective performance. The colour of the points indicates the count of fulfilled (or broken) security goals, which was evaluated using RISKEE and Bayesian attack graphs. Only the solutions which fulfilled all security goals were interesting for realization. The next task was to find those with the lowest power consumption among these secure solutions while still providing high performance for an affordable price tag. The possible solutions were then compared to existing solutions designed and built by a big company for producing secure microchips, and they matched.

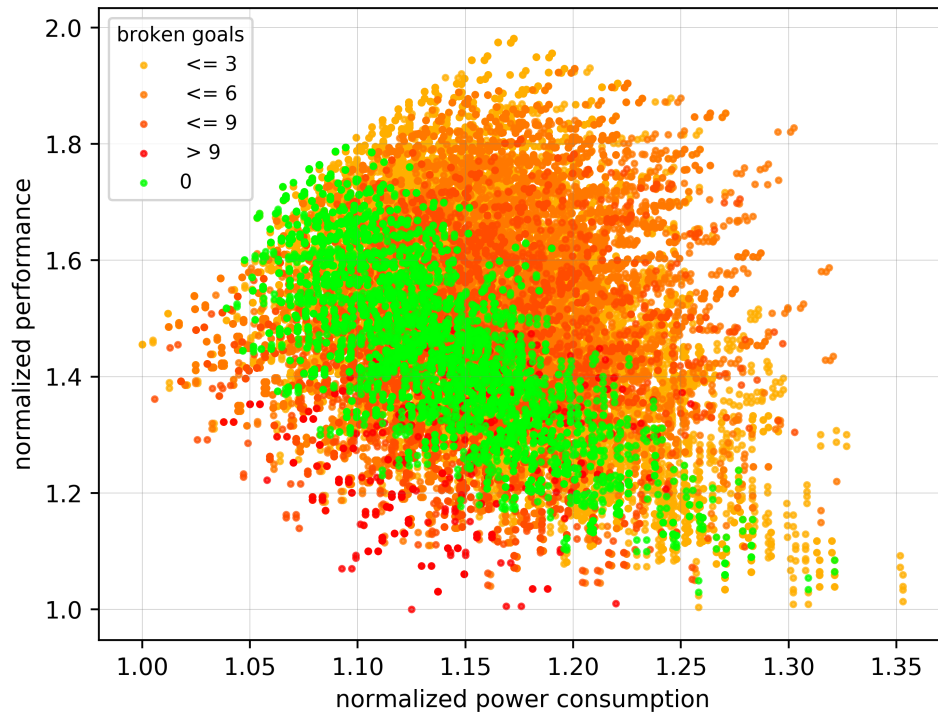


Figure 6.8: Scatter plot of risk evaluations by RISKEE for all solutions in a design space.

In this use case, the emphasis was on applying RISKEE to a real-life scenario and the integration into another software to calculate comparable cyber-security metrics for many design variants. Therefore, we did not do an expert elicitation but defined the needed risk attributes ourselves or derived from other existing assessments like Common Criteria or CVSS. The risk graph was created dynamically during runtime to represent each design variant. After applying the RISKEE propagation algorithm, the resulting probability distributions for the risk were reduced to the mean to have an orderable single-point metric used as a discriminator for the security goals. This part of the evaluation shows that RISKEE can be integrated into design space exploration tools.

Conclusion

*It is difficult to make predictions,
especially about the future.*

Danish Proverb, attributed
to Niels Bohr

Summary: *In this chapter, we describe some open issues and possible paths which can be followed up in future work. This includes refinements about temporal considerations like, e.g., modelling the decay of system resistance and increasing future uncertainty. Furthermore, the specification of dynamic concepts like evolving attackers is still open. Finally, using RISKEE to solve the administrator's dilemma and automatic correction of experts' biases are two huge topics that could be followed up in future dissertations.*

◇◇◇

To conclude this thesis, we first discuss the future work and then summarize the key points presented in this work. Regarding future work, here only the very general points are stated. Tasks like performance optimizations or increasing the robustness and usability are still open, but here we focus more on the scientific ideas for future publications instead of technical issues. In the end, the conclusion summarizes the key points again and states general remarks about the impact of this dissertation in the greater context of risk assessment and expert judgment.

7.1 Future Work

Here the future work concerning the method and the implemented framework is described. We discuss changes of risks over time and also changing and evolving attackers. Furthermore, we had the idea of a component system, which would make it easier to model bigger systems and improve performance. Moreover, extending the work to solve the Administrators' Dilemma of a multi-objective optimization problem would be a future path. We think that there is still some potential for a sophisticated influence analysis and graph-theoretic approaches to find the optimal mitigation methods. Also, an unfinished task was the conversion between existing security standards and evaluation and the quantitative risk attributes in RISKEE. Finally, we had the idea of extending the expert judgment method by finding performance scores for the experts and even computing correction factors to improve the judgments automatically and adaptively.

7.1.1 Temporal Considerations: Resistance Decay and Future Risk Predictions

Our current model takes a snapshot of the expert judgments and calculates the total risks out of them. While the judgments may be valid for the next year, they do not allow projecting into the far future. To do that, the uncertainties must be adjusted accordingly, and the system resistance must decay over time. The vulnerability and its subfactors – system resistance and attacker strength – are based on the current assumptions about the maximum strength of attacks. However, attackers get stronger over time, and with that, the ratio between the system resistance and attacker strength changes. If a system does not change, its resistance will decay, and the vulnerability to the attackers' ever-growing strength will get higher. Therefore, projecting five years into the future should yield a much higher risk and uncertainty than analysing the current year's estimations. In mathematical prediction models, such uncertainty is represented by the prediction or confidence intervals, but for risk assessment, in cybersecurity, these techniques are not applied yet.

7.1.2 Modelling an Evolving Attacker

With every attack, the attacker gets more knowledge and capabilities since he learns something about the system. We want to investigate this dynamic aspect of an attack in future work. Currently, it is assumed that the attacker does not change its strength and capabilities over the multiple steps of a single attack. According to the cyber kill chain, [175, 176], an attack could stretch out over multiple days and even months, which means that the attacker capabilities could change tremendously in between. Until now, we did not consider such aspects in the model; therefore, some future work could analyse the impact and dynamics of evolving attackers.

7.1.3 Component Concept and Subsystems

One idea which was not elaborated deeper was to implement a component concept in the risk graph. This could have several benefits: Firstly, nodes in the graph could be grouped in a component that can be developed and reused independent of the primary model, like in the component concept for fault trees [177]. Secondly, the component model could also involve attack groups or vulnerability groups, which have similar properties. Gressl et al. [178] targeted this by matching attacks with vulnerabilities to see which components are affected by specific attacks. On the other hand, multiple similar vulnerabilities could be tackled in one step, e.g., by applying the same configuration to the machines within the same vulnerability group. The current model in RISKEE does not include such groupings. Such groups, categories, and sub-clusters could be analysed and elaborated to extend the model's expression capabilities in future work. Another aspect to this is that the performance could be improved by, e.g., clustering sequential nodes together or using dynamic programming on repeating sub-trees in the risk tree to reuse already calculated results.

7.1.4 Solving the Administrator's Dilemma

In the end, the whole process of risk analysis is about making the right decisions on how much risk is tolerable and which measures should be taken to decrease the risk to a tolerable level. For this, a multi-objective optimization problem has to be solved: How can we reduce the risk to a tolerable level, using only limited resources, time, skills, and personnel? Furthermore, often also social and business objectives further complicate this decision. Generally, this is called the "Administrator's Dilemma" [179, 63]: "[...] *how to select a subset of security hardening measures from a given set so that the total cost of implementing these measures is not only minimized but also within budget and, at the same time, the cost*

of residual damage is also minimized” [179]. Shortly: How to select the best mitigations with the least resources? RISKEE does not solve this problem but helps by giving a quantitative picture of the risks and analysing the impact of mitigation measures by calculating the risk reduction in terms of money. Gressl et al. [180] already try to solve this in their security-enhanced design space exploration, but completely solving the administrator’s dilemma is still up for future work.

7.1.5 Graph-Theoretic Considerations

The graph structure of risk graphs also allows for graph-theoretic algorithms, like searching for minimal cuts or applying semantic ordering and calculating centrality or connectedness metrics. Such metrics and minimal-cut sets could be used to find the nodes with the most connections, the most paths to impact nodes, or having a very strong influence on the total systematic risk. Calculating the shortest paths in the graph could determine the most obvious attacks and further define weights on the edges to make it more realistic. Maybe also analysing the topological aspects of the risk graph could reveal some insights. In this thesis, we did not investigate graph-theoretic approaches further, but this would be a good continuation point for the future.

7.1.6 Conversion between Risk Estimation Models

One major issue of the different risk estimation models is that they are incompatible with each other. Every standard and model defines its categories unique to it and tailored to the respective domain. With RISKEE and its quantitative approach, we get a ratio scale based on some monetary values, which can be compared. The task for future research would now be to either convert this result into the other risk categories, or even better, to convert the qualitative judgments of, e.g., CVSS, Common Criteria, or SAHARA into quantitative values, which can be used in RISKEE and result in a risk-based on a ratio scale with its respective probability distribution. Houmb et al. [181] used specific attributes of CVSS to at least reuse the probabilities for their Bayesian approach. The next step would be to extract the impact values and frequencies to calculate the risk. By adapting and converting the values between the standards, it would also be much easier to embed RISKEE into existing methods and standards to be used more widely in the future. Here, a great benefit is that RISKEE does not redefine the whole risk assessment process but is compatible with existing processes and could just replace the calculation techniques.

7.1.7 Automatic Expert Calibration and Correction using Machine Learning

During the work on the thesis, one idea was to apply machine learning to find the best calibration weights for the experts. Maybe in the future, calibration, and information score could be entirely replaced by training a neural net using the calibration question and get the optimal weights. Even further, we thought of developing a regression model that weights the experts and even corrects their judgments by their biases. For example, if an expert always states too low predictions, they could be shifted upwards to improve them, and if an expert tends to give too narrow predictions (hence not catching the true value), we could apply a correction factor to stretch them. However, this correction is a double-edged sword since it would decrease the information score while increasing the calibration score. However, it is still preferred to have inaccurate results (huge uncertainty) than wrong results (not catching the true value). Applying machine learning or using optimal filters like, e.g., the Extended Kalman Filter could also optimize these correction factors and improve the experts’ judgment performance, but this is up to future work.

7.2 Conclusion

In this thesis, we showed an approach for assessing the risk of cyber-security in distributed networked systems. This approach is based on structured expert judgment and the RISKEE propagation algorithm using risk graphs. In structured expert judgment, experts give estimations of some uncertain attributes using probability distributions. In RISKEE, we used PERT-distributions for this, which allow the specification of the minimum value, the maximum value, and the most likely value. In addition, the confidence in this most likely value can also be stated. The results are probability distributions that reflect the expert's estimations. These estimations are then combined in a weighted linear opinion pool. The weights for the pool are determined in a calibration survey by measuring the judgment quality of the experts. Here we used two metrics inherited from the classical model for structured expert judgment: information score, which measures the uncertainty in the estimation, and calibration score, which measures how well an expert conforms to an assumed model of an ideal expert. Together, these scores determine the judgment quality of an expert and the weight for the combination of the estimations. The actual attributes which are estimated are attack frequency, vulnerability, and impact. These attributes have to be assigned to the possible attacks which could occur in the analysed system. Furthermore, the attacks can be split up into multiple steps over a whole attack path, and these can be combined into attack graphs. This results in risk graphs that model the attacks and communication paths in a system. The uncertain risk attributes are propagated from the attack surface of a risk graph to the impact nodes, where the actual risks can be calculated. Afterwards, these risks are propagated back again to see the individual influences. This is done using the RISKEE propagation algorithm. The result is a probability distribution that depicts all the possible outcomes in the form of a loss-exceedance curve which shows the probabilities of exceeding a certain amount of losses. This can be used to make an informed decision about mitigation measures and prevention strategies.

We examined the correctness of the used model for structured expert judgment by comparing it to the classical model, which showed a strong correlation among the results. Furthermore, we showed the applicability of the method by applying it in a workshop and assessing the risks for a cyber-security incident. Lastly, we tested the RISKEE propagation algorithm and the application of risk graphs in a design space exploration that compared the results to a Bayesian attack graph.

The method and the framework presented in this thesis were designed with an emphasis on cyber-security in mind. However, with a few minor adaptations, it could be applied to all kinds of graphical models that have to deal with uncertain values. Furthermore, the idea of forward and backward propagation of risk could be applied to other models.

While most of the research used in the thesis was done earlier, the thesis itself was written during the corona pandemic in 2020/2021. In hindsight, maybe the RISKEE method could have even been applied to assess the risks of the corona crisis since this also involved huge uncertainty and lack of knowledge. Perhaps this would have helped get a clearer picture of the actual risks and decide on the measures in a more informed way, freed from anxiety, fear, and other human biases.

Stay healthy; Stay safe; Stay secure – Slàinte mhath!
Michael Krisper

Appendix: Publications

Here, the publications for this thesis are listed with their full text. Every paper is preceded by a one-page summary that states the core ideas and contributions. Figure A.1 shows the publications (structured by year and conference). The papers in this thesis are presented at conferences and published in the conference proceedings, except for paper [P6] about the problems in risk matrices published online in arXive and paper [P8], a workshop report published on the conference web page. Aside from [P6] and [P8], all the papers are peer-reviewed, and papers even went through a shepherding and mentoring process over multiple iterations. The publications were presented and discussed at high-quality conferences, having at least a CORE ranking of B. The papers were written together with other members of the Industrial Informatics group and other colleagues at the Institute for Technical Informatics at Graz University of Technology. Some papers resulted from combined efforts with the Austrian Institute of Technology (AIT), Vienna.

Publications

[P1] Physical Quantity: Towards a Pattern Language for Quantities and Units in Physical Calculations (p. 96) This paper presents design patterns for implementing the propagation of physical units in mathematical equations. These patterns are the basis for later patterns on implementing uncertainty propagation.

[P2] Use-Cases for Uncertainty Propagation in Distributed control Systems (p. 118) This paper shows some general use-cases for uncertainty propagation in dependable systems. It shows that knowing the uncertainty over multiple calculation steps (in a graph structure) allows for sophisticated methods like, e.g., approximate computing or device fingerprinting.

[P3] Patterns for Implementing Uncertainty Propagation (p. 132) In this paper, patterns for calculating uncertainty propagation were described. It combines self-describing values (based on the physical quantities patterns) with operator overloading to propagate the uncertainty attributes forward. While this paper focuses on Gaussian uncertainty, it also supports more sophisticated kinds using arbitrary distributions.

[P4] Towards Integrated Quantitative Security and Safety Risk Assessment (p. 140) This paper shows that existing methods have some gaps in modelling the defender side of cyber-security attacks because they base on threat-modelling, which emphasizes attacks and the attacker's capabilities

more than that of the defender. For that, the basic notions of RISKEE were used (graphical system models with range estimations and propagation of uncertainty). It also establishes the combination of multiple experts to get more realistic and reliable results.

[P5] RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber-Security (p. 156) This paper established the RISKEE method for calculating risk in a graphical model of a system using the weighted expert judgment of estimated value ranges with probability distributions and propagating them forward and backward overall attack paths in a risk graph.

[P6] Patterns for Communicating Numerical Uncertainty (p. 170) This paper explores different ways how to communicate uncertainties to humans. Humans are biased, especially when it comes to probabilities and high uncertainties like in risk assessments. To avoid that, appropriate visualizations and ways of communication are discussed in this paper.

[P7] Assessing Risk Estimations for Cyber-Security Using Expert Judgment (p. 186) This paper presents the demonstration of a cybersecurity assessment using the RISKEE method and expert judgment. By asking several experts for range estimations using probability distributions and weighting them based on their judgment quality, the resulting risk is more accurately and precisely reflected than equally weighting or stating them as single point values.

[P8] Expert Judgment for Cyber-Security Risk (p. 202) This workshop report further describes the case study for assessing cybersecurity risks using expert judgment and RISKEE as the method. This adds to the paper [P7] by showing more details about the used process (the IDEA protocol, Delphi method, and structured expert judgment), as well as the calibration scores and the results of the two judgment rounds. This was published on the conference website in addition to the paper [P7].

[P9] Problems with Risk Matrices using Ordinal Scales This paper discusses and summarizes several problems with risk assessment using ordinal scales and risk matrices. Such types of assessments are used in many established risk management methods. However, using ordinal scales and risk matrices leads to quantification errors, range distortions, risk inversion, undefined arithmetic, and many more problems which could make an assessment worthless. Thus, we make a plea for using quantifiable methods instead of qualitative assessment.

[P10] Towards an Automated Exploration of Secure IoT/CPS Design Variants (p. 220) In this paper, RISKEE is compared to a classical Bayesian attack graph method for design space exploration of smart card chip designs. The results show that RISKEE returns more informative and significant results at the cost of higher computational needs and with that lower performance. The results of the automated design space exploration with the inclusion of security aspects are comparable to the actual system designs which domain experts have come up with in real products. Therefore, it can be used to find initial proposals for secure system designs.

[P11] Towards Security Attack and Risk Assessment during Early System Design (p. 236) This paper advances the previously established automated design space exploration method further by optimizing the runtimes of the algorithm while still getting high-quality results. It proposes pruning the search space using the Bayesian approach and then feeding only the valid subset of solutions to RISKEE, which returns higher quality results by including the frequency and impact of attacks but

needs more time for the calculations. This results in an automated design space exploration, which is fast due to the Bayesian pre-filtering and has high quality due to using the RISKEE on this sub-set of solutions.

Other Publications Here, other papers by the author are listed, which are not directly related to this dissertation, but still were created along the way. All these publications were also peer-reviewed and presented at various conferences:

- Krisper, M., Iber, J., Rauter, T., & Kreiner, C. (2017). **Insertion Spaces**. Proceedings of the 22nd European Conference on Pattern Languages of Programs EuroPLoP 2017. Irsee, Germany. <https://doi.org/10.1145/3147704.3147717>
- Krisper, M., Iber, J., Kreiner, C., & Quaritsch, M. (2017). **A Metric for Evaluating Residual Complexity in Software**. In J. Stolfa, S. Stolfa, R. V. O'Connor, & R. Messnarz (Eds.), *Systems, Software, and Services Process Improvement, EuroSPI 2017* (Vol. 748, pp. 138–149). Springer International Publishing. Ostrava, Czech Republic. https://doi.org/10.1007/978-3-319-64218-5_11
- Rauter, T., Höller, A., Iber, J., Krisper, M., & Kreiner, C. (2017). **Integration of Integrity Enforcing Technologies into Embedded Control Devices: Experiences and Evaluation**. 2017 IEEE 22nd Pacific Rim International Symposium on Dependable Computing (PRDC), 155–164. Christchurch, New Zealand. <https://doi.org/10.1109/PRDC.2017.29>
- Iber, J., Krisper, M., Dobaj, J., & Kreiner, C. (2018). **Dynamic Adaption to Permanent Memory Faults in Industrial Control Systems**. *Procedia Computer Science*, 130, 392–399. International Conference on Ambient Systems, Networks, and Technologies ANT 2018. Porto, Portugal. <https://doi.org/10.1016/j.procs.2018.04.058>

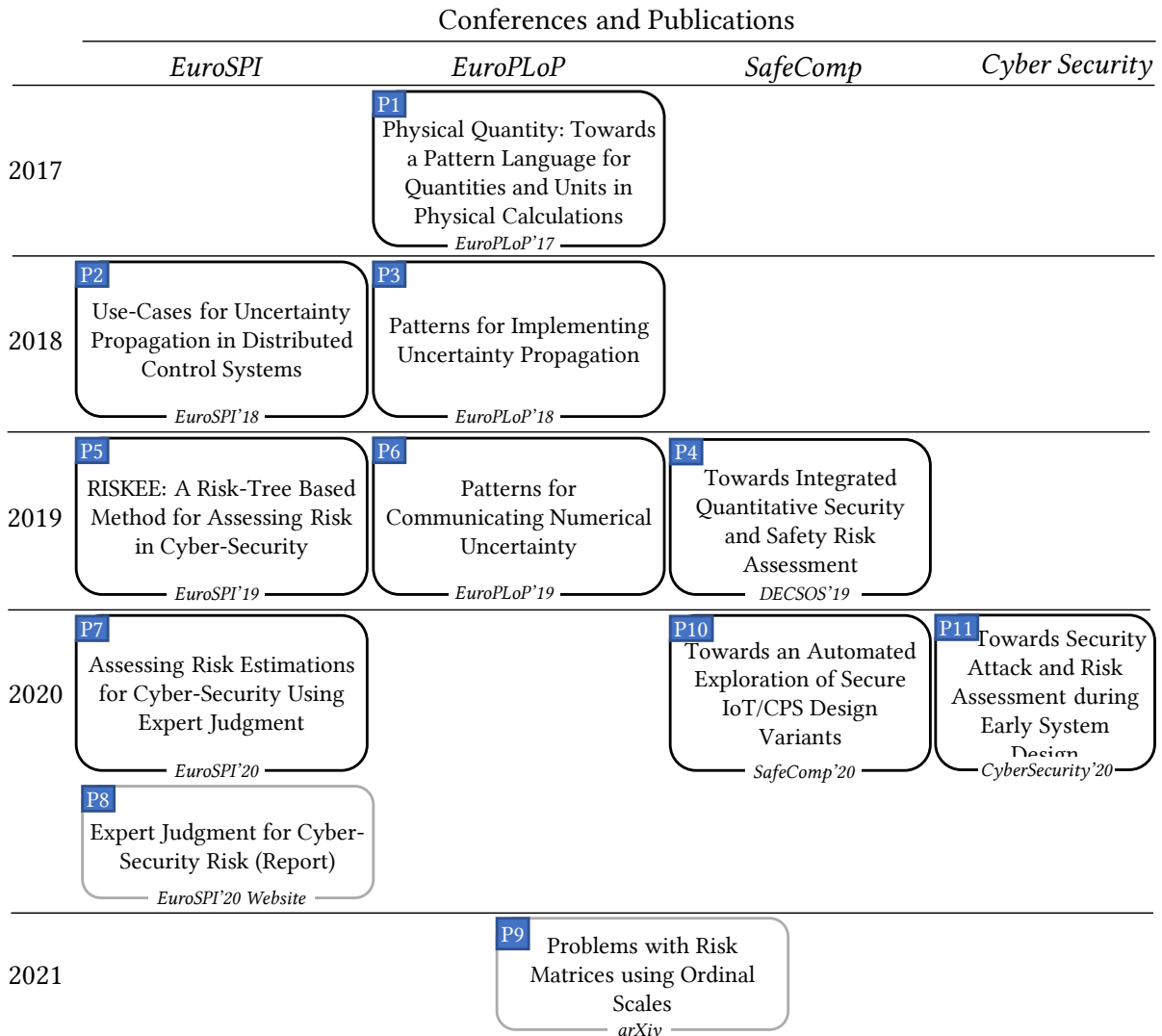


Figure A.1: Paper-Overview: An overview of the included publications.

A.1 [P1] Physical Quantity: Towards a Pattern Language for Quantities and Units in Physical Calculations

- Authors: **Michael Krisper**, Johannes Iber, Tobias Rauter, Christian Kreiner
- Year: 2017
- DOI: 10.1145/3147704.3147715.
- Bibliography-Reference: [182]
- Presented and discussed at EuroPLOP'17 conference.
- Published in EuroPLOP'17 conference proceedings (ACM).

Summary This paper presents design patterns for implementing the propagation of physical units in mathematical equations. Design patterns are solution templates for recurring problems. The context is implementing applications for physical simulations or calculations. The problem is that in normal programming languages, the implemented equations just use floating-point types. Floats only depict numerical scalar values, while the units are neglected. Especially in physics, the unit system is fundamental. A robust system of units has to ensure that all used units in the equations fit together. This becomes especially evident when using numerical optimization algorithms or geometric algorithms (e.g., interpolation of values in a triangulated map of measured values), where the algorithms themselves are unitless but calculate values that have physical semantics. Using a unit system allows for an additional check of correctness during runtime, as well as already during compile time. The following patterns are established in this paper:

- **Physical Quantity:** Create an object containing the numerical value alongside its SI-units stored as a 7-tuple of exponents for each of the base units in the SI unit system. This object must also override arithmetic operators: Addition, subtraction, and comparisons are only allowed for objects with equal units and return a value with the same unit-tuple. Multiplication and division either add or subtract the exponent-values in the resulting unit-tuple. This is done every time an operator or mathematical function is used at runtime.
- **Typesafe Quantity Interface:** Create explicit types for specific units in the SI unit system. Use these types for parameter definitions in functions to ensure type-safety at compile time. This can be combined with the Physical Quantity by taking it as the base class and just fixing the unit-tuple to a specific unit combination.

In addition to that, the following proto-patterns (pattern candidates) are identified: Convertible Quantity, Uncertain Quantity, Quantity Matrix, Quantity Collection, Validatable Quantity, Quantity Factory. Especially the Uncertain Quantity was elaborated in further papers about uncertainty propagation.

My Contribution This paper was completely written and conceived by me. The contribution of the other authors was in the form of discussions, feedback for improvements, and corrigenda. Not mentioned as authors but also influencing the paper were the shepherd Andreas Rüping and the workshop participants who gave feedback during the conference.

Copyright ©2017 Authors. The version included in this dissertation is a permitted reprint of the published version, which can be accessed in the ACM Digital Library here: <https://dl.acm.org/doi/10.1145/3147704.3147715>

Physical Quantity: Towards a Pattern Language for Quantities and Units in Physical Calculations

MICHAEL KRISPER, JOHANNES IBER, TOBIAS RAUTER and CHRISTIAN KREINER, Institute of Technical Informatics, Graz University of Technology

In this paper, an approach is taken towards a pattern language for physical quantities in software applications. The central pattern, `PHYSICAL QUANTITY`, is described as well as some needed candidate patterns revolving around. The `PHYSICAL QUANTITY` design pattern is a specialized version of the `QUANTITY` analysis pattern, optimized for the SI unit system. It is intended for the physical and mathematical domains where calculations, arithmetic, conversion, and simulations are the most used functionalities. Its emphasis is on type safety, dimensional analysis, performance, and convenient syntax. Supporting candidate patterns for handling tolerances, validation, conversion, or matrix operations are shortly described. The target audiences are software engineers and practitioners working in the area of physical simulations and calculations.

CCS Concepts: •**Software and its engineering** → **Patterns**; *Designing software*;

Additional Key Words and Phrases: design patterns, type safety, quantities, unit system, si units, simulations, calculations

ACM Reference Format:

Michael Krisper, Johannes Iber, Tobias Rauter, and Christian Kreiner. 2017. Physical Quantity. *EuroPLoP* (July 2017), 20 pages. DOI: <https://doi.org/10.1145/3147704.3147715>

1. INTRODUCTION

In this paper, we describe the `PHYSICAL QUANTITY` pattern as well as a pattern language revolving around it. `PHYSICAL QUANTITY` is a design pattern for physical and mathematical calculations and simulations that use the SI system of units as a basis. Like the `QUANTITY` pattern by Martin Fowler [Fowler 1996a], the `PHYSICAL QUANTITY` combines a numerical value with corresponding unit information but is tailored towards efficient storage of the units in the SI system, fast arithmetic, and dimensional analysis checks at run time. In this paper, also the `TYPESAFE QUANTITY INTERFACE` is presented, which adds many specific types for units to do unit-checks even at compile-time.

This paper is organized as follows: First, the background and basic information about the SI system is given. Afterwards (Section 3) Related Work with the position and context of this paper is described. Section 4 gives an overview of the pattern language. Afterward, the first two patterns are described in canonical pattern form [C2 2011]. The `PHYSICAL QUANTITY` is described in Section 5, and `TYPE-SAFE QUANTITY INTERFACE` in Section 6. In Section 7 the candidate patterns are shortly outlined. Section 8 presents libraries and frameworks which also implement the `QUANTITY` pattern and were an inspiration for the `PHYSICAL QUANTITY`. In the end, a conclusion is given.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or to publish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EuroPLoP '17, July 12-16, 2017, Irsee, Germany

© 2017 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-4848-5/17/07...\$15.00

<https://doi.org/10.1145/3147704.3147715>

2 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

2. BACKGROUND

Unit-systems are used in many applications ranging from scientific, physical, or mathematical applications to medical applications and financial applications. In physical simulations, the main usage is modeling the real world in software models and calculating and simulating physical formulas and equations to achieve some results. Examples for such are simulating combustion engines to determine their optimal working parameters, or optimizing the flow of fluids in pipelines, or calculating electrical installations, or monitoring power plants.

The idea for the PHYSICAL QUANTITY pattern and its related patterns arose during working on VECTO: a project for simulating the fuel consumption and CO2 emissions of heavy-duty vehicles at the Graz University of Technology [Hausberger et al. 2016]. In this project, we had to implement physical and mechanical laws in the form of equations and formulas for modeling the power train of trucks and buses. In the first iterations of the software, a floating-point data type was used, but it soon became apparent that we must enforce the units of variables in order to ensure the correctness of equations. The QUANTITY pattern by Martin Fowler [Fowler 1996a] was used as a structural basis for our design. During implementation, we stumbled upon requirements and forces which are unique to the domain of physical calculations and simulations and were not covered in previous works. This led to the specialization of the QUANTITY pattern for the physical domain: PHYSICAL QUANTITY. The patterns described here are influenced by the SI unit system implemented in the VECTO project [Hausberger et al. 2016], but also incorporates many aspects of other existing frameworks and products.

2.1 The Need for Unit Systems

Many applications which apply physical calculations to simulate real-world scenarios use some form of a standardized system of units, like the very common SI-System [ISO 2009]. For the calculations, these applications have to use a numeric type internally (e.g., integer, float, or double). They have to handle input and output of such values, e.g., reading files of simulation parameters or displaying the results in a graphical user interface appropriately. A simple way of doing this is to use floating-point variables for all calculations and using their formatting capabilities to bind them to the interface. This simple way of carrying out physical calculations is error-prone due to conversion errors, wrong equations, wrong formats. Programming languages, compilers, and even the most sophisticated IDE's can not give hints on whether a mathematical equation is implemented correctly or not. In addition to that, there are many issues with floating-point types (like rounding and cut-offs) [Reiser and Knuth 1975; Kahan 1996; 2001], and special numbers like infinity and Not-a-Number (NaN) have to be handled. With a unit system, it is possible to write mathematical equations for physical relations between quantities. Symbols and variables in equations may have a unit that has to be handled correctly according to the rules of unit arithmetic and dimensional analysis. Examples of SI units used in equations are (units are enclosed in brackets “[]”):

- Acceleration \vec{a} is the change of speed \vec{v} over time t : $\vec{a} [m/s^2] = \Delta \vec{v} [m/s] / \Delta t [s]$
- Force \vec{F} is mass m times acceleration \vec{a} : $\vec{F} [N] = m [kg] * \vec{a} [m/s^2]$
- Work W is a force \vec{F} acting along a path \vec{s} : $W [Ws] = \vec{F} [N] * \vec{s} [m]$
- Power P is work W applied over time t : $P [W] = W [Ws] / t [s]$

In the examples above, several SI units were used: m for Meter, s for Second, kg for Kilogram, N for Newton, and so on. To ensure correctness of the units in the equations dimensional analysis is used. For example, when you divide a length in meter by a time in second, the result must be a velocity in meter per second [m/s]. The arrows above variables indicate that these are directed measures (vectors). Vectors have a direction they are pointing to, which very often also must be taken into account.

2.2 The SI Unit System (Système international d'unités)

The SI unit system is a standardized system of measures [ISO 2009]. It establishes seven dimensions, which are the basis for all physical values: length, mass, time, electric current, temperature, amount, and luminosity. For every dimension, it exactly defines one base unit by physical constants or prototypes. An SI unit consists of a numerical value v (for value) and a corresponding unit $[Q]$. $[Q]$ is a 7-tuple consisting of the exponents for the 7 SI base units [ISO 2009; BIPM 2006; NIST 2008] as can be seen in Definition (1).

$$\begin{aligned} \text{SI Unit} &:= v [Q] \\ v &:= \text{a scalar value } (v) \text{ or a vector } (\vec{v}) \\ [Q] &:= [\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta] \hat{=} [[m]^\alpha \times [kg]^\beta \times [s]^\gamma \times [A]^\delta \times [K]^\epsilon \times [mol]^\zeta \times [cd]^\eta] \end{aligned} \quad (1)$$

Dimension	Base Unit	Symbol
length	meter	m
mass	kilogram	kg
time	second	s
electric current	ampere	A
thermodynamic temperature	kelvin	K
amount of substance	mole	mol
luminous intensity	candela	cd

Table I. : SI Base Units according to the standard [ISO 2009; BIPM 2006]. Every one of them is exactly defined in terms of physical constants or prototypes.



Fig. 1: SI BASE UNITS: m, kg, s, A, K, mol, and cd are the base units in the SI system [ISO 2009; BIPM 2006].

2.2.1 Derived Units. In the SI system, also derived units are defined. These are specific combinations of base units, which sometimes have their own names for simplicity or historic reasons [NIST 2009]. Examples for such are: Frequency (in Hertz) $[1/s = Hz]$, Force (in Newton) $[kgm/s^2 = N]$, Power (in Watt) $[kgm^2/s^3 = W]$, Pressure (in Pascal) $[kg/ms^2 = Pa]$. Commonly they are written as numerator and denominator, but every combination of units can be represented in the 7-tuple given in Definition (1) — even derived ones. Here are some examples of this:

$$\begin{aligned} - v [Q] &= v [[m], [kg], [s], [A], [K], [mol], [cd]] \\ - 5 kg &= 5 [0, 1, 0, 0, 0, 0, 0] \\ - 2.8 m/s^2 &= 2.8 [1, 0, -2, 0, 0, 0, 0] \\ - 7 Nm = 7 kgm^2/s^2 &= 7 [2, 1, -2, 0, 0, 0, 0] \end{aligned}$$

2.2.2 Prefixes. Prefixes simplify the handling of very big and very small numbers. They depict decimal coefficients for every base unit to avoid writing many 0's (zeros), e.g. 25000 meter can also be written as 25 kilometers; 0.001 meter is 1 millimeter. Some of the most used prefixes are giga (10^9), mega (10^6), kilo (10^3), deka (10^1), centi (10^{-1}), milli (10^{-3}), micro (10^{-6}), nano (10^{-9}). A comprehensive list is defined in [NIST 2008]. It should be noted here, that the base unit for weight already includes such a prefix: kilogram.

4 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

3. RELATED WORK

Martin Fowler described the QUANTITY as an analysis pattern in his book *Analysis Patterns - Reusable Object Models* [Fowler 1996a]. He thoroughly discussed the application in observation and measurements and showed how this could be used in medical (e.g., Blood Pressure Values) and the financial sector (e.g., Money). In addition to that, Joe Yoder [Yoder et al. 2000] showed the application of these patterns to the medical sector. Although it is pretty obvious to use quantities in the physical and mathematical realm, we could not find any pattern literature about this. Nevertheless, many libraries implemented this (see Section 8 *Known Uses*).

We distinguish between three different areas of application based on their requirements to the unit-system: physical/mathematical applications, medical applications, and financial applications (see Figure 2). In this paper, we specialize in the physical/mathematical area.

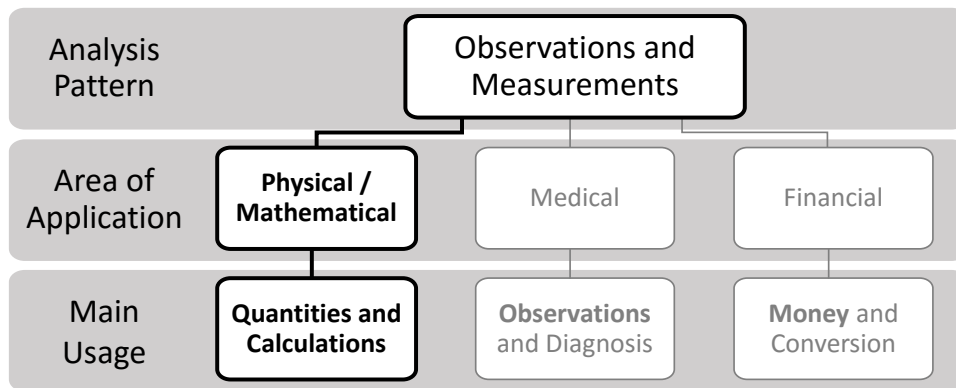


Fig. 2: AREA OF APPLICATION: The PHYSICAL QUANTITY patterns are residing in the area of physical and mathematical calculations.

3.1 Distinction To Other Areas Of Application

3.1.1 Medical. In medicine the observational aspects of values prevail. After recording the units, observations and diagnoses are made, and therapies are proposed based on the values. Here the challenge is detecting diseases based on the measurement values and indicators and finding reasonable therapies. Due to the direct impact on the life quality and health of the patients, the justification of therapies and historical data have to be recorded in detail, and those data have to be set in relation to each other. The recordings of historic and diverse data, together with their semantic meaning, is needed in this field. Here Martin Fowler's Architectural Patterns for Observations and Measurement are very useful [Fowler 1996a]. Also, Yoder [Yoder et al. 2000] described the OBSERVATION pattern based on the medical field of application.

3.1.2 Financial. The financial sector heavily relies on the values of objects and different currencies. Ever-changing exchange rates and conversion factors result in transactions that are heavily dependent on the point in time they are made. Shares and stocks are traded with always changing values, and financial applications have to cope with these changes in real-time. Banks account for huge amounts of money and therefore have to apply special semantics and rules for calculating money values (e.g., division rules, special rounding rules, handling of extra pennies). Martin Fowler wrote about this in his MONEY pattern [Fowler 1996b] as well as his QUANTITY pattern [Fowler 1997].

4. PHYSICAL QUANTITY PATTERN LANGUAGE

In this section, we give an overview of the pattern language revolving around the PHYSICAL QUANTITY pattern. Figure 3 shows a graphical overview of the pattern language. PHYSICAL QUANTITY is a specialization of the QUANTITY pattern by Martin Fowler [Fowler 1997]. It combines values with the SI unit system and supports dimensional analysis checks in the basic arithmetic operations. The TYPESAFE QUANTITY INTERFACE pattern enforces the use of explicit quantity types in method signatures and interfaces to check them at compile-time. This results in a robust type system that ensures the correctness of the used units and dimensions in all calculations and function calls. The PHYSICAL QUANTITY can be expanded by the CONVERTIBLE QUANTITY pattern, which allows conversion between different units. To limit the allowed values in a quantity, the VALIDATABLE QUANTITY provides a pattern for defining limits and bounds to the values which can be validated during run time. In systems where quantities have defined measurement tolerances, the UNCERTAIN QUANTITY allows for storing them and taking them into account in calculations (propagation of uncertainty). The QUANTITY COLLECTION pattern stores a list of similar quantities in a list efficiently. This is most useful when working with a big list of values that all have the same unit type. To bring them all to live the QUANTITY FACTORY implements a mechanism to create quantity objects easily.

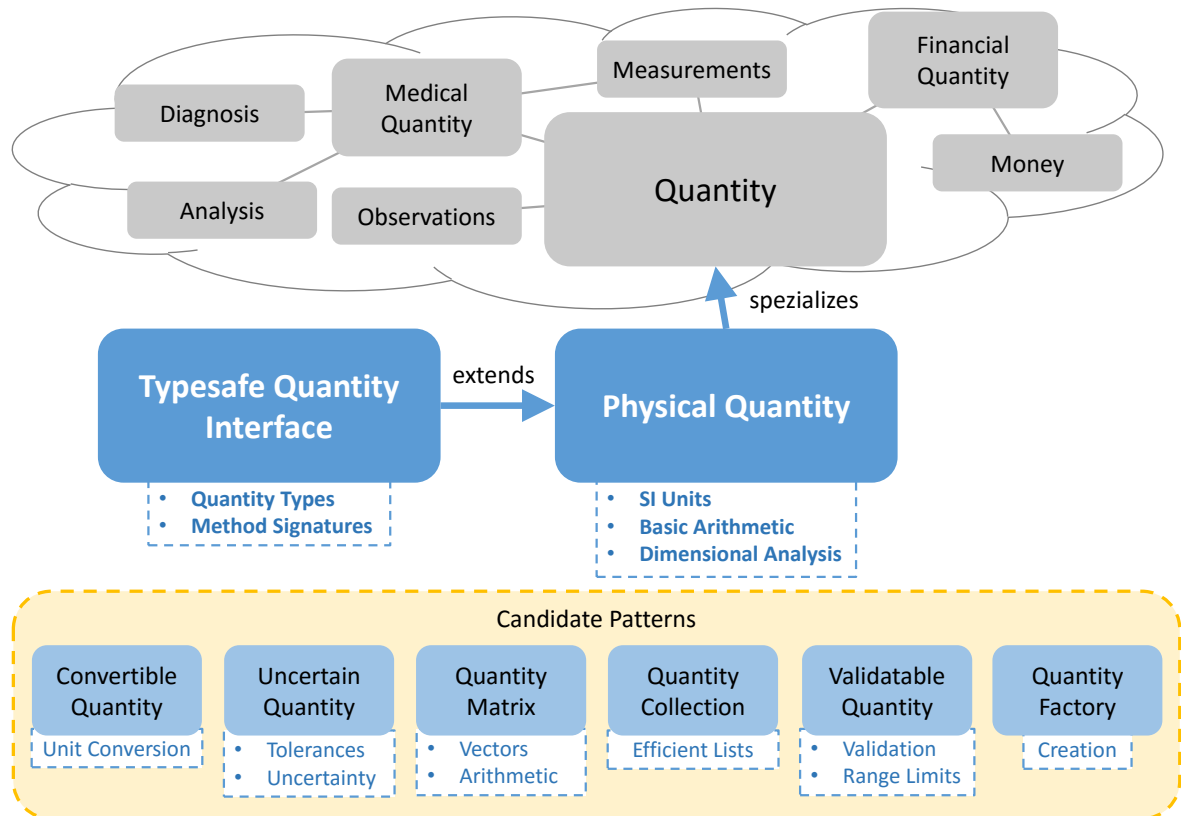


Fig. 3: PHYSICAL QUANTITY PATTERN LANGUAGE: A brief overview of the related patterns and candidates.

6 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

5. PATTERN: PHYSICAL QUANTITY

Also known as: *Quantity, SI Unit, Unit-System, Unit of Measurement*

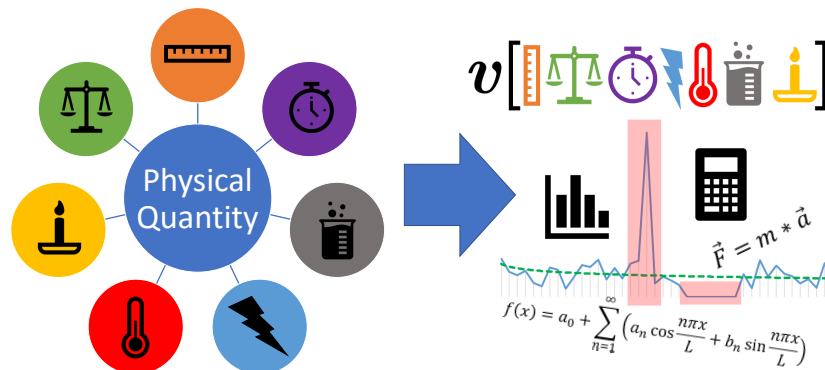


Fig. 4: PHYSICAL QUANTITY: An efficient and unit-safe specialization of the QUANTITY pattern for physical and mathematical calculations with the SI unit system.

5.1 Context

During the development of software that contains some physical or mathematical formulas and equations with values in a unit system (e.g., SI units). Software architects, designers, or developers have to decide on the concrete implementation of this.

5.2 Problem

How can a software application be designed to ensure the correctness of physical units in its calculations in a robust, easy to use and elegant way? How can numerical instabilities and floating-point inaccuracies in calculations be handled? The following forces describe this problem in more detail.

5.2.1 *Forces.* The following forces describe the requirements for a system which has to handle units in physical calculations:

(A) Dimensional Analysis: A system of physical quantities must support the analysis of the corresponding units. Especially the SI system should be used. In many operations, the units have to be tested if they fit together and have to be simplified according to the rules defined by the ISO Standard [ISO 2009]. Therefore, one of the most important force is to ensure the correct usage of the quantities based on their units.

(B) Arithmetic: PHYSICAL QUANTITY can be used in arithmetic calculations (e.g., addition, multiplication, etc.). These arithmetic operations must check if the units fit and return results in the semantically correct unit. Some operations are only possible with the same units (e.g., addition, subtraction), while others allow different units (e.g., multiplication). The following operations must be supported:

—*Addition and subtraction:* These are only valid with quantities with the same units. The result also has the same unit as the input quantities. The operators for these operations are + and −.

—*Multiplication and division:* These are allowed to be used with arbitrary units. The result is the correct combination of the input quantities. The operators for these operations are * and /.

(C) Comparison: Quantities have to be compared to each other, whether they are smaller, greater, or equal. Comparisons are only valid for quantities with the same unit. At least one ordering relation has to be defined (e.g., smaller than), but for convenience, more operations would come in handy:

- Less than: $<$, \leq
- Greater than: $>$, \geq
- Equality: $=$

A comparison should use absolute tolerances to mitigate rounding errors or numerical inaccuracies due to the nature of floating-point numbers. Values that only differ by a tiny tolerance value (e.g., $\varepsilon = 10^{-6}$) should be treated equal, and comparison has to ensure that the difference is greater than the defined tolerance.

(D) Data Range and Precision: In physical calculations, often very small or very big numbers are used. Therefore the underlying numerical data type should support that. Also, the precision (amount of significant digits) of the data type should be sufficiently high. A typical data range could cover -10^6 to 10^6 , and the precision could be six significant digits after the comma, but these values strongly depend on the application. In this example, the ε for comparison tolerances could also be set to a fitting value of 10^{-6} . Floating-point data types introduce many inaccuracies which have to be compensated by a robust numerical system of quantities [Reiser and Knuth 1975; Higham 2002; Wilkinson 1994; Goldberg 1991].

(E) Syntax: Syntax is a very important aspect in implementing PHYSICAL QUANTITY. Elegant and readable syntax leads to better understandability, more convenient usage, and better maintainability. Bad syntax leads to errors and frustration. A system of quantities should be easy to use, and the syntax should intuitively guide the developer to use it correctly (principle of the least astonishment, [Geoffrey 1987; Raymond 2004]) — or as Scott Myers put it: *“Make interfaces easy to use correctly and hard to use incorrectly”* [Myers 2004]. Regarding physical simulations also physicists, technicians, or mechanical engineers are taking part in the developing process and should understand the source code.

(F) Performance: During a calculation run of physical simulations, potentially millions of arithmetic operations could be performed, and equally many quantity objects could be created. Slow running operations on the quantities could lead to poor performance of the overall simulation. Therefore, often used operations should perform fast, and the design of physical quantities should support performance optimizations (e.g., allowing for compiler optimizations and caching/pipelining). In contrast to Martin Fowler’s opinion [Fowler 1996a], performance is an issue if you are performing simulations with frequent calculations. Especially operations which are used very often (e.g., $> 10^6$ times). Examples for this typically are creating quantities, performing arithmetic calculations (plus, minus, multiply, divide), dimensional-analysis (reducing, comparison), and formatting.

(G) Infinity and Not-A-Number: Infinity and NaN values are indicators for computational errors. The default behavior in many languages is to continue with the calculation, which can only result in subsequential errors. A robust system of units in physical application must handle such invalid values. It should be immediately detectable that an error happened by throwing an exception.

(H) Error Handling and Robustness: A robust system for handling errors must be used, especially when it comes to undefined or impossible operations on numbers (like $0/0$). However, throwing an exception is not always the best option (it slows down performance). In some situations, it could be useful to return a reasonably defined numeric value (e.g., $0^0 := 1$). Such definitions have to be justified because they could lead to wrong results or hide computational errors.

8 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

5.3 Solution

Introduce a class representing a quantity in the unit system which combines a numeric value and an efficient unit representation. Implement basic arithmetic and semantic rules to represent and compute arbitrary quantities.

Physical Quantity
~ value
~ units[7]
+ PhysicalQuantity(value, units[7])
+ operator +, -(q1, q1: PhysicalQuantity): PhysicalQuantity
+ operator *, /(q1, q1: PhysicalQuantity): PhysicalQuantity
+ operator <, >, ==(q1, q1: PhysicalQuantity): PhysicalQuantity

Fig. 5: PHYSICAL QUANTITY: An efficient implementation of the QUANTITY pattern for the SI system.

The PHYSICAL QUANTITY is an implementation of the QUANTITY pattern [Fowler 1996a] tailored towards the SI system of units. The key element is the efficient storage of units. To avoid unnecessary lookup and indirections, the exponents of the seven base dimensions are stored as an array of fixed size. Also, the basic arithmetic and comparison operators are already implemented to check the units.

5.3.1 *Static Structure.* Figure 5 shows the static structure of the PHYSICAL QUANTITY pattern. It consists of the following parts, which are described hereafter: the value, the units, the constructor, arithmetic, and comparison operators. For arithmetic rules we use the same syntax as in the definition of the SI units (Section 2.2): v represents the value, $[Q]$ represents the 7-tuple of units.

- value:** The value is the numerical value of the quantity. The data type double will be here the best option for most cases. Different data types could be used, which is discussed in the variants in section 5.5.5. Infinity and Not-A-Number are forbidden values (according to Force (G)).
- units[7]:** Here, the exponents of the respective base units are stored. Positive Numbers depict positive exponents (units in the numerator), negative numbers stand for negative exponent (unit in the denominator). Depending on the kind of equations, these exponents could have different ranges and are discrete or decimal. For most applications, a range of integers between -3 and $+3$ would be enough. A reasonable decision would be to use the smallest integer type available, which still can represent all needed exponents. The PHYSICAL QUANTITY pattern defines the units according to the SI standard in following order: [Meter, Kilogram, Second, Ampere, Kelvin, Mole, Candela].
- Constructor:** The constructor takes a numeric value and the respective unit exponents. With it, an arbitrary quantity can be created. To resolve Force (G), the constructor checks the value for Infinity and Not-A-Number and throws an exception if those values were given (Force (H)). However, to ensure that the developer uses the units-array correctly, it is advisable to supply FACTORYMETHODS, which creates the units correctly instead of using a general constructor. This relates to Force (E) and one of the candidate patterns: QUANTITY FACTORY (Section 7).
- Addition and Subtraction:** Add and subtract quantities of the same type. Before the operation, the unit-array is checked, and if they fit, the values are added and returned. The resulting value has the same units as the input parameters. Here are the formalized rules:

$$\begin{aligned}
 + & : v_1 [Q] + v_2 [Q] := v_1 + v_2 [Q] \\
 - & : v_1 [Q] - v_2 [Q] := v_1 - v_2 [Q]
 \end{aligned}$$

—**Multiplication and Division:** Here, the exponents have to be added or subtracted, and the values have to be multiplied or divided. The resulting value is returned as a new quantity with the combined units, following these rules:

$$* : v_1 [Q_1] * v_2 [Q_2] := v_1 * v_2 [Q_1 + Q_2]$$

$$/ : \frac{v_1 [Q_1]}{v_2 [Q_2]} := \frac{v_1}{v_2} [Q_1 - Q_2]$$

—**Comparison Operators:** These also are only valid on quantities with equal units. If the units are equal, the actual comparison of the values can be made. To mitigate floating-point rounding issues, the comparison is implemented with a small tolerance factor. They return a boolean result, according to the following rules:

$$< : v_1 [Q] < v_2 [Q] \Leftrightarrow v_1 < v_2 + \varepsilon$$

$$> : v_1 [Q] > v_2 [Q] \Leftrightarrow v_1 + \varepsilon > v_2$$

$$= : v_1 [Q] = v_2 [Q] \Leftrightarrow v_1 - \varepsilon < v_2 < v_1 + \varepsilon$$

5.4 Consequences

5.4.1 *Runtime Checks.* The PHYSICAL QUANTITY checks at run time if the units fit according to the rules. This is done in all arithmetic operations every time they are called. This solves multiple forces: (A), (B), and (C). Nevertheless, ensuring unit safety also introduces a performance overhead, which is contrary to Force (F) and allows no syntax checks, which contradicts Force (E). The compiler or an IDE cannot check via static code analysis if the units fit. Only via rigorous automated testing with unit test, integration tests, and system tests, this can be tested. There could be undetected exceptions due to wrong units during run time. In order to increase performance (Force (F)) and do syntax checking at compile time (Force (E)) the TYPESAFE QUANTITY INTERFACE pattern should be used.

5.4.2 *Unit Definitions.* There has to be a common standard in the application, what the base units are, and this common standard has to be strictly followed. This corresponds to Force (A). The PHYSICAL QUANTITY pattern defines in which order and how the units are stored in an application, but if the developers begin to mix up the units, the PHYSICAL QUANTITY cannot guarantee type-safety anymore. If the TYPESAFE QUANTITY INTERFACE is also used, the explicit quantity types prohibit wrong use via the type system, and the compiler could mitigate this problem.

5.4.3 *Performance.* While performance was one of the design goals (Force (F)), this pattern introduces an overhead compared to directly using double values. In its heap-variant, it needs at least one level of indirection, which has to be dereferenced (the numeric value is encapsulated inside a class), and the allocation and garbage collection also takes longer than just the allocation of a double-typed value on the stack. On the other side, if the pattern is implemented in its stack-variant, the copying of the value and the units-array takes longer than to copy a double-typed value or a reference. In both cases, the unit-checks need additional comparisons instructions before every arithmetic and comparison-operator, which eats up some performance. However, at least the unit-checks could be avoided with the TYPESAFE QUANTITY INTERFACE pattern (see Section 6).

5.4.4 *Fixed Unit Array.* The units are stored as a consecutive array of bytes, which is rather small and internally only needs one layer of indirection by lookup up the address of the array. Since the size is fixed and does not grow or shrink its memory allocation is very easy. Also, looking up elements in the array is easy, and it can be compared very efficiently by comparing the consecutive memory content.

5.4.5 *Floating Point Equality.* It is important to choose a precision value depending on the values which appear in the domain. All values which differ by less than this can be considered the same. Be aware that different domains could have different precision epsilons, and also sometimes, relative

precision is more useful than absolute precision. This is elaborated extensively in [Goldberg 1991]. In this pattern, an absolute precision of $\varepsilon = 10^{-6}$ is used. Sometimes a higher or lower precision is needed than in the standards cases. For a general framework, it would be good if this value is adjustable to the developer's needs. This consequence relates to Force (D).

5.4.6 Ambiguous Interfaces. PHYSICAL QUANTITY only supports dimensional quantity checks during run-time. If it is used as parameter type in method signatures, variable types, or interfaces, it is not clear which unit it represents. Methods that use PHYSICAL QUANTITY do not know its unit beforehand and have to test and adjust to it at run time. This is a liability which contradicts Force (E), but can be solved via the TYPESAFE QUANTITY INTERFACE pattern (see Section 6).

5.5 Variants

5.5.1 Store Only the Used Units. One way to optimize memory consumption would be to leave out units which may not be needed in the application (e.g. Candela, Mole). Although this limits the possibilities, this would be a reasonable variant, if the excess units are never needed. In this variant, the unit array would have lesser elements and therefore consume less memory and may even align better to the CPU caching mechanisms.

5.5.2 Units as Member Variables. Another variant would be to use seven member-variables to store the unit exponents. In such a way one indirection is saved, because the members are directly stored in the same memory block as the object itself. Depending on the types used and the size of the CPU caches this could be beneficial or harmful for the performance, and must be measured and reasoned upon if it is a wise decision to use it this way.

5.5.3 Value Type vs. Reference Type. Regarding the memory allocation, there are two variants of this pattern: Allocate the needed memory for the quantities on the heap or the stack. A Heap allocation has the advantage that it is available in the whole application, and just the reference has to be returned and used. This introduces an additional layer of indirection via the reference and additional overhead in programming languages that apply automatic garbage collection. The other variant (stack) is accessible more directly, but the values need to be copied if they should be used outside of the creating stack. Allocating and accessing data on the stack is generally faster than on the heap, but if these values are needed in other functions, they have to be copied, which then again takes time for copying. It strongly depends on the application and the implementation of the methods, which of these variants is faster. In our reference project VECTO, we decided to implement it as a class on the heap in C# because then we can use this as a parent class for all the explicit quantities we used (see TYPESAFE QUANTITY INTERFACE on page 6 for more information). While in C++, the developer can decide if the object is stored on the stack or the heap, in C#, this depends upon the object definition. If an object is declared with the `class` keyword it is a Reference Object and automatically declared on the heap. If it is declared as `struct` it is accounted as a Value Type and declared on the stack. This has huge performance implications since the stack is linearly growing, and the data is directly accessible, while the heap needs one or more levels of indirection and may get fragmented. Also, garbage collection has more work to do when the heap is used often.

5.5.4 Absolute vs. Relative Precision. The precision is dependent on the problem domain of the pattern and ideally would be adjustable dynamically. The same accounts for the way how precision is measured: Either absolute or relative precision. Relative precision is more useful in domains where the value ranges differ by huge amounts. This compensates for the huge differences, while an absolute precision would be way too precise for some values. This relates to the UNCERTAIN QUANTITY candidate pattern (Section 7) where uncertainty and tolerance are amongst the base traits of a quantity.

5.5.5 *Base Data Type.* The base data type hugely influences the precision of the calculations. Here we describe the different options regarding the base data type:

- Double / Float:* Well suited for physical units because they support a huge data range with high precision. For convoluted calculations, the type float may introduce to high inaccuracies and, therefore, double would be more appropriate. This variant is recommended by the authors and is also used in the example. If floating-point data types are not available (e.g., embedded devices without FPU), also, an integer data type could be chosen with a base unit which is fine-grained enough to support the needed precision (e.g., milliseconds). In such small base units, most of the occurring values may be whole integers, and therefore an integer type can be used.
- Integer:* Only suited when all used numbers in the application can be represented as integer numbers, or if floating-point numbers are not available on the platform. In such cases, the basic units should be chosen small enough so that the needed precision is achieved (e.g., milliseconds).
- Decimal / Big Decimal:* If a big decimal numeric type is available, it could also be chosen as a basic data type. This would have the benefit that arithmetic is more accurate, but the disadvantage that it is much slower.
- Fixed Point Types:* Well-fitting for money, but not for physical units. Physical units often have very different ranges of values (very big, very small), and fixed points data types do not support this very well.

5.6 Related Patterns

Quantity. The QUANTITY is the basic analysis pattern by Martin Fowler [Fowler 1997]. It describes the idea of combining a numerical value with the respective units.

Typesafe Quantity Interface. If the quantities should be checked for unit safety at compile time and also when the methods' signatures and interfaces should be made more clear and intuitive. This also mitigates the performance issues the PHYSICAL QUANTITY introduces due to permanent unit-checks.

Immutable Value. The PHYSICAL QUANTITY can be implemented as an IMMUTABLE VALUE [Buschma et al. 2007] because its arithmetic and other operations do not change. After creation, the PHYSICAL QUANTITY stays the same as long as it is alive.

Flyweight. Some numbers will be used very often in PHYSICAL QUANTITY (e.g., 0, 1, -1, 2, 0.5). These often used numbers could be implemented as Flyweights in order to save the application from creating and destroying objects for those commonly occurring values. This saves memory and avoids frequent memory allocation and garbage collection.

Factory Method. Since PHYSICAL QUANTITY has to be created in some type-safe and easy way, this could be done with FACTORY METHODS or more sophisticated creation patterns (like BUILDER, ABSTRACT FACTORY, PARSER).

12 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

6. PATTERN: TYPESAFE QUANTITY INTERFACE

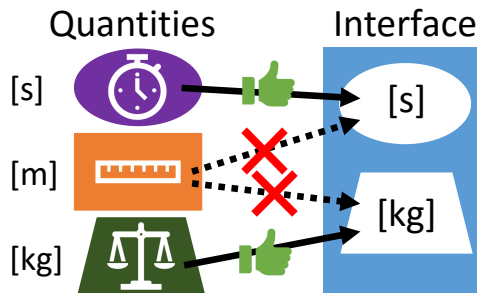


Fig. 6: TYPESAFE QUANTITY INTERFACE: Define explicit quantity data types which represent a specific unit in the unit system. Use these types everywhere to specify the required quantity-type.

6.1 Context

During the development of software containing some physical or mathematical formulas and equations with values in a unit system (e.g., SI units). Software architects, designers, or developers have to decide on the interfaces and types which will be used throughout the application.

6.2 Problem

Quantities should support dimensional analysis: type-checks based on their respective units. This should help to find errors in equations, formulas, and calculations as early as possible. The earlier an error in the equations can be detected, the better. While the PHYSICAL QUANTITY already supports units-checks at run time, it would be preferable to detect such errors already at compile-time or write-time. The problem with run time only is that errors are not detected until the respective code is executed. The problem with compile-time only is, that during calculations, arbitrary combinations of units could be created temporary (e.g., \sqrt{s} , or kg^2/m^4) which have no meaningful type representation, and are even reduced or canceled out in the result. It is not feasible to implement every possible combination of units only for satisfying the compiler. Nevertheless, if the types do not fit at compile-time, there should be a clear indicator of the error in the equation, which can be detected early. If a general combination of units occurs during an equation, there should be a possibility to cast it in a type-safe way to the correct explicit quantity type.

6.2.1 Forces

(A) Type Safety: A system of quantities should enforce that the units fit together in method signatures, interfaces, member types and variable types.

(B) Compile Time Checks: Static Code Analysis and compilers should detect type errors in units as syntax errors during writing the source code during writing-time and compile-time.

(C) Implicit Documentation: Method signatures should implicitly document the required parameter types to mitigate erroneous calls with wrong quantities. This helps programmers to understand the source code intuitively.

6.3 Solution

Implement explicit quantity types for every needed type of quantity in the system, together with all needed arithmetic operators. Use these explicit quantity types in method signatures and properties in order to enforce the correct type usage.

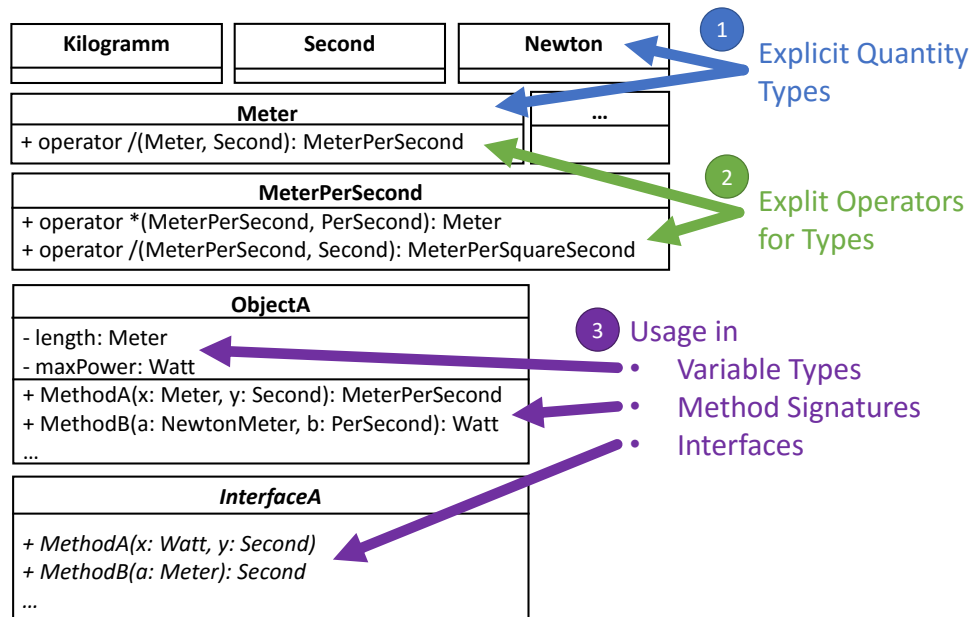


Fig. 7: TYPESAFE QUANTITY INTERFACE: Explicitly defined quantity types with implementation of operators and their usage in method signatures.

The explicit quantity types are implemented at compile time and therefore can also be checked by the compiler or some static code analysis tool at that time. If an equation or method signature does not fit the respective quantity type, the compiler immediately can check this and signal it to the user as syntax error.

6.3.1 Static Structure. The TYPESAFE QUANTITY INTERFACE pattern defines many classes that represent explicit quantities of the used unit system. Examples of such are Meter, Kilogram, Watt, Ampere, MeterPerSecond. This also can be seen in Figure 7. The static structure of this pattern consists of three parts:

- (1) **Explicit Quantity Types:** For every needed quantity, there should be an own type defined, which represents this. This ensures that the compiler can distinguish between them.
- (2) **Explicit Operators:** The quantity types should implement all needed operators and mathematical functions. This ensures that they can easily replace existing values in functions. This also makes it easier to use for the developers.
- (3) **Usage in Signatures and Variables Types:** Finally, everywhere a quantity is used, the explicit quantity types should be used. This includes normal method signatures, as well as interfaces, and even inside methods, the variable types should be quantities.

6.3.2 *Dynamics.* The dynamics in the TYPESAFE QUANTITY INTERFACE pattern are as follows:

- At writing-time, an IDE can resolve the type names and already hint the developer to the correct usage of the types.
- At compile-time, the compiler can check the code for syntax errors where types were misused.
- At run-time, the quantity types provide the same functionality as integrated data types with the added benefit of unit-safety. During run-time, this may introduce some performance overhead due to using objects instead of the integrated native data types. This is discussed in the Consequences (see Section 6.4).

6.3.3 *Example.* We illustrate the usage of the TYPESAFE QUANTITY INTERFACE on an example taken from a real project we are working on. The project is called VECTO [Hausberger et al. 2016] and has the purpose of calculating the CO₂ consumption of heavy-duty vehicles like trucks and buses. It consists of a component-based architecture that represents every part and piece of the power train as components, which are loosely connected by defined interfaces. In this example, we show only the last part: the combustion engine. The engine is connected to the gearbox and provides some functions to be used by it (e.g., return the maximum power, calculate the fuel consumption for the current torque and engine speed). These functions use explicit quantity types as types in their method signatures. Therefore, it is clear for the developer, which quantities it expects and which quantity it returns. Also, these methods can be communicated to a mechanical engineer and would intuitively be understandable.

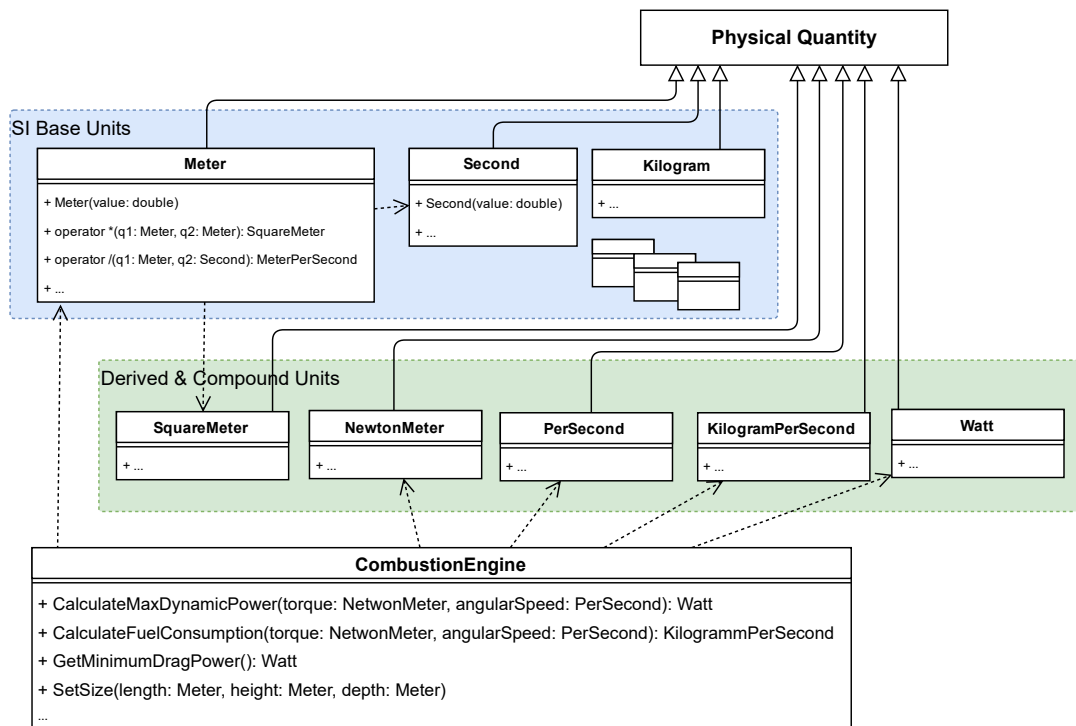


Fig. 8: TYPESAFE QUANTITIES IN VECTO: Explicit Quantity Types are derived from a PHYSICAL QUANTITY base class and used in component and method signatures.

6.4 Consequences

6.4.1 *Explicit Types.* In order to use this pattern, explicit quantity types have to be used wherever appropriate. This resolves Force (A). While this is easy to do in your code, external libraries may rely on built-in data types like double or integer, and the types have to be converted before these external libraries can be called. This additional conversion may be cumbersome, but could be encapsulated in helper-methods or the quantity types themselves via techniques which are supported by the framework (e.g., implementing the `IConvertible` interface in .NET).

6.4.2 *Compile Time Checks.* With explicit quantity types and their usage in the method signatures and interfaces, the compiler can check during compile time, if the types fit together, resolving Force (B). Even before that, the IDE can check this via static code analysis, and give correct code completion hints with the correct types.

6.4.3 *Increased Coupling between Types.* Using explicit quantity types in method signatures tightly couples them to the using objects. In this case, that is a beneficial feature, because we want to ensure correct usage of the types. This means that only specific combinations of types can be used together and, therefore, a higher coupling. Although this increases the coupling between the units, it makes them much easier to work with (resolving Force (C)). For example, if we divide Meter by Second the result should be of type MeterPerSecond, which is the only semantically correct result.

6.4.4 *Objects vs. Native Data Types.* : By using the quantity objects instead of native data types (like double or integer), the performance may degrade due to additional indirection needed and different storage (heap vs. stack). Also, for resolving the operators between types, additional indirection via the v-table may be needed. This can be mitigated by using value types (struct in C#) instead of reference types, but this inhibits inheritance.

6.4.5 *Null-Values.* : In comparison to native data types, object references could be null. This has to be adjusted for and involves additional checks if a quantity object is null or has an actual value. At least this can be immediately detected because most languages throw a null-reference exception when a null-object is referenced. However, handling null-values is a liability that has to be considered.

6.4.6 *Class Explosion for Arbitrary Units.* : In many applications, the amount of needed explicit quantity types will be at around 10 to 20 classes for the different quantities. However, if an application allows users to create arbitrary quantities in calculations, the amount of classes needed for all possible combinations explodes. In such cases, it is not feasible to implement every combination.

6.5 Related Patterns

Quantity. The QUANTITY is the basic analysis pattern by Martin Fowler [Fowler 1997]. It describes the idea of combining a numerical value with the respective units.

Physical Quantity. Is a specialization of the QUANTITY pattern which is tailored towards the SI system of units. It allows for arbitrary unit combinations and implements the basic arithmetic operators. A combination with TYPESAFE QUANTITY INTERFACE is thinkable in a way that for all commonly used and known quantities, the explicit quantity types are used, and for an additional unknown or intermediate result, the PHYSICAL QUANTITY is used. In such a case, a mechanism to convert between them has to be implemented, e.g., by inheritance of the explicit quantity class from a base class that resembles the PHYSICAL QUANTITY pattern.

Immutable Value. The explicit quantity types can be implemented as IMMUTABLE VALUE because the arithmetic and other operations do not change it [Buschmann et al. 2007].

16 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

Factory Method. Quantities have to be created in a type-safe and easy way. This could be done with FACTORY METHODS or more sophisticated creation patterns (like BUILDER, ABSTRACT FACTORY, PARSER) which are described in the Gang of Four Book [Gamma et al. 1995].

Typesafe Entity Container. The explicit quantity classes in the TYPESAFE QUANTITY INTERFACE are a variant of the idea of the TYPESAFE ENTITY CONTAINER [Schmoelzer et al. 2006]. There, the explicit interfaces make it possible to syntax check the source code at compile-time and support developers while writing the source code with code assistance. In this analogy, the PHYSICAL QUANTITY pattern corresponds to the Entity Container.

6.6 Variants

6.6.1 Common Base Class for Explicit Quantity Types. This variant involves the implementation of a common base class for all explicit quantity types. The abstract base class implements the most used functionality but cannot be instantiated directly. This avoids code duplication in the quantity types and was used in VECTO (see Example in Section 6.3.3).

6.6.2 No Base Class. This variant completely leaves out the base and only implements the specialized unit classes and their relations. In such a way, it is not possible to create arbitrary units combination and limit the unit system to the defined ones. Although this removes some flexibility, it would be feasible in situations where type-safety is of the highest importance. One disadvantage is that much code has to be duplicated in order to implement the needed functionality. This can be mitigated via the composition of functionality, but still, the calls to the composited classes have to be implemented.

6.6.3 Template-Baseclass. Implement the PHYSICAL QUANTITY base class with a template-parameter and derive the explicit quantity types from it. In such a way, basic addition, subtraction, and generic implementations for multiplication and division can already be provided without having to re-implement them over and over. This saves much coding effort, and therefore also prevents possible copy-and-paste bugs.

7. CANDIDATE PATTERNS

The following problem areas would enhance the PHYSICAL QUANTITY pattern and TYPESAFE QUANTITY INTERFACE pattern. The patterns are shown here still have to be fleshed out entirely and described in detail. We plan to complete this in future work.

Problem: Pattern	Description
Conversion: CONVERTIBLE QUANTITY	Solves the problem of converting between different quantities. This includes handling of prefixes (e.g., kilometer vs. meter), different names (ton vs. kg, hour vs. minute), and even different systems (e.g., miles vs. kilometer). Many of these conversions occur in the input and output layer of the application, while the application internally works with the defined base units. [Novak 1995]
Uncertainty: UNCERTAIN QUANTITY	Handles uncertainty due to measurement of values with appropriate propagation of uncertainty in arithmetic. This could include relative tolerances (e.g., $\pm 5\%$), as well as absolute tolerances (e.g., $\pm 3km/h$).
Validation: VALIDATABLE QUANTITY	Adds the capability to validating quantity values (e.g., minimum and maximum limits). These limits could be defined on the quantity type itself (e.g., prohibit infinity or not-a-number), or be defined on specific members and instances (e.g., the weight of a car must be between 0[kg] and 7500[kg]; or the resulting value of a calculation must not be greater than 300[Nm]).
Collections: QUANTITY COLLECTION	Pattern for memory-efficient storage and handling of quantity values in big lists. In big lists of the same quantity, the unit information can be stored once, while the values could be stored in built-in lists of the underlying numeric data type (e.g. List<double>)
Matrix Math: QUANTITY MATRIX	Expands the quantity type arithmetic to support matrix and vector operations (e.g., matrix multiplication, cross-product, dot-product, inversion, determinant, euclidean-distance, ...) and a variety of dimensions (e.g., 2D, 3D).
Creation: QUANTITY FACTORY	An elegant mechanism to create the quantities. The syntax should be straightforward to read and understand because this would be used all over the place in the whole application. In .NET this could be implemented via extension methods which attach directly to the native data types float, double and int.

8. KNOWN USES

8.1 Java

- JSR363**: In August 2016 the Java Community Process has finalized a specification request of a Java package for modeling and working with measurement values, quantities and their corresponding units [Shaikh et al. 2016].
- JScience**: Library for scientific applications which also includes the SI unit system and quantities [JScience 2011]
- UOM.SI: Units of Measurement**: A library of SI quantities and unit types base on BIPM standards [Keil 2017].

8.2 C++

- Boost.Units**: A library implementing dimensional analysis and [Schabel 2010].

18 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

—**TUOML: The Units of Measure Library**: Another library for C++ implementing units of measure [Nicewarner 2013].

8.3 mbeddr

Mbeddr is a programming language for embedded devices based on an extensible version of C. It supports physical units as part of the language [Voelter et al. 2012].

8.4 F#

In **F#**, quantities are also included in the language itself in a very generic implementation, which allows defining arbitrary identifiers as units [Microsoft 2016; Pemberton 2015]. The check is done at compile time only [Wlaschin 2012; Kennedy 2008].

Powerpack. F# has an official add-on called “Powerpack” which contains many predefined definitions for SI units and other unit systems [Microsoft 2014; 2012].

8.5 C#

—**NGenericDimensions** [Mafu 2013].

—**Units.Net**: A library for C# implementing parsing, arithmetics, and conversion of units and quantities [Larsen 2017].

—**VECTO - Vehicle Emission Calculation Tool**: EU Project for Calculating CO2 Emissions for Heavy-Duty Vehicles in Europe. Written in .NET (C#, VB) and implementing an optimized version of SI Units [Hausberger et al. 2016; Kies et al. 2013].

8.6 Python

In Python, it is difficult to implement a type-safe version of the `PHYSICAL QUANTITY` for method parameters, because they cannot be specified, but at least arithmetic, conversion and run-time checks for dimensional analysis is supported.

—**Unum**: A library supporting all seven base units for SI, quantifiers, arithmetic operators and also integrates well within numpy [MacLeod and Denis 2013].

—**quantities**: Library defining physical units and constants, conversion, and arithmetics. This library also supports uncertainty and error propagation [Dale 2011].

—**scimath**: This library also implements the `PHYSICAL QUANTITY` (especially the 7-tuple with the exponents). It allows conversions, arithmetic, and support a different unit system (SI-CGS, SI-MKS, Imperial Units). It also integrates nicely into numpy for matrix and vector operations [Enthougt 2011].

9. CONCLUSION

In this paper, the first parts for a pattern language for physical quantities were described. These patterns support units in physical and mathematical calculations and simulations. They provide a robust typing system that helps to avoid unit errors in the implementation of equations and formulas. They also implement often needed functionality like arithmetic operators. Additional patterns were shown which support converting quantities, efficiently store many quantities in lists, or working with uncertain values and tolerances. This pattern language and the contained patterns are most useful in physical simulations and calculations where a robust unit system is needed.

Proceedings of the 22nd European Conference on Pattern Languages of Programs

Acknowledgments

We want to thank our shepherd Andreas Rüping for his helpful comments and feedback on this paper. He helped us clear our clouded minds to see the structure of this pattern language more clearly. We also want to thank the EuroPLoP community and workshop participants at EuroPLoP 2017 for their helpful input.

REFERENCES

- BIPM. 2006. *The International System of Units (SI)* (8th ed.). 186 pages.
- Frank Buschmann, Kevlin Henney, and Douglas Schmidt. 2007. *Pattern-Oriented Software Architecture Volume 4: A Pattern Language for Distributed Computing*. 639 pages.
- C2. 2011. Canonical Form. (2011). <http://wiki.c2.com/?CanonicalForm>
- Darren Dale. 2011. Quantities. (2011). <http://pythonhosted.org/quantities/>
- Enthought. 2011. Scimath Documentation. (2011). <http://scimath.readthedocs.io/en/latest/>
- Martin Fowler. 1996a. *Analysis Patterns: Reusable Object Models*. Addison-Wesley Professional. 384 pages.
- Martin Fowler. 1996b. Money. (1996). <https://www.martinfowler.com/eaCatalog/money.html>
- Martin Fowler. 1997. Quantity. (1997).
- Erich Gamma, Richard Helm, Ralph E. Johnson, and John Vlissides. 1995. *Design Patterns - Elements of Reusable Object-Oriented Software* (1 ed.). Addison-Wesley, Boston, USA. 395 pages.
- James Geoffrey. 1987. *The Tao Of Programming*. (1987). <http://canonical.org/>
- David Goldberg. 1991. *What every computer scientist should know about floating-point arithmetic*. Vol. 23. 5–48 pages. DOI: <http://dx.doi.org/10.1145/103162.103163>
- Stefan Hausberger, Martin Rexeis, Raphael Luz, Christian Kreiner, Michael Krisper, Markus Quaritsch, Philipp Gretzl, and Helmut Eichlseder. 2016. VECTO tool development. (2016).
- Nicholas J Higham. 2002. *Accuracy and Stability of Numerical Algorithms*. 1—663 pages. DOI: <http://dx.doi.org/10.2307/2669725>
- ISO. 2009. ISO:80000-1:2009. (2009). <https://www.iso.org/obp/ui/#iso:std:iso:80000:-1:ed-1:v1:en>
- JScience. 2011. JScience. (2011). <http://jscience.org/>
- William Kahan. 1996. IEEE Standard 754 for Binary Floating-Point Arithmetic. *University of California, Berkeley* May 1995 (1996), 1–30. DOI: <http://dx.doi.org/10.1109/71.536937>
- William Kahan. 2001. Why do we need a floating-point arithmetic standard? (2001).
- Werner Keil. 2017. UOMSI. (2017). <http://uom.si>
- Andrew Kennedy. 2008. Units of Measure in F#. (2008). <https://blogs.msdn.microsoft.com/andrewkennedy/2008/08/29/units-of-measure-in-f-part-one-introducing-units/>
- Antonius Kies, Martin Rexeis, Gerard Silberholz, Raphael Luz, and Stefan Hausberger. 2013. *Options to consider future advanced fuel-saving technologies in the CO2 test procedure for HDV*. Technical Report. Forschungsgesellschaft für Verbrennungskraftmaschinen und Thermodynamik mbH. 108 pages.
- Andreas Gullberg Larsen. 2017. Units.Net. (2017). <https://github.com/anjdreas/UnitsNet>
- Chris MacLeod and Pierre Denis. 2013. Units in Python. (2013). <https://pypi.python.org/pypi/Unum>
- Josh Mafu. 2013. NGenericDimensions™. (2013). <http://ngenericdimensions.codeplex.com>
- Scott Meyers. 2004. The Most Important Design Guideline? In *IEEE Software*, Martin Fowler (Ed.). 14–16. <http://www.aristeia.com/Papers/IEEE>
- Microsoft. 2012. FSPowerPack.Community. (2012). <https://www.nuget.org/packages/FSPowerPack.Community>
- Microsoft. 2014. The Old F# "PowerPack". (2014). <http://fsharp.powerpack.codeplex.com/>
- Microsoft. 2016. Units of Measure (F#). (2016). <https://docs.microsoft.com/en-us/dotnet/articles/fsharp/language-reference/units-of-measure>
- Keith Nicewarner. 2013. The Units of Measure Library. (2013). <http://tuoml.sourceforge.net>
- NIST. 2008. *SP811 - Guide for the Use of the International System of Units (SI)*. Technical Report.
- NIST. 2009. Derived Units, Outside the SI. (2009). <http://physics.nist.gov/cuu/Units/outside.html>
- Gordon S. Novak. 1995. Conversion of units of measurement. *IEEE Transactions on Software Engineering* 21, 8 (1995), 651–661. DOI: <http://dx.doi.org/10.1109/32.403789>

20 • M. Krisper, J. Iber, T. Rauter and C. Kreiner

- Steve Pemberton. 2015. F# Units of Measure - A Worked Example. (2015). <http://stevenpemberton.net/blog/2015/03/11/FSharp-Units-Of-Measure/>
- Eric S. Raymond. 2004. *The art of Unix programming*. Addison-Wesley. 525 pages.
- John F. Reiser and Donald E. Knuth. 1975. Evading the drift in floating-point addition. *Inform. Process. Lett.* 3, 3 (1975), 84–87. DOI: [http://dx.doi.org/10.1016/0020-0190\(75\)90022-8](http://dx.doi.org/10.1016/0020-0190(75)90022-8)
- Matthias C. Schabel. 2010. Boost.Units 1.1.0. (2010). http://www.boost.org/doc/libs/1.55.0/doc/html/boost_units.html
- Gernot Schmoelzer, Christian Kreiner, Zsolt Kovacs, and Michael Thonhauser. 2006. Reflective, Model-Based Data Access with the Type-Safe Entity Container. In *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, Vol. 2. 87–92. DOI: <http://dx.doi.org/10.1109/COMPSAC.2006.155>
- Almas Shaikh, Chris Senior, Jean-Marie Dautelle, Karen Legrand, Leonardo Lima, Martin Desruisseaux, Mohamed Taman, Otavio Santana, Rajmahendra Hedge, and Werner Keil. 2016. JSR 363 Units of Measurement. (2016), 1–45.
- Markus Voelter, Daniel Ratiu, Bernhard Schaetz, and Bernd Kolb. 2012. mbeddr: an Extensible C-based Programming Language and IDE for Embedded Systems. (2012). <http://voelter.de/data/pub/wavefront-updatedSubmission2.pdf>
- James H. Wilkinson. 1994. *Rounding Errors in Algebraic Processes* (reprint ed.). Dover Publications. 161 pages. <https://www.amazon.com/Rounding-Errors-Algebraic-Processes-Wilkinson/dp/0486679993>
- Scott Wlaschin. 2012. Units of measure - Type safety for numerics. (2012). <https://fsharpforfunandprofit.com/posts/units-of-measure/>
- J.W. Yoder, F. Balaguer, and Ralph Johnson. 2000. From analysis to design of the observation pattern. (2000), 18. <http://www.joeyoder.com/Research/metadata/Observation/ObservationModel.pdf>

A.2 [P2] Use-Cases for Uncertainty Propagation in Distributed control Systems

- Authors: **Michael Krisper**, Johannes Iber, Jürgen Dobaj
- Year: 2018
- DOI: 10.1007/978-3-319-97925-0_30
- Bibliography-Reference: [183]
- Published at EuroSPT'18 conference proceedings (Springer, CCIS, volume 896).
- Presented at EuroSPT'18 conference in Bilbao, Spain.

Summary This paper shows some general use-cases for uncertainty propagation in dependable systems. It shows that knowing the uncertainty over multiple calculation steps (in a graph structure) allows for sophisticated methods like, e.g., approximate computing or device fingerprinting.


The paper discusses the following use-cases and how the addition of uncertainty could benefit them:

1. **Quality Evaluation based on Uncertainty:** The quality of a signal can be determined using the uncertainty and noise information.
2. **Predictive Maintenance:** The noise of a signal could be used as an indicator if a component needs to be maintained, recalibrated, or replaced.
3. **Sensor Fusion:** Uncertainty information could be used in sensor fusion to emphasize the sensor with higher precision.
4. **Approximate Computing:** Knowing the uncertainty of a value also implies possible precision. This can be used in approximate computing to only evaluate the significant figures and speed up the calculation.
5. **Fault Detection:** Sudden changes in the uncertainty could be used in the safety function to detect faults.
6. **Sensor Fingerprinting:** The combination of sensors and their particular uncertainty could be used to calculate a fingerprint of the system to prove that the hardware composition is the same as it is supposed to be.
7. **Graceful Degradation:** Increasing uncertainties could activate safety functions for degraded functionality, e.g., the maximum flying height of a drone or maximum speed of an autonomous car could be limited dynamically by the sensory uncertainty.

My Contribution This paper was completely written and conceived by me. The contribution of the other authors was in the form of discussions, feedback for improvements, and corrigenda.

Copyright ©2018 Springer International Publishing Switzerland. The version included in this dissertation is a permitted reprint of the published version from Proceedings of the European & Asian System, Software & Service Process Improvement & Innovation (EuroAsiaSPI), September 2018, which can be accessed via SpringerLink: https://link.springer.com/chapter/10.1007/978-3-319-97925-0_30

Use-Cases for Uncertainty Propagation in Distributed Control Systems

Michael Krisper^() , Johannes Iber , and Jürgen Dobaj 

Institute for Technical Informatics, Graz University of Technology,
Inffeldgasse 16/I, 8010 Graz, Austria
office@iti.tugraz.at
http://www.iti.tugraz.at

Abstract. This paper describes how data quality can be used to gain trust between components in distributed control systems by adding information about quality to data values. Especially numeric uncertainty is a helpful tool for making highly informed decisions. To illustrate the benefits and challenges, several use-cases are discussed in the context of industrial and automotive settings. The target audience are architects and developers of cyber-physical systems in industrial and automotive domains, researchers in such domains and software developers who are writing software for embedded or distributed control systems which also use uncertain sensor measurements.

Keywords: Quality · Uncertainty · Measurement
Error propagation · Tolerance · Predictive maintenance
Sensor-fusion · Approximate computing

1 Introduction

In the past, control systems were isolated and closed systems which have been under control of one manufacturer or closed and protected environments. This paradigm changed with the recent upcoming of ubiquitous and distributed devices, the Internet of Things (IoT), and Cyber-Physical Systems (CPS). Control systems now are often distributed and may be highly dependent on other systems which are developed by other companies. This leads to many exciting kinds of problems. In order to cope with many of those problem fields, standards were established to harmonize technical compatibility (e.g. communication, data formats, and protocols) as well as warranty, safety and contracting issues (safety and quality standards like ISO61508, ISO26262, ISO25012, AUTOSAR, ASPICE, Functional Safety, 6σ). While safety and quality standards cover the whole development and production process until delivery, the newest trend shifts the actual binding time of decisions far beyond delivery to the actual usage and runtime of a product [24, 26]. Amorim et al. describe in their papers [1, 2] how to do this by using contracts consisting of demands and

guarantees between components and evaluating them at runtime. In their contracts they depend on using attributes which can be evaluated during runtime. The challenge is now to evaluate information about the environment during runtime. In order to do this, the notion of data quality comes into play. The ISO 25012 [15] defines characteristics of data quality and in this paper, we especially look at accuracy, precision, consistency and credibility. We illustrate this in use-cases which use the measurement uncertainty of sensors as an attribute for data quality. These use-cases concern not only safety but also other dependability attributes like availability and reliability. The main idea is to evaluate the quality of a sensor via its uncertainties of measured values. This data quality attribute should be used and propagated over the whole signal path of a system in order to make informed decisions and to have more information about a systems' state available at runtime. If the sensor has small measurement errors and is not biased, the measured data has high quality and can be trusted. In such a way a consumer of a value can decide dynamically at runtime if the values are trustworthy and react based on changes of data quality e.g. when a sensor gets dirty or faulty.

We want to encourage and motivate developers and architects to be involved in this culture of quality based thinking in order to increase business value in changing environments and contexts. This corresponds to the main values and principles of the Software Process Improvement Manifesto (SPI Manifesto) [23], which is a guide for exchanging wisdom and experiences in all areas of software process improvement.

The remaining paper is structured as follows: Sect. 2 gives an overview over the related work and background. The main part of the paper focuses on the use-cases for uncertainties beginning from Sect. 3 to Sect. 9. The paper closes with a conclusion in Sect. 10.

2 Background and Related Work

Uncertainty in Measurements. According to the ISO GUM Standard “*Error is an idealized concept and errors cannot be known exactly.*” [17]. Measuring the reality is a task which always involves some kind of uncertainty. Measurement devices cannot be infinitely precise or measured objects may change, and it may even be impossible to measure everything exactly. In lack of infinitely precise measurements, we try to tackle uncertainty by equipping our sensors and values with tolerance ranges or limits. Such tolerances tell us how precise our values can be. The ISO GUM (JCGM 100:2008) standard and all additions define how to express uncertainty in measurements [17–21].

Uncertainty in Distributed Control Systems. Distributed control systems cover many areas like control systems, embedded systems, or cyber-physical systems (CPS) as they are called nowadays. They all involve some kind of uncertainty. Classic literature about this topic is e.g. “Distributed Systems” by Tannenbaum et al. [29], or “Distributed Systems Architecture” by Puder et al. [28]. According to Amorim et al. “CPS usually operate in uncertain environments and are

370 M. Krisper et al.

often safety-critical” [1]. This dangerous combination of uncertainty and safety-critical devices could lead to threatening situations causing harm, injuries and the death of humans as well as substantial damage of properties and financial losses. According to the HAZOP model, data errors in control systems are categorized as *Provision Errors*, *Timing Errors*, and *Value Errors* [4, 11, 12]. To handle these kinds of errors, contracts with requirement definitions can be used [3]. For example, in the ConSerts M model [1, 2], every component provides guarantees to the consuming component, while requesting demands to the serving components. In such a way it is possible to define safety contacts at design time which are evaluated at runtime and acted upon via safety mechanism through self-adaptive systems [14].

Uncertainty in Probabilistic Programming. Basically every measured value can be represented as random variable, which is done in probabilistic programming. Such random variables have continuous or discrete probability distributions which are used for inference, arithmetic and conditionals in a program. They have some generic mechanisms in common [25]: (i) *a probabilistic model*, (ii) *propagation rules*, and (iii) *inference techniques*. Andy Gordon wrote a survey about the current state of probabilistic programming [13]. Bornholt et al. published several papers about their implementation approach of such mechanisms [5–7]. Another aspect of this is approximate computing, which creates software with just enough precision as needed. Darulova et al. investigated many aspects of approximate computing and created a framework for compiling programs with uncertainties, to be faster and use less memory [8–10, 16].

3 Use-Case 1: Quality Evaluation Based on Uncertainty

“Researchers in computer systems either do not know about measurement bias or do not realize how severe it can be.” [27].

Using uncertainty for measured values allows for informed decisions, better evaluation of the environment and sophisticated safety arguments, which can contain lower and upper bounds for safety margins (Fig. 1).

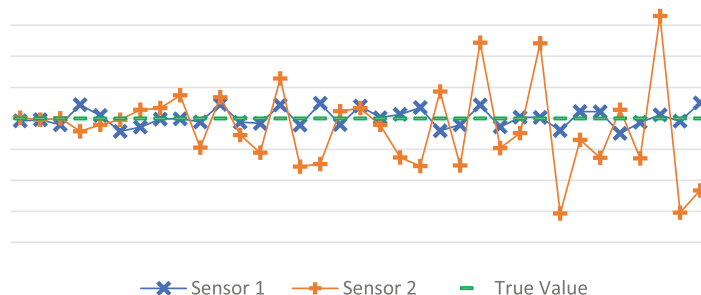


Fig. 1. Quality evaluation based on uncertainty in Sensor Data: which of the two sensors is more trustworthy?

Very often decisions and calculations are done with data directly coming from a sensor or other systems, and are trusted to be 100% correct. Contracts and design arguments protect us from getting biased or uncertain data, but do we really know for sure even during runtime? Mytkowicz et al. [27] showed that experimental measurements of computer systems regarding performance are always flawed by not using diverse environments and could potentially lead to wrong claims by not considering measurement bias. That is why they proclaim a call to action to consider measurement bias in computer systems. While this mainly applies to experimental measurements, it is also important for sensors, which exhibit even more uncertainty, which can't be reproduced as easily as in experimental environments. In safety, redundancy and diversification are key concepts for reducing failure rates and common cause failures. We propose to expand these concepts to allow better informed decisions about systems.

Wrong decisions potentially could lead to endangering human life, harm and injuries, but also enormous financial damage. For example, autonomous self-driving cars constantly monitor the environment and decide in adaptive control-loops which action is the most appropriate. Such decisions are guided by data from multiple sensors in order to drive safely and avoid accidents.

This use-case exhibits following forces or challenges:

- Sensor Data is always uncertain (as is every measurement). Therefore, it could be inaccurate, and without modeling these uncertainties this could lead to wrong assumptions and decisions.
- Exact tolerances are often unknown. Of course, you could assume the worst-case tolerances from the data sheet of a sensor, but oftentimes they are way overrated, and still one cannot guarantee during runtime that they are still satisfied.
- Decisions based on inaccurate or oversimplified data could lead to wrong results (injuries, fatalities, ...). Assuming a measured value is infinite precise is very dangerous and careless.

The goal in the context of this use-case is to make safe and informed decisions with the help of error-margins and safety assumptions to avoid and mitigate injuries and erroneous behavior. For this, we need systems which have mechanisms for defining uncertainties, for propagating them, and finally for decision-making with known guarantees and confidence.

4 Use-Case 2: Predictive Maintenance

The evaluation of uncertainty and measurement tolerances could potentially increase the prediction accuracy for predictive maintenance. By establishing degradation models which reason about how failure and quality of a component are related, a manufacturer can predict how long the product lifetime will be based on the current state of quality in the product evaluated during runtime. In such a way it would be possible to avoid unnecessary maintenance efforts, but

372 M. Krisper et al.

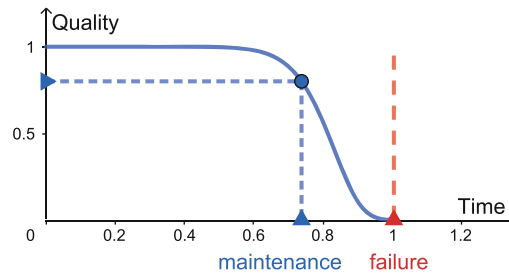


Fig. 2. Usage of quality information for predictive maintenance: as soon as the quality of the sensor is under a predefined limit, maintenance actions should be done before the component fails.

just replacing or reconditioning those parts where and when the attention really is needed (Fig. 2).

Think of the trivial example of motor oil. We have long periodic maintenance intervals because modern engines are working much cleaner as ever before. Nevertheless, motor oil is still replaced in regular intervals (either time span or driven distance) although it may not be necessary in many cases. A sensor which measures the viscosity or contamination of the oil could give feedback about its state and inform the driver when it is time to be changed.

The challenges in this context are:

- Sensors may get damaged, polluted and fail over time. The precision and quality of the produced data also decreases with such decay processes.
- Periodically scheduled maintenance may be inefficient, because parts could still be fully functional even after some time and would be replaced prematurely.
- The other side of periodic maintenance is when parts fail or decay earlier than the cycle has foreseen this may go unnoticed until the maintenance time. In such cases maintenance should have been earlier to ensure correct functionality of the components.

The goal in this use-case is to save costs for unnecessary maintenance while ensuring that all safety goals are met and the functionality of all components is ensured. Just doing maintenance when it is really needed has many advantages to the whole ecosystem of products. The predictive maintenance model and its evaluation during runtime also has another very beneficial side effect: By continuously monitoring the health state of the components we can detect early or unexpected failures during runtime. By knowing and monitoring the quality level (especially the sensor tolerances) one can predict the failure rate more accurately during runtime. By using a model for failure-rates based on the runtime tolerances we can predict the point in time when the sensor will fail. Based on that, maintenance should be planned.

5 Use-Case 3: Sensor Fusion

Use data quality, e.g. uncertainty, to combine several input values in order to get results with even higher quality, accuracy and less uncertainty. Also, use it to give the most accurate data more weight than the inaccurate (Fig. 3).

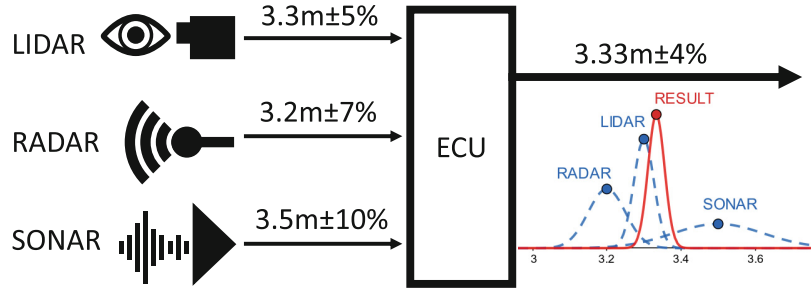


Fig. 3. Sensor fusion by combining the individual sensor value and their according probability distributions. The result has higher quality and lower uncertainty than every single sensor.

Sensor fusion is a huge area in control systems which is researched for many decades now but still makes huge advancements when it comes to new technologies and how to combine them. For more information about Sensor Fusion we propose the book by Klein: “Sensor and data fusion: a tool for information assessment and decision-making” [22]. In this use-case we concentrate on the fusion of data signals which are semantically similar (measuring the same information), e.g. distance, energy consumption, or signal strength. The only things which differ for similar measurements are the actual measured value, and the uncertainties exhibited by the sensor and the measurement. This makes it easier to combine the values by just combining their respective measurements seen as random variables with uncertainties. Equations for combining multiple independent input values with normal distributions (μ is the expected value or mean, σ is the standard deviation):

$$X_0, \dots, X_n \sim \mathcal{N}(\mu_i, \sigma_i^2) : \hat{X} = \frac{1}{n} \sum X_i \sim \mathcal{N}\left(\frac{1}{n} \sum \mu_i, \frac{1}{n^2} \sum \sigma_i^2\right)$$

It is noteworthy, that for the above equations we assumed independent random variables. This is not the case all the time, because the sensors try to measure the same “true” value, and are therefore correlated to each other. If the “true” value changed, all sensor values are expected to change accordingly – therefore, they are dependent and for sensor fusion also the covariances of the sensors should be considered. Another assumption here is that all sensor uncertainty is described using normal distributions for their measurements. When they have different distributions, this should be considered accordingly in order to maintain their probability properties.

The challenges in this context are:

374 M. Krisper et al.

- A single sensor may be inaccurate and its quality may change during usage over time.
- Computing simple averages amongst multiple sensor hides the uncertainty coming from the sensors and therefore gives a wrong image of certainty.
- Better and worse data are mixed together, which should be accounted accordingly.

The goal is to use data from multiple sensors to get more accurate information about the environment. The combined value should exhibit lower uncertainty than any single sensor value. Therefore, it is needed to evaluate the quality of the sensors during runtime and use this quality information for the sensor-fusion.

We propose to weight the sensors according to their uncertainty, in order to prefer more accurate sensors over the ones which are imprecise. This has two highly beneficial consequences:

1. **Environmental Adaption:** When one sensor is better for near distance measurements (e.g. in low-speed situations) and another is better at far distance measurements (e.g. high-speed situations), weighting them according to their precision would result in an automatic adaption to the current environment and always using the best source of data for a given situation.
2. **Failover:** In cases where a sensor fails completely, it can be completely overruled by the still functional working sensors, because its uncertainties would get very high and therefore its value would be weighted very low. This would result in failover situation where the system still continues to function, despite a sensor failing. Of course, this only is possible for systems which are designed to have redundant signal paths or even diverse sensors, in order to avoid common cause errors. Amorim et al. described an architecture which makes use of alternative data sources in case of failures [2].

6 Use-Case 4: Approximate Computing

Perform calculations only with the needed precision to increase performance and save memory.

Figure 4 shows the time needed for calculating pi with a variant of the Gregory-Leibniz Series ($\pi = 4 \sum_{k=0}^{\infty} \frac{(-1)^k}{2k+1}$). The figure shows that for increasing the accuracy (decreasing the error) much more time is needed. For example, if we need the value to have a precision of 10^{-3} , the algorithm only needs about 0.5 ms, but if we need a precision of 10^{-4} , we would have to let it run for 5 ms (10 times longer). Of course, in this case, there are much faster methods available, but it shows how beneficial approximate computing could be for algorithms which do not have a fast alternative. By aborting the calculation as soon as the needed precision is reached one can save much computing time [8].

In addition to performance also memory could be saved by using approximations. Many applications use double or float as data types for their floating-point variables, but only need precision of a few decimal places. These could be replaced by fixed point arithmetic which perform much faster while still supplying the needed precision [8–10, 16]. The challenges in this context are:

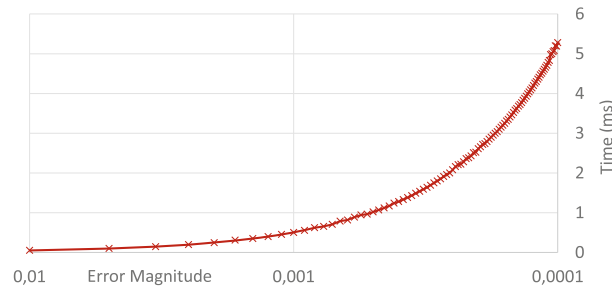


Fig. 4. Performance benchmarks for calculating π with the Gregory-Leibniz Series. To increase the accuracy of the solution, much more computing time is needed.

- Exact calculations often take unnecessary long time.
- Approximate calculations may be faster, but you need to balance the precision to your needs.

For this use-case the goal is to speed up calculations and save memory by using approximations which only use as much preciseness as is needed. In order to do that, uncertainty information comes in handy because it could be applied in two ways: Firstly, the maximum precision depends upon the precision of the input values, respectively the input sensors. It does not make sense to apply more exact algorithms when the uncertainty of the input data is already very high. Secondly, the needed precision in calculations depends upon the ultimately required precision of the output value. For example, it would be futile to numerically optimize some algorithms to the 10^{-6} decimal place, while the calculated output value is then rounded to whole integer numbers.

7 Use-Case 5: Fault Detection

Use quality information (e.g. uncertainty or standard deviation), to detect additional faults in components which would go unnoticed otherwise.

Using quality information like accuracy or uncertainty gives the possibility to define additional checks for fault detection. Thresholds on the *quality* of a signal can be defined in addition to the range-checks which are defined at design time according to the data-sheet or interface description of a component (e.g. the HSI: Hardware-Software-Interface-Specification) (Fig. 5).

The challenges are:

- Sensor Quality (e.g. tolerances) may change over time.
- Safety functions rely on good quality information to e.g. apply boundary or threshold checking.
- Tolerances coming from data sheets may be exaggerated and represent maximum values, which leads to very conservative assumptions.

The goal of this use-case is to have tolerance information available during runtime in order to be used for safety functionality and to detect faulty and

376 M. Krisper et al.

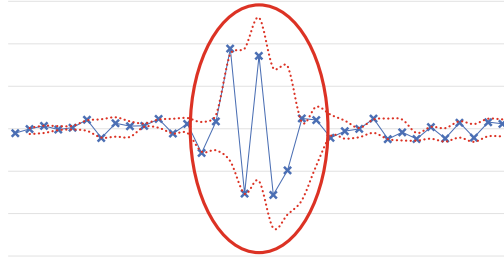


Fig. 5. Detect faults by evaluating the uncertainty of a value series.

maybe even harmful situations. When a system would know the tolerances and uncertainties of the used signals and sensors at runtime, it could easily detect when something goes out of bounds, or when tolerances of a value suddenly increase without any reason. This demands that the sensor quality is measured periodically in order to have recent information available to guarantee the liveness and correctness of the error margins.

8 Use-Case 6: Fingerprinting

Use quality information as fingerprints to identify individual components (Fig. 6).

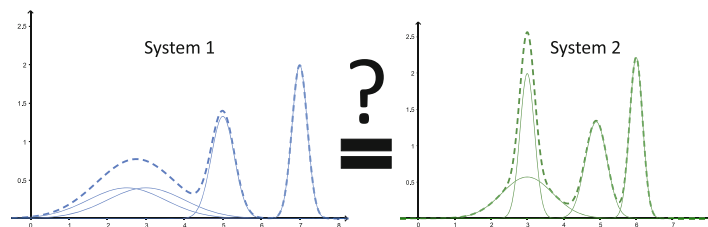


Fig. 6. Identify systems based on their individual calibration or uncertainty profile.

During production the uncertainties of sensors are measured and their calibration is set. This configuration of calibration data is a set unique to each system. By utilizing the initial calibration and uncertainty information one can calculate a unique fingerprint for the identification of a system and use this as an identification later on. The challenges of such a use case are:

- Devices and Sensors cannot be trusted a priori.
- Devices and Sensors in combination have a pretty unique configuration of calibration data, uncertainties and tolerances.
- Authentication mechanism need additional functionality (TPM, certificates, identities, key-exchange) which may be too expensive.

The goal is to identify sensors and devices according to their fingerprints without having to implement additional security features or hardware. This can be done by utilizing the uncertainty and calibration data of the systems' sensors during production and storing this profile information as fingerprint. During runtime measure the system again and compare to the stored fingerprint. If the profile is mostly the same, this is a strong indicator that the measured system actually is the same.

9 Use-Case 7: Graceful Degradation

Degrade functionality of a system based on the quality of the sensors and sophisticated safety assumptions (Fig. 7).

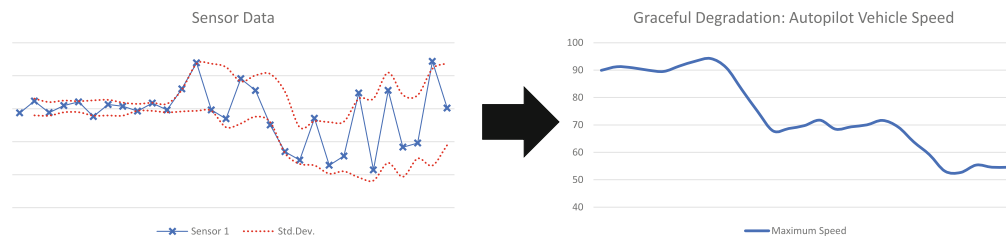


Fig. 7. Depending on the quality of the sensor input data, an autopilot can regulate the maximum speed for a degraded state.

Graceful degradation is a technique which is applied as safety measure if a system's functionality must be ensured but cannot be fully supplied. There can be two reasons why a degraded system is valuable: Firstly, when the system has to go into a safe-state, this oftentimes cannot be done immediately, but has to degrade gracefully over time, so that the driver or user can accustom to the new situation. Secondly, it is often preferred to have at least a degraded functionality than no functionality at all. Amorim et al. wrote about graceful degradation and how it can be applied to situations when the contracts are not fulfilled at runtime [2]. They depicted the situation where a sensor fails to operate and their solution was to search for other data sources which can be used despite the possibility that they may be more inaccurate. If the other inputs can't provide the needed ASIL level, the car should still be controllable, but in degraded mode in order to minimize possible hazards (the maximum speed is reduced).

Despite graceful degradation, also graceful improvement would be possible: if the uncertainty and data quality gets better, a system could adaptively increase the functionality. If the constraints are still met due to better sensors with smaller measurement tolerances, the maximum speed could even be increased while maintaining the same safety level.

10 Conclusion

In this paper we showed seven use-cases where the use of uncertainty as indicator for data quality is very beneficial for the dependability of a system. Trust in the sensors, the data, and the whole signal path can be increased by evaluating data quality of the numerical values. Especially using numerical uncertainty is helpful in making highly informed decisions which could potentially save lives. In the future, we plan to investigate each use-case in detail and find appropriate techniques and integration possibilities for existing systems in real life projects and scenarios. In the spirit of the SPI manifesto [23] we want to motivate and encourage manufacturers, developers and software as well as system architects to apply uncertainty and quality considerations in their systems to change their daily business for the better.

References

1. Amorim, T., et al.: Runtime safety assurance for adaptive cyber- physical systems: ConSerts M and ontology-based runtime reconfiguration applied to an automotive case study. In: Solutions for Cyber-Physical Systems Ubiquity, pp. 137–168. IGI Global (2018). <https://doi.org/10.4018/978-1-5225-2845-6>
2. Amorim, T., Ruiz, A., Dropmann, C., Schneider, D.: Multidirectional modular conditional safety certificates. In: Koornneef, F., van Gulijk, C. (eds.) SAFECOMP 2015. LNCS, vol. 9338, pp. 357–368. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24249-1_31
3. Beugnard, A., Jézéquel, J.M., Plouzeau, N.: Contract aware components, 10 years after. Electron. Proc. Theor. Comput. Sci. **37**, 1–11 (2010). <https://doi.org/10.4204/EPTCS.37.1>
4. Bondavalli, A., Simoncini, L.: Failure classification with respect to detection. In: Proceedings of the Second IEEE Workshop on Future Trends of Distributed Computing Systems, pp. 47–53, September 1990. <https://doi.org/10.1109/FTDCS.1990.138293>
5. Bornholt, J., Meng, N., Mytkowicz, T., McKinley, K.S.: Programming the internet of uncertain things, pp. 1–7 (2015)
6. Bornholt, J., Mytkowicz, T., McKinley, K.S.: Uncertain<T>: a first-order type for uncertain data, p. 21 (2013)
7. Bornholt, J., Mytkowicz, T., McKinley, K.S.: Uncertain<T>: a first-order type for uncertain data, pp. 51–66. ACM Press (2014). <https://doi.org/10.1145/2541940.2541958>
8. Darulova, E.: Programming with numerical uncertainties, p. 172 (2014)
9. Darulova, E., Kuncak, V.: Sound compilation of reals, pp. 235–248. ACM Press (2014). <https://doi.org/10.1145/2535838.2535874>
10. Darulova, E., Kuncak, V., Majumdar, R., Saha, I.: Synthesis of fixed-point programs, pp. 1–10. IEEE, September 2013. <https://doi.org/10.1109/EMSOFT.2013.6658600>
11. Ezhilchelvan, P.D., Shrivastava, S.K., Elphick, M.J.: A characterisation of faults in systems. University of Newcastle upon Tyne, Computing Laboratory (1985)
12. Fenelon, P., Hebborn, B.: Applying HAZOP to software engineering models. In: Risk Management and Critical Protective Systems: Proceedings of SARSS 1994 (1994)

13. Gordon, A.D., Henzinger, T.A., Nori, A.V., Rajamani, S.K.: Probabilistic programming. In: Proceedings of the on Future of Software Engineering, FOSE 2014, pp. 167–181. ACM, New York (2014). <https://doi.org/10.1145/2593882.2593900>
14. Iber, J., Rauter, T., Kreiner, C.: A self-adaptive software system for increasing the reliability and security of cyber-physical systems. In: Solutions for Cyber-Physical Systems Ubiquity, pp. 223–246 (2018). <https://doi.org/10.4018/978-1-5225-2845-6.ch009>
15. ISO, IEC: ISO/IEC 25012:2008 data quality model (2008)
16. Izycheva, A., Darulova, E.: On sound relative error bounds for floating-point arithmetic, pp. 15–22. IEEE, October 2017. <https://doi.org/10.23919/FMCAD.2017.8102236>
17. JCGM: JCGM 100:2008 evaluation of measurement data Guide to the expression of uncertainty in measurement, September 2008
18. JGCM: JCGM-WG1-SC1-N10 guide to the expression of uncertainty in measurement (GUM) - supplement 1: numerical methods for the propagation of distributions (2004)
19. JGCM: JCGM 104:2009 evaluation of measurement data - an introduction to the “guide to the expression of uncertainty in measurement” (2009)
20. JGCM: JCGM 102:2011 evaluation of measurement data - supplement 2 to the “guide to the expression of the uncertainty in measurement” - extension to any number of output quantities (2011)
21. JGCM: JCGM 106:2012 evaluation of measurement data - the role of measurement uncertainty in conformity assessment. Chem. Int. - Newsmagazine IUPAC **35**(2) (2013). <https://doi.org/10.1515/ci.2013.35.2.22>
22. Klein, L.A.: Sensor and data fusion: a tool for information assessment and decision making. SPIE Press, Bellingham (2004)
23. Korsaa, M., et al.: The SPI manifesto and the ECQA SPI manager certification scheme. J. Softw.: Evol. Process **24**(5), 525–540 (2012). <https://doi.org/10.1002/smr.502>
24. Kreiner, C.: A binding time guide to creational patterns, pp. 1–10. ACM Press (2015). <https://doi.org/10.1145/2739011.2739025>
25. Krisper, M., Iber, J., Dobaj, J., Kreiner, C.: Uncertain values, error-propagation, and decision confidence, p. 5. ACM, Irsee (2018, unpublished)
26. Krisper, M., Kreiner, C.: Describing binding time in software design patterns, pp. 1–15. ACM Press (2016). <https://doi.org/10.1145/3011784.3011811>
27. Mytkowicz, T., Diwan, A., Hauswirth, M., Sweeney, P.F.: Producing wrong data without doing anything obviously wrong! p. 12 (2009)
28. Puder, A., Römer, K., Pilhofer, F.: Distributed Systems Architecture: A Middleware Approach. Elsevier, Amsterdam/Boston (2006)
29. Tanenbaum, A.S., van Steen, M.: Distributed Systems: Principles and Paradigms, 2nd edn. Pearson Education, Harlow/Essex (2014)

A.3 [P3] Patterns for Implementing Uncertainty Propagation

- Authors: **Michael Krisper**, Johannes Iber, Christian Kreiner
- Year: 2018
- DOI: 10.1145/3282308.3282323
- Bibliography-Reference: [167]
- Presented and discussed at EuroPLOP'18 conference
- Published in EuroPLOP'18 conference proceedings (ACM, ICPS)

Summary In this paper, patterns for calculating uncertainty propagation were described. It uses the notion that every value also models its uncertainty in some way. Especially the representation via descriptive statistics using the normal distribution and standard deviation was elaborated. Nevertheless, different approaches were also discussed, e.g., describing it via a probability density function. It is also described how the uncertainty components can be propagated, e.g., with Gaussian error propagation or using Monte-Carlo simulation. To evaluate some aspects, I implemented a framework for Gaussian uncertainty propagation, which implements automatic differentiation to calculate the error components according to the Gaussian law of uncertainty propagation.

It contains the following design patterns:

- **Uncertain Number:** Define an object representing a numeric value with uncertainty. To do that, it has to store the value, the inherent uncertainty, and its dependent variables, as well as the propagation strategy. It also has to override the arithmetic operators and apply the propagation strategy on each arithmetic operation.
- **Propagation Strategy:** The propagation strategy is how the uncertainties get propagated over the arithmetic operations. It could implement, e.g., Gaussian error propagation, Monte-Carlo simulation, or other strategies. It could use the entire hierarchy of dependent variables or just depend on a single level of propagation. This has consequences on the precision and performance of the calculations.

My Contribution This paper was completely written and conceived by me. The contribution of the other authors was in the form of discussions, feedback for improvements, and corrigenda. Not mentioned as authors but also influencing the paper were the shepherd Veli-Pekka Eloranta and the workshop participants who gave feedback during the conference.

Copyright The version included in this dissertation is a permitted reprint of the definitive version, which can be accessed via the ACM Digital Library here:
<https://dl.acm.org/doi/10.1145/3282308.3282323>

Patterns for Implementing Uncertainty Propagation

Michael Krisper

Institute for Technical Informatics
Graz University of Technology
Graz, Austria
michael.krisper@tugraz.at

Jürgen Dobaj

Institute for Technical Informatics
Graz University of Technology
Graz, Austria
juergen.dobaj@tugraz.at

Johannes Iber

Institute for Technical Informatics
Graz University of Technology
Graz, Austria
johannes.iber@tugraz.at

Christian Kreiner

Institute for Technical Informatics
Graz University of Technology
Graz, Austria
christian.kreiner@tugraz.at

ABSTRACT

In this paper, the design patterns UNCERTAIN NUMBER and PROPAGATION STRATEGY are presented. They are useful for storing uncertainties of values and propagating them throughout calculations in an application. UNCERTAIN NUMBER represents a numerical value and its respective uncertainty. PROPAGATION STRATEGY represents the propagation method, to correctly propagate the uncertainty throughout an application. This is done according to the Law of Propagation of Uncertainty as defined in the Guide to Expression of Uncertainty in Measurements. This paper addresses software architects, designers and developers having to work with uncertain data e.g. coming from sensors or other measurements.

CCS CONCEPTS

• **Mathematics of computing** → **Probability and statistics**; *Probabilistic inference problems*; *Probabilistic reasoning algorithms*;
• **Computing methodologies** → **Uncertainty quantification**; *Probabilistic reasoning*; *Uncertainty quantification*; • **Software and its engineering** → **Design patterns**; *Software design engineering*;

KEYWORDS

uncertainty, tolerance, probability, error-propagation, confidence, inference

ACM Reference Format:

Michael Krisper, Johannes Iber, Jürgen Dobaj, and Christian Kreiner. 2018. Patterns for Implementing Uncertainty Propagation. In *23rd European Conference on Pattern Languages of Programs (EuroPLoP '18)*, July 4–8, 2018, Irsee, Germany. ACM, New York, NY, USA, 6 pages. <https://doi.org/10.1145/3282308.3282323>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EuroPLoP '18, July 4–8, 2018, Irsee, Germany

© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-6387-7/18/07... \$15.00

<https://doi.org/10.1145/3282308.3282323>

1 INTRODUCTION

This paper describes design patterns about the handling and propagation of numerical uncertainty. This is important for all systems which have to handle uncertain data e.g., data coming from experimental measurements or sensors in control systems and sensor networks.

Every measurement exhibits some degree of uncertainty because many factors influence it, may it be a systematic error or random error, caused by a multitude of reasons, and, therefore, can never be perfectly accurate. This is especially interesting for control systems in industrial settings because they rely on sensors to measure the environment and have become more and more ubiquitous in our world. Every modern car, smartphone, smart production system, or smart home uses a variety of sensors to detect changes in the environment and react to that accordingly. Therefore it is essential to get the data correctly and also incorporate the uncertainty or errors of the measurements. The notion of uncertainty is not limited to control systems and can be applied to every system which has to work with uncertain input values.

The target audience is software architects and developers working in the field of control systems, embedded systems, sensor networks, or physical simulation software. This includes all kinds of applications using uncertain input data (e.g., industrial, automotive, embedded systems, IoT, sensor networks).

1.1 Background

The *Guide to the Expression of Uncertainty in Measurement (Guide)* [6] describes the defacto standard way of expressing and propagating uncertainty throughout calculations. It uses the so-called *Law of Propagation of Uncertainties (LPU)*, which is described in more detail in section 2. In the past decade, several implementations and patterns emerged to reflect the described method of the Guide, and in this paper, we take another more modernized approach to describe the design patterns dealing with the propagation of uncertainty.

Measurements cannot be perfectly accurate; they always involve some uncertainty that comes from many different sources and in different forms. Sources for uncertainty are e.g., environmental factors, hardware limitations of sensors, sampling and transformation errors, inaccurate measurements, representation inaccuracies (e.g., floating-point values), or even wrong assumptions about the influence factors.

Uncertainty can be lowered, limited, or bounded using appropriate sensors, well-suited noise filters, statistical methods, and mathematical techniques. While high-quality sensors and statistical methods can give some guarantees of tolerances, limits, or confidence-levels, a perfectly accurate measurement is not possible - the techniques only promise that the real underlying error is smaller than a given tolerance.

One thing that is often forgotten is the fact that sensors degrade over time (as all hardware does), and their errors and uncertainty could also change over time. Therefore, static assumptions taken at design time may not be valid for the whole lifetime of a product. This makes it essential to evaluate the uncertainty at runtime and take appropriate measures to keep it in acceptable limits (e.g., periodic re-calibration of the sensor, predictive maintenance, replacement).

1.2 Related Work

We used several other projects and patterns as inspiration for extracting the patterns described in this paper. One of the biggest influences was, of course, the GUM TREE design pattern by Blair D. Hall [5, 9], which is described in great detail and several publications over the last decade. Also, many of the classical design patterns like COMPOSITE, STRATEGY, INTERPRETER/SYNTAX TREE [4] where quite useful to get ideas how to implement uncertainty propagation. We also took inspiration from actual implementations like the python library *uncertainties* by Eric-O. Lebigot [7], and a project from Luca Mari implementing uncertainty propagation using automatic differentiation in Java [8]. For learning and experiencing the implementation issues firsthand, we also implemented a system for uncertainty propagation in C#.

1.3 Structure

This paper is structured as follows: Section 2 summarizes the mathematical methods described in the Guide to understand how the Law of Propagation of Uncertainties works.

Afterwards the following three design patterns are described:

- **UNCERTAIN NUMBER:** A pattern for representing numerical values together with their respective uncertainties.
- **PROPAGATION STRATEGY:** A pattern for propagating uncertainties correctly throughout simple calculations.

For the descriptions of the patterns, we use the canonical pattern form, consisting of the sections context, problem, forces, solutions, consequences, and known uses [3].

2 PROPAGATION OF UNCERTAINTIES

Every measurement exhibits some degree of uncertainty, which is generally called measurement error. This measurement error has a multitude of reasons and can be decreased by using more accurate measurement devices, repeating the measurement very often, or using mathematical filter techniques that can filter out the errors to some degree. Nevertheless, the resulting values still contain some errors and uncertainties, and therefore, it has to be modeled somehow in order to create robust and safe systems. Measurement error can be modelled as follows:

$$Y = y - E \quad (1)$$

The value Y is the *measurand*, which is the true value we want to measure. It differs from the measured value of y by a difference of E , which represents the error of the measurement. Since the error E is different with every measurement, we only can model it as a probability distribution, telling us the probable behavior and distribution of the error values over many measurements. Since the measurement errors are very often symmetrical around the true value - which means that it is equally probable to underestimate and overestimate the true value - the normal distribution is a reasonable estimate of how the error will behave. However, this is not always the case. Nevertheless, even in other cases, a correspondent normally distributed replacement can be found and used in the model.

The probability density function of the normal distribution is:

$$\mathcal{N}(\mu, \sigma^2) = f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad (2)$$

where μ (the mean) and σ (the standard deviation) are the two descriptive parameters fully defining the normal distribution. They can be calculated from a sample population of values as follows:

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i \quad (3)$$

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N x_i - \bar{x}} \quad (4)$$

Here the formula of σ for the "sample standard deviation" is shown, which is the corrected or unbiased version of the standard deviation. It is useful when applied to just a fraction or sample of the whole population. Since the setting for this paper is measurement data coming from sensors, we can never get hold of the whole population, because this would mean to measure forever. Therefore this is the appropriate version to use.

When multiple measured values x_1, x_2, \dots, x_n are combined in a calculation, this can be modelled as function which looks like the following:

$$y = f(x_1, x_2, \dots, x_n) \quad (5)$$

This is called a measurement equation, measurement model, or transfer function. What this means is, that the function f incorporates all input variables x_1, x_2, \dots, x_n and calculates the output value y . Since all the input values are estimates of the true values (which means they have uncertainty), also the resulting output value is just an estimate.

In practice, such a holistic transfer function is very complicated and difficult to find. Usually, this is done by defining smaller and easier functions and combining them. This can be represented as follows:

$$y = f_1(f_2(\Lambda_2), \dots, f_k(\Lambda_k)) \quad (6)$$

Here Λ_i stands for the input values of the corresponding function f_i (also called the influence set). The influence sets of the sub-functions can be disjoint, but do not have to be; actually, it is very common that they overlap. This means that some of the sub-function use the same input variables. The decomposition can be

repeated for each of the sub-functions until we reach the elementary variables, which cannot be decomposed any further.

2.1 The Law of Propagation of Uncertainty

The LPU defines how the error propagates mathematically through calculations. The most generic formula for error propagation is as follows:

$$u(y) = \sqrt{\sum_{i=1}^n \sum_{j=1}^n u_i(y)r(x_i, x_j)u_j(y)} \quad (7)$$

where the component equation for u_i is the following:

$$u_i(y) = \frac{\partial f}{\partial x_i} u(x_i) \quad (8)$$

and $r(x_i, x_j)$ is the correlation coefficient between x_i and x_j which lies between $[-1, 1]$. Variables correlate 100% with themselves, therefore $r(x_i, x_i) = 1$. If variables don't correlate, their coefficient is 0.

If we completely neglect the correlations (if we assume that all variables are independent) the equation becomes:

$$u(y) = \sqrt{\sum_{i=1}^n \left(\frac{\partial f}{\partial x_i}\right)^2 u(x_i)^2} \quad (9)$$

For respective formulas for function decomposition is as follows:

$$u(y) = \sqrt{\sum_{i=1}^n u_j(x_i)^2} \quad (10)$$

$$u_j(x_i) = \sum_{x_k \in \Lambda_i} \frac{\partial f_i}{\partial x_k} u_j(x_k) \quad (11)$$

which must be evaluated for all variables in the influence set. In the case when x_i is elementary, the defined value the uncertainty is returned; otherwise, the decomposition continues recursively.

3 PATTERN: UNCERTAIN NUMBER

Attach uncertainty information to your numerical values and add functionality for propagating it through arithmetic operations.

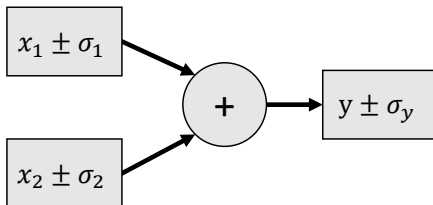


Figure 1: The UNCERTAIN NUMBER Pattern stores and propagates uncertainties through arithmetic operations.

3.1 Context

Applications that have to handle data with uncertainty e.g., using data that comes from sensors or measurements.

3.2 Problem

How can the uncertainty of input data be considered without having to consider it all time?

3.2.1 Forces.

- Standard numerical data types (e.g., int, double) have no means of storing uncertainty.
- The default arithmetic operators (+, -, *, /) for numerical data types do not incorporate or propagate uncertainties.
- Without considering uncertainties, developers and users of systems do not know how precise the results of calculations are. This could lead to dangerous situations if used in critical systems.
- Uncertainty propagation only works if used throughout all arithmetic operations.
- The mathematics of the LPU can be quite intimidating for developers, and could also be a source for errors if implemented wrong (automatic differentiation, syntax trees, function decomposition).
- Simple implementations often neglect correlations, which could lead to entirely wrong estimations of the uncertainties.
- When performance is an issue (e.g., embedded systems), the overhead of propagating uncertainties should be minimal, or at least configurable (e.g., eager or late calculation).
- Systems are often limited in the amount of available memory. Thus, memory consumption needs to be carefully thought out. The amount of used memory and the amount of available information should be balanced.
- Legacy systems often cannot be changed all at once. Therefore stepwise incorporation of UNCERTAIN NUMBER into legacy code should be possible.
- The usage of such a system for uncertainty propagation should be intuitive and easy to understand.

3.3 Solution

UncertainNumber
+Value : double
+Uncertainty : double
+Dependents : UncertainNumber[]
-propagation : PropagationStrategy
+UncertainNumber(v : double, u : double)
+UncertainNumber(p : PropagationStrategy)
+Uncertainty(u : UncertainNumber) : double
+operator+(u : UncertainNumber) : UncertainNumber
+operator-(u : UncertainNumber) : UncertainNumber
+operator*(u : UncertainNumber) : UncertainNumber
+operator/(u : UncertainNumber) : UncertainNumber

Figure 2: The UNCERTAIN NUMBER design pattern.

The core idea behind the UNCERTAIN NUMBER is to create a wrapper for numeric data types (e.g., double), which additionally stores the uncertainty value and implements arithmetic operators to propagate the uncertainty correctly. It uses the PROPAGATION STRATEGY pattern to accomplish this. The key is also to store the dependent

variables for each calculation, because they are needed to determine the internal correlations correctly. A `UNCERTAIN NUMBER` can be seen as a Composite Design Pattern consisting of two types of nodes:

- *Elementary Numbers*: This is directly measured values. The value and uncertainty are defined explicitly and do not depend on anything else. This would correspond to the leaves of a tree.
- *Compound Numbers*: This type represents the result of some calculation with other numbers. Its value, as well as its uncertainty, depends upon the originating variables. This would correspond to the nodes of a tree.

The class for `UNCERTAIN NUMBER` has the following attributes:

- *Value*: The numerical value for the `UNCERTAIN NUMBER`. This value represents a value that was directly observed and measured, or it represents the result of a calculation. It can always be immediately computed in every step and is the same as normally applying the numerical operators (without using `UNCERTAIN NUMBER` at all).
- *Uncertainty*: A numerical representation of the uncertainty. This value is directly defined for elementary numbers, or it is the result of uncertainty propagation for compound numbers. It can be calculated eagerly during the creation of the class, or later on when needed. It is calculated using Equation 10.
- *Dependents*: A list of dependent variables that influence the value. For elementary numbers, this only contains the object itself. For compounds, it contains all objects which influence the resulting value.
- *Propagation*: A propagation strategy function which calculates the corresponding uncertainty components for dependent variables. The `PROPAGATION STRATEGY` pattern is used for this. For elementary numbers, this returns its defined uncertainty, and for compound numbers, this depends upon the operation which is executed. This always follows the rule for the LPU, as defined in Equation 11.
- *Uncertainty-Function*: This function returns the uncertainty for a specific variable, and is just a transparent wrapper for easier access to the propagation strategy. This is used by the propagation mechanism, which will be explained later on.
- *Constructors*: There are two constructors: One is used to create elementary numbers, and the other is used to create compound numbers. The first one is the one that is used by the developer to create uncertain numbers the most. The second is for the implementation of mathematical operations and also used by the operators of the `UNCERTAIN NUMBER`.
- *Operators*: The most important mathematical operators like `+`, `-`, `*`, `/` are implemented to help the developer using the `UNCERTAIN NUMBER`. This allows us to change the data types of legacy applications easily, and everything will work automatically because the compiler can find the corresponding operators.

3.3.1 Example. The following source shows how to use the `UNCERTAIN NUMBER` design pattern. It implements a formula for calculating the velocity and distance of an object based on acceleration

measured by an accelerometer sensor. The inputs are the acceleration and the time, which both are uncertain numbers.

Listing 1: An example for using `UncertainNumbers` in actual source code. Here the distance is calculated based on some acceleration value.

```
public UncertainNumber CalculateDistance ()
{
    var t = new UncertainNumber (1, 0.001);
    // t = 1.000 +/- 0.001, elementary

    var a = new UncertainNumber (3, 0.2);
    // a = 3.0 +/- 0.2, elementary

    var v = a * t;
    // v ~ 3.0 +/- 0.2, compound

    var d = v * t + a * t * t / 2;
    // d ~ 4.5 +/- 0.3, compound
}
```

3.4 Consequences

- `UNCERTAIN NUMBER` provides means to store and propagate uncertainty throughout a system.
- Arithmetic operators are implemented to make the usage easy. These operators take care of the uncertainty propagation automatically.
- No mathematics knowledge is needed on using the pattern because it is all implemented and hidden away inside the `UNCERTAIN NUMBER` class.
- Correlations are considered as part of the propagation algorithms.
- Performance can be an issue if very long chains of calculations are done. In such cases, the whole syntax tree has to be evaluated in order to calculate the uncertainties. This could be mitigated by implementing the uncertainty as a lazy evaluation property, only calculating it when needed, and caching already calculated values.
- More memory is needed for every measured and calculated value because they need to store the value, uncertainty, and all the dependent variables.
- Unless `UNCERTAIN NUMBER` is used throughout a system, it still has the problem that the chain of uncertainty propagation is not complete.
- The direct access to the value as double gives the compatibility that `UNCERTAIN NUMBER` can be used in a legacy system where only double values are supported (although this brakes the propagation chain).
- The Interface of `UNCERTAIN NUMBER` is straightforward. The operators can be used in a familiar way, and also they make it easy to incrementally refactoring legacy systems to use `UNCERTAIN NUMBER`.

3.5 Known Uses

- **UncertainNumber** by Michael Krisper (C#, Python, C++): A framework that implements uncertainty propagation according to the GUM-Method and the Law of Propagation of Uncertainty. This is our implementation, which we created at the Institute for Technical Informatics at the Graz University of Technology.
- **GUM-TREE Pattern** by Blair D. Hall (C++/Python/R) [5, 9]: This very well described and researched design pattern has the same purpose and internal function for propagating the uncertainties, but is implemented using inheritance.
- **quantities package** by Eric-O. Lebigot (python) [7]: A package for representing physical quantities in python, which also includes uncertainty propagation. This is also implemented as a syntax tree, but the propagation is implemented using explicit partial derivatives for the variables instead of the automatic differentiation used in this paper.
- **Infer.NET (C#)**: A very sophisticated framework by Microsoft Research designed for probabilistic programming in .NET to be used in areas like machine learning.
- **Uncertain-T>(C#)** [1, 2]: Another framework for uncertainty by Microsoft Research designed for probabilistic programming but with much simpler interfaces.
- **A System for uncertainty propagation** by Luca Mari (Java) [8]: This is a framework for Java, which also implements automatic differentiation in order to do uncertainty propagation.

3.6 Related Patterns

- **INTERPRETER, SYNTAX-TREE**: This well-known design pattern by the GoF is the basic idea for the propagation trees used in this pattern [4].
- **GUM-TREE**: A design pattern that also implements uncertainty propagation according to the Guide to the Expression of Uncertainty in Measurement by B.D. Hall [5].
- **COMPOSITE**: This is the basis for the propagation trees where the node are either leaves (elementary numbers) or inner nodes (compound numbers) [4].
- **STRATEGY**: This is used for the propagation strategies to avoid inheritance (favor composition over inheritance) [4].

4 PATTERN: PROPAGATION STRATEGY

You have multiple uncertain values and want to do calculations. To maintain the uncertainty information, you must propagate it through all operations in an application.

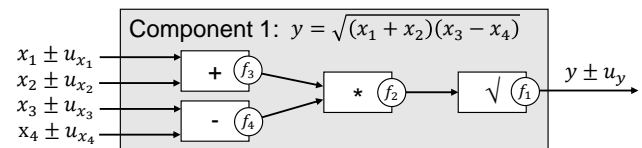


Figure 3: The uncertainty of values propagates with every calculation throughout a component.

4.1 Context

Applications doing calculations with uncertain numbers.

4.2 Problem

Applications calculate their results using uncertain data. During calculation, the uncertainties should propagate automatically without having to consider them manually on every operation.

4.2.1 Forces.

- Standard programming languages have no means for automatic differentiation, which must be used to do uncertainty propagation.
- Standard math libraries do not have the capabilities implemented for uncertainty propagation.
- Uncertainty propagation only works if used throughout all arithmetic operations.
- The mathematics of the LPU can be quite intimidating for developers, and could also be a source for errors if implemented wrong (automatic differentiation, syntax trees, function decomposition).
- Correlations must be considered since they have a significant impact on the result of the uncertainty estimation.
- There are many different mathematical operations that must be implemented.
- Systems are often limited in the amount of available memory. Thus, memory consumption needs to be carefully thought out. The amount of used memory and the amount of available information should be balanced.
- It should be easy to extend for new mathematical operations, without changing all existing source code.

4.3 Solution

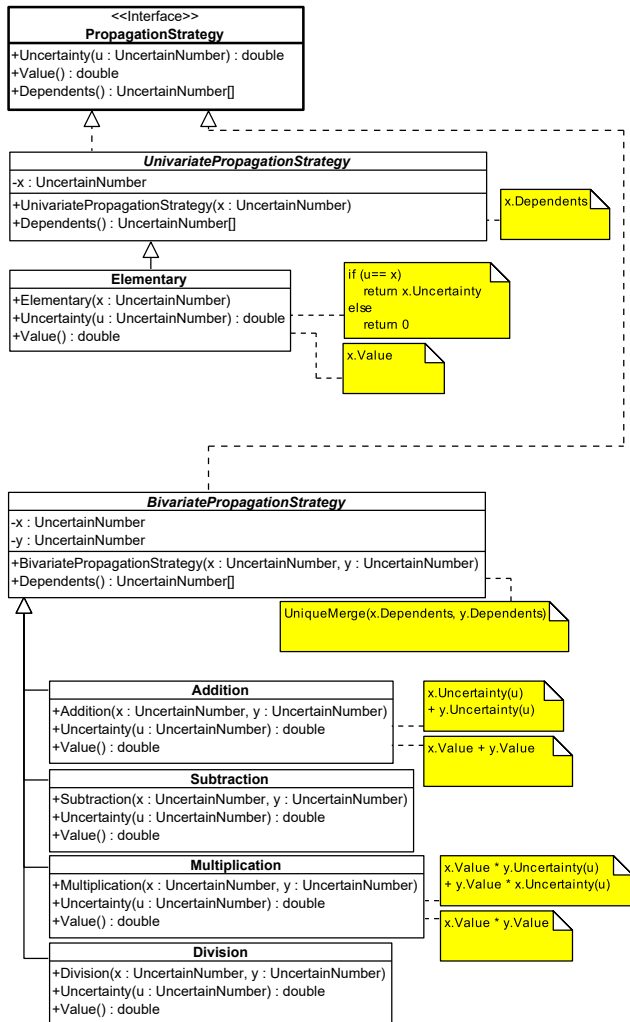


Figure 4: PROPAGATION STRATEGY pattern which consists of a simple interface and several specializations for the individual mathematical operations.

This pattern is structurally like the Strategy pattern by the Gang of Four [4]. The PropagationStrategy is an interface for objects which represents a single type of arithmetic operations like e.g., Addition, Multiplication, Log, or Sinus. It defines the following properties:

- *Uncertainty(u)*: A Method for return the respective uncertainty component for the variable u.
- *Value()*: A Method returning the respective numeric value for the operation.
- *Dependents()*: A method returning a list of dependent UncertainNumber variables for this operation.

Derived from that are UnivariatePropagationStrategy and BivariatePropagationStrategy, which represent operations with one dependent variable and two dependent variables, respectively. Furthermore, the actual strategies which implement the operation are

derived from one of those. Special is the Elementary class, which implements the propagation strategy for elementary numbers, which by definition do not need propagation. What it does is just returning the respective numbers for the numeric value and the uncertainty value - nothing more.

The BivariatePropagationStrategy builds the basis for all strategies with two dependent variables, e.g., Addition, Multiplication, Division. This is already used by the UNCERTAIN NUMBER pattern in its operators for the respective arithmetics operation: +, -, *, /.

4.4 Consequences

- Automatic differentiation is built into the Propagation Strategy and can be used to propagate uncertainties throughout an application.
- The mathematical equations used for uncertainty propagation is implemented to work according to the Guide to the Expression of Uncertainty in Measurements and the Law of Propagation of Uncertainty.
- Also, in accordance with the Guide, the correlations are considered for the uncertainty propagation.
- Since the dependent variables have to be stored, the system needs more memory than just the usage of the built-in numeric operations.
- The propagation strategy is easy to extend because new mathematical operations just have to be derived from Univariate or Bivariate PropagationStrategy, and the respective 3 Methods have to be implemented. Because they are isolated, this can be done in a few lines of code, and with the help of mathematical tools like wolfram alpha, the partial derivative for most functions can be found fast and easily in order to implement it in the Uncertain-Function.

ACKNOWLEDGMENTS

The authors would like to thank their shepherd Veli-Pekka Eloranta for providing helpful feedback throughout the shepherding process. We also want to thank the workshop group for giving us constructive feedback about the style, content, and readability of the paper.

REFERENCES

- [1] James Bornholt, Na Meng, Todd Mytkowicz, and Kathryn S. McKinley. [n. d.]. Programming the Internet of Uncertain< T> hings. 1–7.
- [2] James Bornholt, Todd Mytkowicz, and Kathryn S McKinley. [n. d.]. Uncertain<T>: A First-Order Type for Uncertain Data. ([n. d.]), 21.
- [3] C2. [n. d.]. Pattern Forms.
- [4] Erich Gamma, Richard Helm, Ralph E. Johnson, and John Vlissides. 1995. *Design Patterns - Elements of Reusable Object-Oriented Software* (1 ed.). Addison-Wesley, Boston, USA. 395 pages.
- [5] B. D. Hall. 2004. THE GUM TREE DESIGN PATTERN FOR UNCERTAINTY SOFTWARE. World Scientific Publishing Co. Pte. Ltd., 199–208. https://doi.org/10.1142/9789812702647_0017
- [6] JGCM. 2004. JCGM-WG1-SC1-N10 Guide to the expression of uncertainty in measurement (GUM) - Supplement 1: Numerical methods for the propagation of distributions. <https://www.bipm.org/en/publications/guides/gum.html>
- [7] Eric O. Lebigot. 2018. Uncertainties: a Python package for calculations with uncertainties. <https://uncertainties-python-package.readthedocs.io>
- [8] Luca Mari. [n. d.]. A computational system for uncertainty propagation of measurement results. 42, 6 ([n. d.]), 844–855. <https://doi.org/10.1016/j.measurement.2009.01.011>
- [9] Hall Blair Durham Willink, Robin Daniel. 2006. Uncertainty Propagation System and Method. , 23 pages.

A.4 [P4] Towards Integrated Quantitative Security and Safety Risk Assessment

- Authors: Jürgen Dobaj, Christoph Schmittner, **Michael Krisper**, Georg Macher
- Year: 2019
- DOI: 10.1007/978-3-030-26250-1_8
- Bibliography-Reference: [184]
- Presented at SafeComp 2019 conference in the DECSOS Workshop
- Published in the SafeComp 2019 conference proceedings.

Summary This paper analyses existing models for threat-modelling and risk-modelling in cybersecurity and shows that existing methods miss some attributes on the defender side. They are heavily focused on the attacker’s capabilities rather than the defenders. This is shown by applying the threat attributes of the SAHARA and FMVEA methods to the DIAMOND model, a graphical model of attacks in a distributed system. It shows that the defender side, as well as mitigation strategies, are often underrepresented. In addition to this, we also applied early ideas of RISKEE by using combined expert judgment and the notion of risk graphs which are revisited and refined again in follow-up papers. These are the key contributions in this paper (specifically point 3 was the part I contributed to the paper):

1. It maps the attributes and classifiers from methods for integrated safety and security evaluation (SAHARA, FMVEA) to the DIAMOND model.
2. It shows a lack of modelling capabilities for victim and mitigation strategies in existing methods based on threat modelling.
3. It applies the FAIR methodology and uses combined expert judgment for estimating the attributes of system resistance on a risk graph using probability distributions. This set the foundations for RISKEE, which followed in later publications in a more elaborated and refined version, but the idea and the main parts were conceived during working on this paper.

This paper was the origin of the idea for RISKEE. Many of the technologies and concepts used in RISKEE were already conceived in this paper. There are differences, though, e.g., in this paper, we used equally weighted experts in a linear opinion pool, which later was replaced by performance-based weighting using calibrations. In this paper we also still heavily relied on the terminology of the FAIR model, which was simplified and adapted afterwards.

My Contribution The majority of novel content in this paper was written by the first author Jürgen Dobaj. The other authors contributed background text and support as well as constructive discussions. My contributions are the parts about the FAIR method (section 2.5), quantitative risk assessment and combination of expert judgment (section 3.2), and shorter parts distributed over the whole paper, e.g., parts of Section 2.3 “Comparison to Established Methods”. The other authors provided texts about the other standards like FMVEA and SAHARA and parts of the introduction. All authors were involved in idea generation, as well as discussing and revising the paper.

Copyright ©2019 Springer Nature Switzerland AG. The version included in this dissertation is a permitted reprint of the published version from the proceedings of SAFECOMP 2019 Workshops, ASSURE, DECSoS, SASSUR, STRIVE, and WAISE, which can be accessed via SpringerLink:

https://link.springer.com/chapter/10.1007/978-3-030-26250-1_8

Towards Integrated Quantitative Security and Safety Risk Assessment

Jürgen Dobaj¹ , Christoph Schmittner² , Michael Krisper¹ ,
and Georg Macher¹ 

¹ Graz University of Technology, 8010 Graz, Austria

{juergen.dobaj,michael.krisper,georg.macher}@tugraz.at

² AIT Austrian Institute of Technology, 1020 Vienna, Austria
christoph.schmittner@ait.ac.at

Abstract. Although multiple approaches for the combination of safety and security analysis exist, there are still some major gaps to overcome before they can be used for combined risk management. This paper presents the existing gaps, based on an overview of available methods, which is followed by the proposal towards a solution to achieve coordinated risk management by applying a quantitative security risk assessment methodology. This methodology extends established safety and security risk analysis methods with an integrated model, denoting the relationship between adversary and victim, including the used capabilities and infrastructure. This model is used to estimate the resistance strength and threat capabilities, to determine attack probabilities and security risks.

Keywords: Security analysis · Safety analysis · Risk assessment · Threat analysis · Threat modeling · SAHARA · FMVEA · Diamond · FAIR

1 Introduction

Formerly, security played only a secondary role in safety- and mission-critical systems, since these systems were not connected to the Internet or the outer world. However, with the introduction of Internet-of-Things (IoT) and cyber-physical system (CPS) concepts into multiple industrial domains, the industry is undergoing enormous change towards highly interconnected and globally distributed automation and control systems, ranging from intelligent transportation systems [5] and industrial systems [27], to smart homes and smart cities [4]. Security mechanisms are responsible for protecting these systems from unwanted access or malicious attacks. Therefore, system security becomes an essential factor affecting the safety of mission-critical systems. Consequently, this requires an holistic dependability engineering approach integrating both, security and safety.

In particular, dependability is defined by multiple attributes (availability, reliability, safety, confidentiality, integrity, maintainability) that must be maintained and assured at a sufficient level. This is commonly achieved by considering the risk of potential threats (faults, errors, failures), followed by applying adequate risk reduction mechanisms (fault prevention, fault tolerance, fault removal, fault forecasting). It should be noted, that the term fault is quite generic, ranging from systematic weaknesses in software to insufficiently designed hardware.

Risk reduction denotes the effort to deliberately reduce risks to a tolerable level, instead of fantasizing about reducing all risks to zero, which makes it inevitable to prioritize risks and risk treatments. Risk treatment is defined as a cyclic process [10] of: (1) assessing existing risk treatments; (2) deciding if the residual risks are tolerable; (3) generating new risk treatments, if not tolerable; and again (4) assessing the new risk treatments. There are multiple risk treatments available, however, in dependable systems it is often required to implement specific measures for achieving a tolerable risk level. Therefore, decisions about risk treatments directly influence system engineering, requiring an evaluation of where engineering resources should be dedicated. Risk is generally defined by the likelihood and impact of a loss event classified according to only partially comparable categories (e.g. safety, financial, operational, privacy/confidentiality, ...). For example, functional safety considers safety impacts based on faults, errors, and failures of electric/electronic/programmable electronic (E/E/PE) elements. As long as risks compare similar categories, a similar likelihood scale can be used, which enables decisions on the required risk reduction mechanisms. However, different categories should not be mixed up, like financial loss and harm to human lives should be considered separately.

In the context of connected systems, it is essential to not only consider safety risks, but also security risks originating from malicious manipulations by e.g., internal or external actors. Such manipulations might have an impact on the same dependability attribute, but the resulting failure may be differently categorized, making it difficult to prioritize the risk treatments accordingly. Hence, there is and will be an ever increasing need to coordinate between the engineering processes that focus on different dependability attributes in system engineering [18]. This coordination requires combined methods as well as a common language for communicating and comparing risks. Whoever decides on the treatment of risks, should therefore, be provided with risk ratings in comparable scales.

In contrast to the statistical failure probability concept known from the safety domain, system security does not exclusively depend on statistical information about vulnerabilities and weaknesses, instead it is mainly driven by the interaction of an (human or machine) attacker against the resistance of a system. Integrating such human aspirations and motivations for mischief or selfish advantage into a likelihood system for risk is difficult, therefore imposing significant restricts to both, the coordination of security with safety risks, as well as with all other dependability attributes. While there are methods for combined considerations [16], they are still lacking some of the properties needed for a full risk

analysis, meaning identification and evaluation, which was a major finding of this paper and gives us open challenges to resolve, for protecting safety critical systems against malicious attacks [14].

In the course of this document, a discussion of related work and state-of-the-art analysis methods is provided in Sect. 2, which we expand by a brief discussion of the methods actual limitations to enable combined security and safety risk assessment. In Sect. 3, we introduce a new model for assessing the probability of security attacks in dependable CPSs. Therefore, we propose an approach that is based on established methods for combined considerations of safety and security features, which is accomplished by an established method for security incident analysis. This model is then used in Sect. 3.2 as qualified information framework to quantitatively classify the probability of cybersecurity attacks, by adapting and extending an established method for IT-security risk assessment. Section 3.3 presents an illustrative example, followed by an outlook and a closing discussion in Sect. 4.

2 Background

In this section an overview of relevant standards and context of related work is given. To that aim, also the differences of cyber-security and safety, as well as the applied integrated methods are briefly described. Additionally, as first contribution of this work, a comparison between established integrated risk assessment methods is given in Sect. 2.3.

2.1 Safety vs. Security

The idea of safety and security co-design has become a major trend of recent publications and is expected to appear more often in the future, also due to the upcoming security standards for safety critical domains, and the requirements on communication and coordination between safety and security. However, one of the main challenges of this merging of safety and security disciplines is the different level of maturity in the standards and the available knowledge in the domains. Safety, as well as security engineering focus on system-wide features and need to be integrated adequately into the existing process landscape; both having a major impact on product development and product release, as well as for company brand.

Therefore, a tight integration and cooperation between these two domains seems obvious and essential. The difference between safety and security, and one of the major show stoppers, is the very different point-of-view and the fundamentally different engineering approaches and nomenclatures. This issue has already been partially described and tackled in [18].

Beside this, functional safety engineering approaches focus on defects, failures, and errors, which can be foreseen (with reservations) at design-time, as well as on mathematical models based on failure probabilities and system models. Therefore, functional safety standards are defining domain specific processes and

methods for the development of safety-critical embedded systems. They target the minimization of systematic failures during development (e.g., requirement not implemented in the development phase) as well as the control of random failures during operation (e.g., component break-down). These standards rely on efficient quality management in project, and systematic hazard identification and management along the entire development life-cycle. Sound technical concept and validation planning, as well as trace management between these different items is a central aspect for safety augmentations.

On the contrary, security standards often just provide a set of high-level guiding principles for the life-cycle process framework, some basic guiding principles on cyber-security, or focus on a subset of the complete engineering process; but there are no common base practices or methodologies which are shared. Common Criteria [13], for example, is a detailed standard for security evaluation, but not applicable to security engineering.

Safety and security features have mutual impacts, sometimes similarities, and interdisciplinary values in common. However, these different attributes might lead to different targets, and mutual impact between safety and security exist. This even goes as far, that safety and security features frequently appear to be in total contradiction to the overall system features. A straight forward example of this contradiction can be shown by an electrical steering column lock system. In the security context, the system locks the steering column when in doubt, because this doubt area might result from an attack. From the safety perspective, however, it is highly undesirable to lock the steering column. Since, the issue involved might well be an occurrence directly before a high speed corner turn and would leave the driver without control over the steering wheel.

In addition to that, using non-integrated methods to manage these different attributes might lead to inconsistencies, which are identified in late development phases. Therefore, a solid information handover and a cooperative dependability engineering by cross-domain expert teams are required [16].

2.2 Qualitative vs. Quantitative Methods

Many established methods use qualitative assessments based on ordinal scales e.g. rating the severity of a safety hazard on a scale between 0 and 3, or rating the threat level of a security threat on a scale from “no security impact” to “moderate relevance” and “high security and possible safety relevance”. Typically these ratings consist of 2 to 3 ordinal scales which get combined either by addition or multiplication to obtain a final risk rating based on thresholds. Such qualitative assessments methods based on ordinal scales and so called “risk matrices” have several shortcomings including e.g., range compression/poor resolution, risk inversion, ambiguity, and neglecting correlations, which is shown in several publications by Hubbard et al. [7,8], and Cox et al. [2,3].

To avoid such pitfalls, we propose to apply a fully quantitative assessment method like factor analysis of information risk (FAIR), which is based upon the estimation of event frequencies, system vulnerabilities, and event impacts using probability distributions that enable to also take the respective uncertainty (or confidence - as it is called in the diamond model) of an estimation into account.

2.3 Comparison of Established Integrated Risk Assessment Methods

As already mentioned, risk management is an essential step in the development of critical systems. On a domain-independent level, risk management is defined by ISO 31000 [10]. For ISO and IEC standards the ISO/IEC Directives, Part 1 [12] requires all product or industry/economic sector specific risk management standards to reference or reproduce ISO 31000. ISO 31000 mentions quantitative and qualitative risk analysis and states that specific results should be consistent and comparable for effective risk management, e.g., all risks in comparable scales. Outside of ISO and IEC standards, there is also work on risk management. National institute of standards and technology (NIST), for example, published in 2012 *NIST Special Publication 800-30 (SP800-30) "Guide for Conducting Risk Assessments"* [15], for guidance in conducting risk assessments on federal information systems and organizations. SP800-30 also refers to the ISO 31000 risk management standards. In its main part, quantitative, qualitative and semi-quantitative approaches are discussed. In its annex a potential approach similar to (FAIR) is presented, where the likelihood of an event is divided into (a) a likelihood, that an adversary is initiating a threat event; and (b) a likelihood, that the threat event results in an adverse impact (i.e., a successful attack). Semi-quantitative values are given, and NIST SP800-30 warns that it can be challenging to assign a likelihood to a particular "bin" (e.g., 0–15, 16–35, 36–70), especially if it is between two levels.

In recent work [17], Macher et al., focused on enhancing the development lifecycle for automotive CPS by analyzing state-of-the-art methods for integrated security, safety, and reliability engineering. Their finally proposed framework is based on security-aware hazard and risk analysis (SAHARA), failure mode, vulnerabilities and effects analysis (FMVEA), and attack tree analysis (ATA), representing a promising approach for the integrated design of safe and secure systems in the automotive domain, which is the reason why we use them in our approach too. A comprehensive overview and comparison of related methods can be found in [16].

While SAHARA and FMVEA have its origin in the automotive sector, our proposed integrated risk management approach is not restricted to a specific domain. Instead, it supports the general level of ISO 31000 and allows to manage uncertainty and missing information in a risk management process.

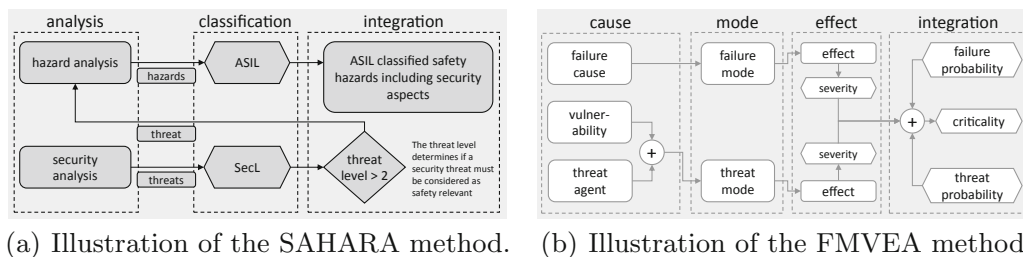


Fig. 1. Overview of the approaches for integrated risk assessment methods.

The SAHARA method, illustrated in Fig. 1(a), depicts a systematic approach to quantify the security impact on dependable safety-related systems on system level [19]. Therefore, the method combines the automotive hazard analysis and risk assessment (HARA) [11] approach with STRIDE [21] threat modelling.

The FMVEA method, illustrated in Fig. 1(b), is based on the failure mode and effects analysis (FMEA) as described in IEC 60812 [9] with additional support for security analysis, also based on the STRIDE threat modelling approach [22]. FMVEA uses a threat&failure-mode-effect model for its safety and security risk analysis targeted towards the item level.

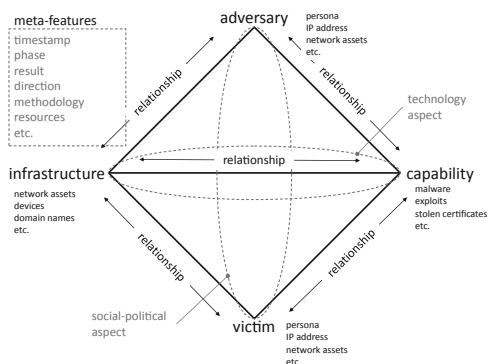
Integrated Risk Assessment. Both methods, SAHARA and FMVEA, describe integrated approaches that extend established safety risk analysis methods to not only classify the risk of system failures, but also the risk of security threats. For this cybersecurity risk classification the methods define schemes similar to those known from safety engineering. However, instead of finding and rating potential system failures and failure causes, cybersecurity risk assessment is targeted towards identifying and rating potential vulnerabilities and threats and the interplay of both with assumed attackers. Therefore, the methods provide rating schemes to assess (i) the attacker strength, denoted by the attacker capabilities, intention, and know-how; (ii) the system resistance, partially denoted by static security measures classified by the system reachability, structure, and required attack tools; and (iii) the impact of a successful attack, denoted by the effects on the system and its environment.

The attacker strength and the system resistance are combined to estimate the probability of a potentially successful attack. The attack probability and impact determine the criticality level of a security threat, which is also used to indicate the safety relevance of a security threat. Beside the criticality level, the SAHARA method also specifies a so called security level (SecL) to provide guidance in selecting the appropriate number of countermeasures that should be considered [19]. The FMVEA, on the other hand, uses the resulting risk priority number (RPN) as a comparable indicator to focus the development efforts on the most critical issues and system areas [22].

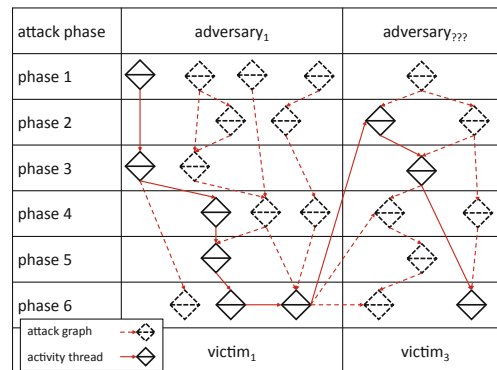
For determining the SecL and RPN, both methods rely on qualitative measures. However, qualitative measures are not suitable for mathematical models to calculate the overall vulnerability of the system. Moreover, the measures lack calibration with the failure probability to enable an integrated combined safety and security risk management [22]. Another, general limitation of both methods is the restriction to analyze only single causes of an effect [22]. Hence, multi-stage attacks could be overlooked, which, however, would be of particular relevance for analyzing security attacks.

2.4 The Diamond Model of Intrusion Analysis

The *diamond model of intrusion analysis* [1] is an established formal method to analyze cyber-incidents after their occurrence. Figure 2(a) illustrates a diamond that represents a basic atomic element of any intrusion activity, also denoted as (security) event. The key assumption of the diamond model is that “for every intrusion event there exists an **adversary** taking a step towards an intended goal by using a **capability** over **infrastructure** against a **victim** to produce a result” [1]. Hence, an event is composed of four core-features (described later in Sect. 3.1): adversary, capability, infrastructure, and victim. These features are arranged in the shape of a diamond, where the edges represent the underlying relationships between the features. The diamond further defines meta-features to support higher-level constructs, which includes linking multiple events to form activity threads and attack graphs. These threads and graphs are illustrated in Fig. 2(b). Activity threads and attack graphs are comparable to attack trees [23]. An **activity thread** consists of a set of diamonds representing an attack path through the graph. An **attack graph** enumerates multiple paths an adversary could have taken in the attack, while an activity thread represents an already identified attack path.



(a) Illustration of the diamond model of intrusion analysis (adapted from [1]).



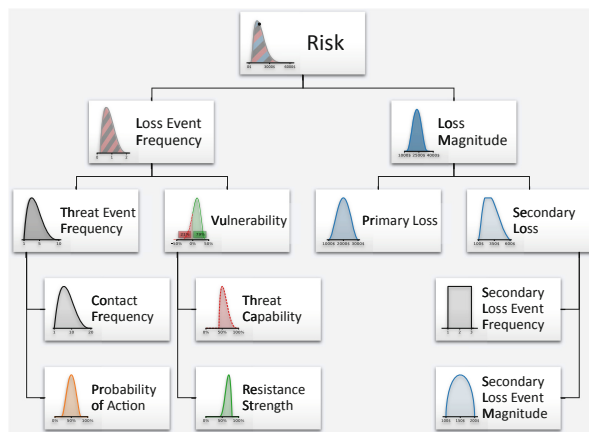
(b) Illustration of activity threads and attack graphs (adapted from [1]).

Fig. 2. Excerpt of the diamond modelling capabilities.

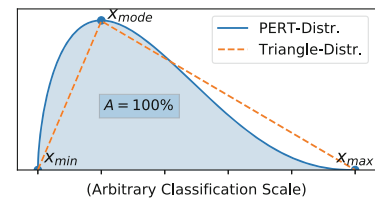
After the detection of an incident, there is generally limited information about the attack sequence and exploited vulnerabilities available. Thus, the major idea of the diamond model is that the events, threads, and graphs form a documentation and information framework that facilitates the structured analysis of such incidents. This helps analysts to ask the right questions to uncover missing links, vulnerabilities, and the actual adversary. By assigning confidence values to both, core- and meta-features, the confidence into the actual analysis is documented.

2.5 The FAIR Method for Risk Analysis

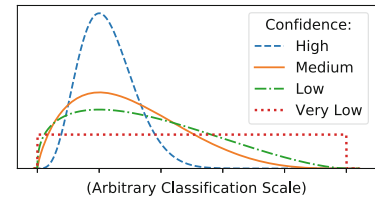
The FAIR method is a way of determining the risk of an attack event [25]. It is based on splitting risk into several sub-factors to more easily evaluate IT-security and operational risk [6, 25, 26], as shown in Fig. 3(a). These sub-factors are rated in the form of expert judgements by describing the minimum, maximum, and most likely value including a confidence rating. The judgements, as shown in Fig. 3(b) and (c), are modelled as program evaluation and review technique (PERT) probability distributions, which originates in project management and was first used by the US Navy to estimate time plans for missions [20].



(a) The FAIR taxonomy showing the sub-factor decomposition, each having a probability assigned (adapted from [26]).



(b) The PERT distribution.



(c) The same judgement with different confidence levels.

Fig. 3. Overview of the FAIR taxonomy and the PERT distribution.

3 Contribution: Towards a Quantitative Integrated Risk Assessment Method

In the preceding sections we compared the SAHARA [19] and FMVEA [22] methods for integrated risk assessment in CPS. Subsequently, the diamond model [1] and the FAIR method [25] are introduced to now propose an integrated quantitative risk assessment model that maps features, described by SAHARA and FMVEA, into the diamond model. We propose a methodology for quantitative security risk assessment by combining all these methods, which enables the analysis of security and failure event chains, as well as a coordinated risk management. Therefore, we are using a combined terminology from both, Diamond and FAIR.

3.1 Combining SAHARA, FMVEA, and Diamond into One Model

As already discussed in the previous sections, there exists potential for improving the established integrated risk analysis methods. Since, the diamond model describes a structured model and process for cyber incidents analysis, we propose that the model is capable to complement the integrated methods enabling a more comprehensive risk analysis.

The key assumption of the diamond model is that *“for every intrusion event there exists an adversary taking a step towards an intended goal by using a capability over infrastructure against a victim to produce a result”* [1]. Such an event is modeled by four core-features: (i) the adversary, (ii) the capability, (iii) the infrastructure, and (iv) the victim; as well as arbitrary definable meta-features. We use this meta-features to map the SAHARA and FMVEA attributes and classifiers into the diamond model, which is shown in Fig. 4. The boxes represent SAHARA and FMVEA attributes, which are rated by the classifiers illustrated as hexagons.

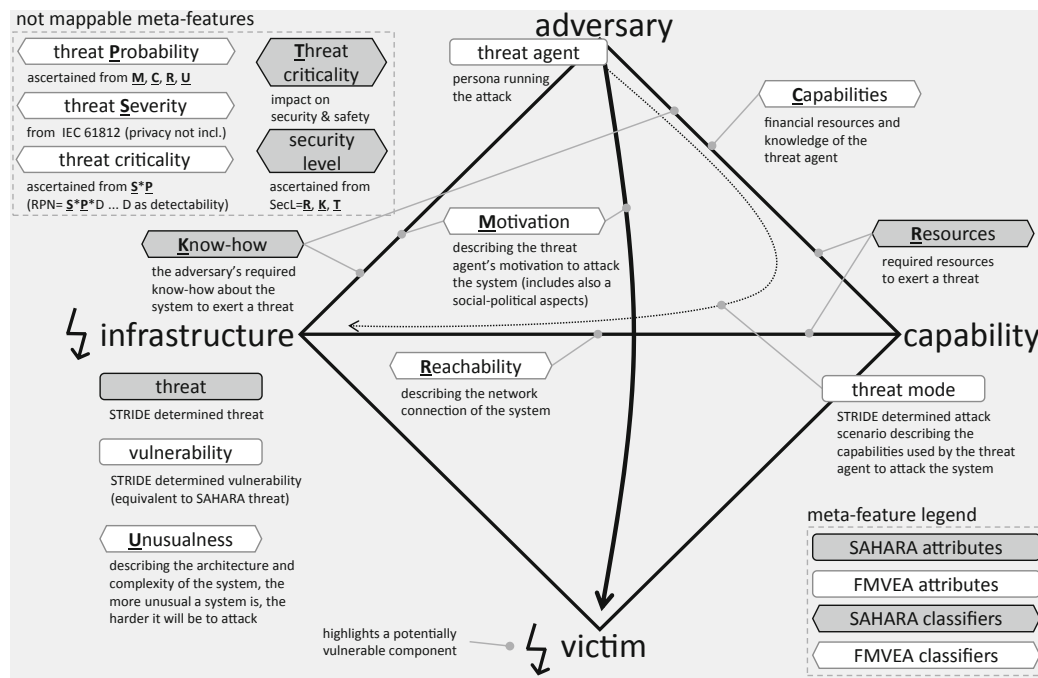


Fig. 4. Illustration of the diamond model including the mapping of the security attributes and their classifiers described by SAHARA and FMVEA.

(i) **The adversary feature** describes a set of adversaries (e.g., in/outside, individuals, groups, and organizations) which seek to compromise a system to satisfy their intent and needs. An adversary could be described by (a) an adversary operator, the person conducting the intrusion activity; and (b) an adversary customer, the entity that benefits from the conducted activity and generally acts as the funding entity, but might be the same person as the operator [1].

The SAHARA method does not explicitly describe or classify an adversary, however, the adversary is implicitly describes by its KNOW-HOW and the RESOURCE classifiers. The KNOW-HOW classifier determines the knowledge and insight an adversary requires to attack the infrastructure, distinguishing between black-, grey-, and white-box views. The method further names adversary examples (e.g., average driver, electrician, mechanic, ...) for each of the views imposing capabilities on the adversary. Hence, KNOW-HOW is mapped onto two edges describing both, the *adversary – infrastructure* and the *adversary – capability* relationship. The RESOURCE classifier determines the required resources an adversary must possess to attack the infrastructure, ranging from no tool support to advanced tool support. The RESOURCE classifiers is mapped onto the *adversary – capability* relationship describing the tools required to deliver an attack over infrastructure, described by the *capability – infrastructure* relationship.

In contrast, the FMVEA method explicitly states the adversary as threat agent classified according to ISO 27005 [24] making the mapping obvious. The ISO 27005 classification also characterizes the adversaries CAPABILITIES described by its financial resources and knowledge. FMVEA refines this by its MOTIVATION classifier, which takes both, technical and social-political aspects into account, described by the *adversary – infrastructure* and *adversary – victim* relationships.

(ii) The capability feature captures the tools and techniques an adversary used within a diamond event. This can be divided into (a) the capability capacity, all vulnerabilities and exposures of the target system that could potentially be utilized by the adversary; and (b) the capability arsenal, the actual set of the adversary’s capability. The capacity is used to also document non-exploited vulnerabilities providing input for potential system improvements [1].

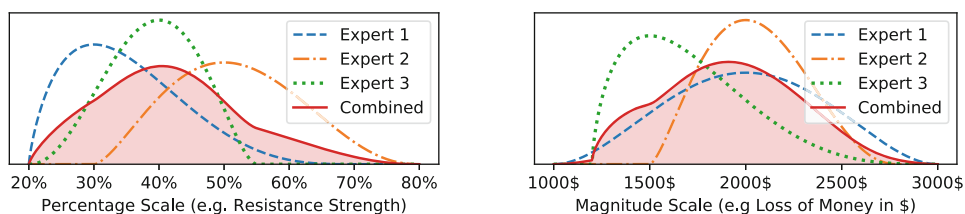
(iii) The infrastructure feature describes physical and/or logical communication structures that are used by the adversary to deliver a capability, maintain control of capabilities, and effect results on the victim. The infrastructure feature is divided into three types: (a) type 1 infrastructure, is fully controlled or owned by the adversary or which they may be in physical proximity; (b) type 2 infrastructure, is controlled by an (witting or unwitting) intermediary, which is typically the infrastructure an adversary uses to obfuscate its actions; (c) service providers, are organizations that (witting or unwitting) provide services critical for availability of type 1 and type 2 infrastructure [1].

Both, SAHARA and FMVEA, are based on the STRIDE [21] threat modelling approach to identify and categorize potential THREATS and VULNERABILITIES of the infrastructure. The FMVEA method uses this categorization to describe and identify potential THREAT MODES specifying the manner in which security fails, which is similar to the failure modes of safety. In terms of the diamond model a THREAT MODE can be described as the resources and infrastructure used by the adversary to deliver its capabilities, represented as the *adversary – capability – infrastructure* relationship.

(iv) **The victim feature** describes the target of the adversary and against whom vulnerabilities and exposures are exploited and capabilities used. It is useful to divide the victim assets into (a) victim persona, the people and organizations targeted whose assets are exploited and attacked; and into (b) victim assets, the attack surface consisting of networks, systems, hosts, etc. against which the adversary directs their capabilities. It should be considered that in multi-stage attacks a victim asset, can be the end target in one event and then leveraged as the infrastructure in further events. Thus, one must always be aware that the target of an activity may not necessarily be the victim. Further, the victim assets often exist both, inside and outside a persona's control or visibility. However, still available for targeting by an adversary, which commonly includes cloud-based data storage and applications [1].

3.2 Extending FAIR for Risk Analysis Based on Diamond Events

After mapping the attributes from SAHARA and FMVEA to the diamond model, we have a qualified information basis for estimating the actual probability, severity and risk of a diamond event. For this estimation we apply and extend the FAIR method to give consolidated and refined expert judgements of the resulting risk within a diamond event. We implemented this in a mathematical framework to combine and propagate probabilities for quantitative security and safety risk analysis, providing the basis for future applications.



(a) Multiple expert judgements of probability values for resistance strength are combined into a mixture distribution.

(b) Multiple expert judgements for magnitude of impact (in this case: the loss of money) are combined.

Fig. 5. Combination of multiple expert judgements shown for different scales of value ranges. The area under the distribution is always normalized to 100%.

Combining Multiple Expert Judgements. We refined the FAIR method by combining multiple expert judgements to obtain a more realistic probability space for the respective value distribution. The resulting mixture probability distribution supports multiple centers of mass, better reflecting the given judgements, and also supporting differing confidence levels between single judgements. Figure 5(a) and (b) show examples of such mixture distributions from multiple experts. It is possible to mix different distribution types within the mixture model, so we are not limited to PERT distributions only.

3.3 Discussion of Enhancements and Open Issues

This section provides an illustrative example, shown in Fig. 6, that summarizes and critically discusses the proposed approach. Like most integrated methods, as outlined in Sect. 2, SAHARA and FMVEA are based on threat modelling to identify potential security risks. Both, SAHARA and FMVEA, use the STRIDE threat modelling approach [21] as starting point for their security analysis, as indicated on the top-left corner in Fig. 6. Our approach utilizes the diamond attack graphs to model and document the identified threats and attack scenarios recognized by applying the STRIDE approach. The obtained attack graphs have the advantage, that the analysis of attack event chains is supported, similar to attack trees [23]. Moreover, their meta-feature concept makes attack graphs more generic, allowing to easily extend the model with additional information, including trace links to requirements, and implementations, as well as capturing and classifying other dependability attributes, like safety. Furthermore, the diamonds emphasize the technical and social-political relationships between adversary and victim, which supports analysts and designers in identifying otherwise not found vulnerabilities and threats, as well as missing information. This observation is illustrated by the SAHARA and FMVEA classifier mapping, which mainly describe the upper diamond half characterizing the system **threat capability** for the FAIR judgement. The lower diamond half, reflecting the victim and its defense capability, is only partially described by the reachability and unusualness classifiers, representing static

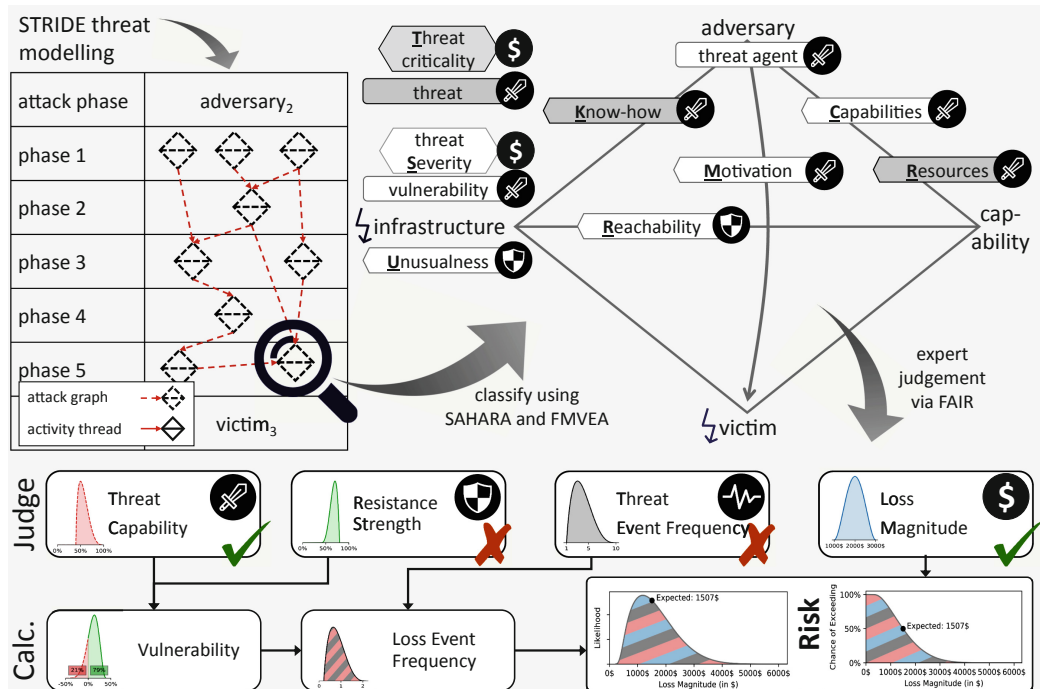


Fig. 6. Illustrative example showing the transition from FMVEA and SAHARA to the Diamond model and evaluating the risk with FAIR. The check-marks indicate, if additional classifiers or information is required to obtain profound expert judgements.

security measures only. To better estimate the system **resistance strength**, the proposed model needs to be extended to provide additional classifiers, also covering active security measures that deter, detect, report, and react against threats.

Actually, none of the methods (i.e., SAHARA, FMVEA, FAIR) supports estimating the **threat event frequency**, since only the probability of the single first attack is rated, and no potentially subsequent attacks (by different adversaries) are considered. While this is aligned with the idea of “failure” in the sense of safety, it is insufficient for analyzing security risks, due to fact that an attack could occur multiple times by multiple attackers without necessarily resulting in faults or failures.

The threat criticality ensures that safety relevant threats are handed over to safety management, and together with the threat severity, they already provide a good basis for estimating the **loss magnitude** of a diamond event.

In short, the proposed method provides the basic premises to integrate/combine security and safety risk assessment by providing means for an integrated quantification approach and a holistic model supporting both, threat and mitigation modelling.

4 Conclusion and Future Work

The primary contribution of this paper is the the definition of a method for integrating security into a combined risk assessment model. To obtain this model we mapped the classifiers described by two integrated risk analysis methods from the automotive domain (i.e., SAHARA and FMVEA), into the diamond model that has its origin in the field of security incident analysis. The obtained mapping reveals that the risk analysis methods do not consider all potentially relevant aspects that are studied in security incident analysis, which clearly encourages the usefulness of a combined model, which can serve as a comprehensive and qualified information basis for evaluating and documenting the actual probability, severity and criticality of security risks.

The mapping further reveals the lack of victim and mitigation strategy models, which are required to estimate the system resistance strength against potential cyber-attacks. Therefore, we encourage the development of a combined method that considers both, threat modelling and mitigation modelling. While *threat modelling* exclusively focuses on the adversary *threat capabilities*, a so called *mitigation model* would also provide a profound basis for judging the system *resistance strength*. Another strong reason for mitigation models is, that system security does not exclusively depend on the existence of vulnerabilities and weaknesses, instead it is mainly driven by the interaction of a human or machine attacker against the resistance of a system. By allowing to also cover the victim assets and deployed mitigation mechanisms, the model proposed in this work, allows estimating the system resistance strength too. However, the proposed model must still be enhance to capture both, passive (already covered) as well as active security measures allowing to deter, detect, report, and react against threats.

Since we are aiming towards a quantitative risk assessment, the model proposed in this work uses the established FAIR method for risk estimation. However, we propose to extend the FAIR method to be applicable on whole attack graphs, instead of single events only, which we are planning to model by an attacker evolving after each successful attack. To finally obtain a model that covers the whole system lifecycle, we are planning to introduce a time based risk prediction model to capture the evolution of attacker capabilities and the system resistance decay over time.

References

1. Caltagirone, S., Pendergast, A., Betz, C.: The diamond model of intrusion analysis. Technical report, Center for Cyber Intelligence Analysis and Threat Research Hanover MD (2013)
2. Cox, A.L.: What's wrong with risk matrices? *Risk Anal.* **28**(2), 497–512 (2008). <https://doi.org/10.1111/j.1539-6924.2008.01030.x>
3. Cox, L.A.: Some limitations of “risk = threat vulnerability consequence” for risk analysis of terrorist attacks. *Risk Anal.* **28**(6), 1749–1761 (2008)
4. Elmaghraby, A.S., Losavio, M.M.: Cyber security challenges in smart cities: safety, security and privacy. *J. Adv. Res.* **5**(4), 491–497 (2014)
5. European Commission: A European strategy on Cooperative Intelligent Transport Systems, a milestone towards cooperative, connected and automated mobility. Technical report, European Commission, November 2016
6. Freund, J.: *Measuring and Managing Information Risk: A FAIR Approach*. Butterworth-Heinemann, Oxford (2015)
7. Hubbard, D., Evans, D.: Problems with scoring methods and ordinal scales in risk assessment. *IBM J. Res. Dev.* **54**(3), 2 (2010)
8. Hubbard, D.W., Seiersen, R.: *How to Measure Anything in Cybersecurity Risk*. Wiley, Hoboken (2016)
9. IEC: IEC 60812: Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA) (2006)
10. ISO: ISO 31000 - risk management - guidelines
11. ISO: ISO 26262 Road vehicles - Functional safety (2011)
12. ISO/IEC: ISO/IEC directives, part 1
13. ISO/IEC: ISO/IEC 15408: Information Technology Security Evaluation (2005)
14. Johnson, C.W.: Why we cannot (yet) ensure the cybersecurity of safety-critical systems. In: *Proceedings of 24th Safety-Critical Systems Symposium*, pp. 171–182 (2016)
15. Joint Task Force Transformation Initiative: Guide for conducting risk assessments. <https://doi.org/10.6028/NIST.SP.800-30r1>
16. Lisova, E., Sljivo, I., Causevic, A.: Safety and security co-analyses: a systematic literature review (2018)
17. Macher, G., et al.: Integration of security in the development lifecycle of dependable automotive CPS (2017)
18. Macher, G., Höller, A., Sporer, H., Armengaud, E., Kreiner, C.: A comprehensive safety, security, and serviceability assessment method. In: Koornneef, F., van Gulijk, C. (eds.) *SAFECOMP 2015*. LNCS, vol. 9337, pp. 410–424. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24255-2_30

19. Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: SAHARA: a security-aware hazard and risk analysis method. In: Design, Automation and Test in Europe Conference and Exhibition (2015)
20. Malcolm, D.G., Roseboom, J.H., Clark, C.E., Fazar, W.: Application for a technique for research and development program evaluation (1959)
21. Microsoft Corporation: The STRIDE Threat Model (2005). <http://msdn.microsoft.com/en-us/library/ee823878%28v=cs.20%29.aspx>
22. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (FMEA). In: Bondavalli, A., Di Giandomenico, F. (eds.) SAFECOMP 2014. LNCS, vol. 8666, pp. 310–325. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10506-2_21
23. Schneier, B.: Attack trees (1999). <http://www.schneier.com/attacktrees.pdf>
24. International Organization for Standardization (ISO), I.E.C.I.: Information technology – Security techniques – Information security risk management (2008)
25. The Open Group: Risk Analysis (O-RA), October 2013
26. The Open Group: Risk Taxonomy (O-RT) 2.0, October 2013
27. Xu, L.D., Xu, E.L., Li, L.: Industry 4.0: state of the art and future trends. *Int. J. Prod. Res.* **56**(8), 2941–2962 (2018)

A.5 [P5] RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber-Security

- Authors: Michael Krisper, Jürgen Dobaj, Georg Macher, Christoph Schmittner
- Year: 2019
- DOI: 10.1007/978-3-030-28005-5_4
- Bibliography-Reference: [98]
- Published at EuroSPT'19 conference proceedings (Springer, CCIS, volume 1060).
- Presented at EuroSPT'19 conference in Edinburgh, Scotland, UK.

Summary This paper established the RISKEE tool and the method for calculating risk in a risk-tree using range estimations with probability distributions and propagating them forward and backward over a graphical model of a system. It uses the first prototype of RISKEE for preliminary assessments and calculations. RISKEE is a tool and method for risk estimation of cybersecurity in networked systems. This paper contains the following contributions:

- **RISKEE as a method:** The RISKEE method describes the process of structured expert judgments based on the Delphi method and range estimations for risk attributes in the domain of cybersecurity. It consists of three phases: First, the calibration, and then two rounds of expert judgments and discussions. During calibration, the experts are tested for their judgment quality. Based on this test result, their judgments will be weighted later on. In the first judgment round, the experts give an initial individual judgment. The results are presented and discussed, and afterwards, a second revised and reflected judgment is given. In the end, these judgments are combined using the weights from the calibration.
- **RISKEE as a tool:** The RISKEE tool is an implementation of the system model and the uncertainty propagation. It models, weights, and combines the expert judgments as PERT distributions and propagates these distributions forward and backward over the attack paths of a specified risk graph. This tool is implemented in python 3 using numpy, scipy, and networkX. The arithmetic for probability distributions is implemented using systematic sampling and reflected and truncated kernel density estimation for smoothing out the results. The tool also implements many methods to visualize the risk graph and the resulting risk curves (loss exceedance curves).

This paper comprises the core contribution of this thesis, combining all the concepts described here. Publications [P1], [P2], [P3], and [P4] have been progenitors and prepared the basis for this paper, and the other publications [P6], [P7], [P8], [P9], [P10], and [P11] reused, applied, and evaluated the RISKEE tool and method.

My Contribution This paper was written entirely by me, but its main idea (uncertain risk assessments propagating over risk graphs) was based on an earlier paper by the same authors (see paper [P4]). I revised and elaborated on these aspects of risk graphs and uncertainty propagation which were only briefly mentioned in the previous publication. The other authors contributed with discussions and feedback to this paper.

Copyright ©2019 Springer Nature Switzerland AG. The version included in this dissertation is a permitted reprint of the published version from Proceedings of the European & Asian System, Software & Service Process Improvement & Innovation (EuroAsiaSPI), September 2019, which can be accessed via SpringerLink: https://link.springer.com/chapter/10.1007/978-3-030-28005-5_4

RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber Security

Michael Krisper¹ , Jürgen Dobaj¹ , Georg Macher¹ ,
and Christoph Schmittner² 

¹ Graz University of Technology, 8010 Graz, Austria

{michael.krisper, juergen.dobaj, georg.macher}@tugraz.at

² AIT Austrian Institute of Technology, 1020 Vienna, Austria

christoph.schmittner@ait.ac.at

Abstract. In this paper, the RISKEE method for evaluating risk in cyber security is described. RISKEE is based on attack graphs and the Diamond model combined with the FAIR method for assessing and calculating risk. It can be used to determine the risks of cyber-security attacks as a basis for decision-making. It works by forwarding estimations of attack frequencies and probabilities over an attack graph, calculating the risk at impact nodes with Monte-Carlo simulation, and propagating the resulting risk backward again. The method can be applied throughout all development phases and even be refined at runtime of a system. It involves system analysts, cyber security experts as well as domain experts for judgement of the attack frequencies, system vulnerabilities, and loss magnitudes.

Keywords: Risk assessment · Risk propagation · Attack trees · FAIR method · Diamond model · Cyber physical security · IT-security

1 Introduction

In earlier work, we established the idea of combining existing methods for safety and security for the automotive and industrial domain in a quantitative way to come up with a fully integrated quantitative risk assessment [9]. During working on that topic, we stumbled upon several problems with existing methods and we are now on a pursuit of solving them.

Risk is the notion of an event which may occur in the future and which may have negative outcomes (for positive outcomes it is called *opportunity*). Classically this can be expressed in mathematical terms like this:

$$Risk = Probability \times Impact \quad (1)$$

Probability is a number between 0 and 1 (0% to 100%) and *Impact* is a number denoting a quantitative measure of the loss if the event occurs (e.g. 1000\$). This impact could be actual monetary loss, where you have to pay some money or

© Springer Nature Switzerland AG 2019

A. Walker et al. (Eds.): EuroSPI 2019, CCIS 1060, pp. 45–56, 2019.

https://doi.org/10.1007/978-3-030-28005-5_4

replace some device, but it also could be loss in the form of reduced income or revenue. The calculation of risk helps to compare and evaluate the events and furthermore to make decisions based on that evaluation. By knowing the total risk of a project, one could reserve enough financial resources to overcome expected losses (recovery), or try to decrease high risk events down to a tolerable level by lowering the probability or decreasing the impact (prevention). It is important to note that risk represents only the *expected* amount of loss, which is an artificial value, because it is derived in a probabilistic way. This value represents the expected average which, due to the law of large numbers, is only realistically accurate in the case of high sample sizes. For small sample sizes this estimation could be completely wrong. This is due to the fact that traditionally risk estimates are just point estimates which do not take into account uncertainty, sample size and confidence. If we would judge the probability and impact in form of a range of values which also includes the uncertainty, we could depict the resulting plausible range for the risk, which even for small sample sizes could give us an estimate with high confidence. This is the basic idea of this papers' contribution, the RISKEE method for risk assessment using attack trees and probability distributions.

Beside neglection of uncertainty, another problem is the usage of ordinal scales [17,28], especially in areas which are difficult to quantify. For example, in safety, risk is not measured in monetary values, but instead with harm or danger to human life, which is much more difficult to measure. Because of that, often ordinal scales are used which define increasing levels of injury or harm. Methods which use such ordinal or even nominal scales are called qualitative or semi-quantitative methods. These levels and thresholds of ordinal scales have several drawbacks, e.g. they are often completely arbitrary, introduce quantization errors, or are ambiguous [6–8], but nevertheless they are commonly used because they seem simple to understand, to use, and to evaluate, although this may be just a perceived impression of benefits, which cannot be proven in reality [14,33]. But what's even worse: Ordinal scales don't allow arithmetic operations. They allow ordering relations like equal, smaller, or larger, but addition or multiplication are not defined. Think of multiplying two t-shirt sizes: x-large * small. Is this reasonable? No. The size of t-shirts is just one example for an ordinal scale, it allows for assessments of smaller or larger, but nothing more. Bizarrely enough, for risk assessment we have no reluctance of multiplying two ordinal scales together. For example in the failure mode and risk analysis (FMEA) [15], the input values for severity going from 1 (none) to 10 (hazardous without warning) are multiplied with the occurrence going from 1 (<0.001% of cases) to 10 (>10% of cases). The result is called risk priority number (RPN) and the risks are prioritized according to this number. This has been proven to be wrong and inaccurate many times over [2,3,11,13,27].

Many existing methods for analysis of security and safety use such qualitative judgements to evaluate the risk of a security breach, or the risk of danger and harm to human life. These methods use expert judgements based on arbitrary quasi-quantitative ordinal scales to judge values like e.g. exposure, severity,

knowledge, resources, criticality and so on. This results in an overly simplified and rough classification, which is to unprecise and error-prone. That is why we began working on a method which used real frequencies, probabilities and impacts to evaluate the risk of a system for its cyber-physical-security, but also safety. We call this method *RISKEE*, and our intention was to develop an easy to use method to evaluate risk in attack-trees in a quantitative way.

The remainder of this paper is structured as follow: In Sect. 2 the background of the work is presented and related work including its shortcomings are discussed. The contribution of this paper is shown in Sect. 3. Section 4 discusses the limitations and current challenges and concludes the paper.

2 Background and Related Work

In this section we describe the background as well as related work for the proposed method. First, we shortly summarize our earlier work on that topic, then we shortly dive into a comparison of existing methods. Afterwards we describe the Diamond model of Intrusion Analysis, and the FAIR method for risk assessment.

2.1 Towards Unified Quantitative Risk Assessment of Safety and Security

In our previous work [9], we connected established methods for safety and security assessment (namely SAHARA [19] and FMVEA [25]) to create an informed knowledge base in form of the Diamond model [4] for evaluating the risk using the FAIR method [30, 32]. While this work was important for our understanding, it also opened up many questions and showed problems in the existing methods. One of the results was, that methods primarily focus on the attacker side, and neglect the victim, which is reasonable since those methods based on threat modelling, which emphasizes threats, not defenses. Due to focusing reasons, we will not tackle this in the current paper, but we have it on our todo list and will be solved in future work.

In the following paragraphs, we repeat the fundamentals of some established methods for risk assessment in safety and security for the following methods: SAHARA, FMVEA, and ATA. We considered them, because they are the proposed methods by an analysis of state-of-the-art methods for integrated security, safety and reliability engineering by Macher et al. [18].

The first method is SAHARA [19], which is based on HARA (Hazard and Risk Analysis [16]), and extends it by using STRIDE [22] to find the security attack vectors. The attack vectors are evaluated on an ordinal scale according to the required resources, know-how and threat criticality, which are combined to a security level according to an evaluation matrix. If the resulting security level exceeds a specific threshold, the attack vector is considered for further safety analysis in the HARA.

The second method is FMVEA [25], which is based on FMEA [15] and again extends it by using STRIDE [22] to find additional failure modes which are caused by security attacks. Through the description of vulnerability and threat agent, the threat probability, severity, and furthermore the criticality, can be determined on an ordinal scale. The criticality can be used for prioritization to find the most critical failure and threat modes which have to be mitigated.

The third method, ATA [1, 34] or attack tree analysis, is of special interest for us because it is a graphical model based on attack trees [26], which are used in RISKEE. It has its origins in safety, especially fault-trees and fault-tree analysis (FTA) [20, 35, 37]. In ATA, the events are not a simple list, but they are arranged in a tree structure where the root node is the attacker’s goal, and the leaves are the steps which are needed to reach this goal. With every layer of the tree the steps get more detailed and refined. The nodes in the attack tree can have specific attributes which are needed to analyze the tree for e.g. the most feasible or dangerous attack paths. The idea of graphical representations of attack paths to analyze cyber security attacks over some given infrastructure was extended over the years to cover whole attack graphs, bayesian networks, belief propagation, markov chains, and petri nets, and many others. For example, Poolsappasit et al. use an attack graph to implement bayesian belief propagation [23] and apply genetic optimization algorithms to calculate pareto-optimal combinations of mitigation techniques.

2.2 FAIR

In FAIR, risk is decomposed into several subfactors, which can be evaluated more easily. It establishes a whole ontology of these subfactors which are mathematically related in a precise way. To estimate these subfactors, expert judgement and historical data is used, but always including the respective uncertainty or confidence in the data. Therefore, the judgements are given as value ranges, or probability distributions which represent the likelihood of values for the respective subfactor. Figure 1 shows the FAIR ontology and its subfactors as well as the according estimations in form of probability distributions.

The modified PERT-beta distribution [24, 36] is then used to model these expert judgements by using the parameters: minimum, maximum, most likely value and confidence.

By using monte carlo simulations [21] and the mathematical relations of the subfactors, FAIR can calculate the probabilities and likelihoods for a range of risk values. The result is a loss exceedance curve, which depicts the outcomes and respective probabilities thereof. This can be compared to the risk appetite curve, in order to decide if the risk is tolerable or not. For further information we

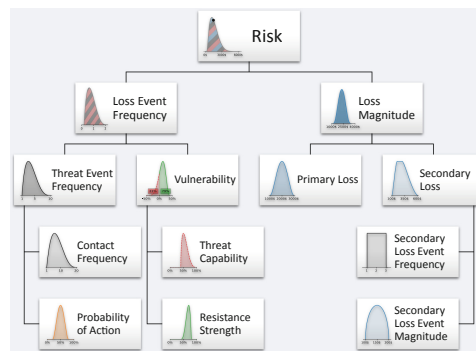


Fig. 1. The FAIR ontology with its subfactors (reused from [9]).

politely guide you to the standards Risk Taxonomy [32] and Risk Analysis [31] by the Open Group, or the respective book by the creators Jones and Freund [12].

3 Contribution: The RISKEE Method

RISKEE (coined from Risk-Tree) is a method for risk assessment and evaluation, which is based on attack-trees/attack-graphs with special emphasis on risk. It works by building a graph of consecutive attack events, judging the frequencies, vulnerabilities and impact magnitudes of events, and calculating a distribution of risks based on that. The initial attack frequencies are carried forward over the node's vulnerabilities until the end nodes. There, the losses are realized and propagated backward again to calculate the respective *virtual risk* for all individual nodes (Fig. 2).

The attack graph depicts different attack paths an attacker must take to reach some goal. The edges define the consecutive order of these events, one step after another. Such attack paths may intersect and split up again when attackers have several possibilities to choose from. Nodes have some necessary attributes which must be defined, to make risk evaluation possible. These values are frequency, vulnerability and magnitude, and can be defined via historical data or expert judgement. Important here is, that they should be given as distribution and ranges which consider the respective uncertainty, not only as single values. The first nodes on an attack path form the attack surface, which is subject to a permanent bombardment of attacks. Here the frequency of attacks and the respective vulnerability (probability of an attack going through) have to be defined. The attack continues then with the intermediate nodes, which only need a vulnerability rating, but already may involve impact. In the end, the last nodes represent the actual goal, involving the actual losses. When an attacker reaches them, the loss is realised.

3.1 Structure and Framework

A risk tree consists of three different types of nodes which are connected and form the individual attack paths: entry events, intermediate events, and goal events. The events are described using the attributes frequency, vulnerability, and impact.

Types of Events. All events in RISKEE represent attack events which form a path to a specific goal for the adversary. In the simplest case this is only one

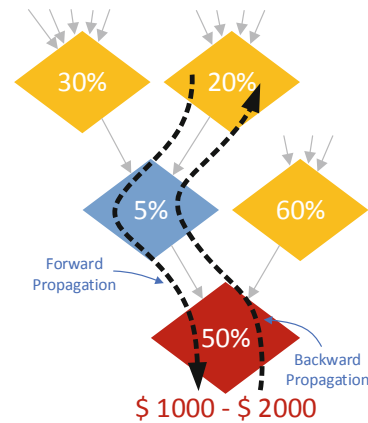


Fig. 2. A risk-tree showing forward propagation of attacks and backward propagation of risk in RISKEE.

event representing attack surface and goal, but in real cases the attack graph consists of multiple events and paths.

- **Entry Events** represent the attack surface. Every attack starts with these nodes (regardless if it is an external or internal attack). They are also the only ones defining the frequency of attacks.
- **Intermediate Events** are events which have to be passed through in order to get to the goal events. To achieve their goal, adversaries have to go through these events. Intermediate events are described mainly by the vulnerability.
- **Goal Events** are events on the end of an attack path, which cause the most loss or harm to the victim. The goal events represent the last event of an attack path (a so called “sink”). The most important attribute for a goal event is the impact magnitude.

Types of Attributes. The events are described using three attributes: *frequency*, *vulnerability*, and *impact*, which are used to calculate the resulting *risk*:

- **Frequency:** The estimation of the number of events over a specific time span (e.g. 4–6 times per year, or 80–100 times a day).
- **Vulnerability:** Is the probability that a threat event will become a loss event. Or stated in other words: that an attack is successful. The vulnerability depicts the difference in strength, between the adversary and victim - like pulling on a rope from two sides. It is the difference of the respective estimations for the Threat Capability (adversary-side) and Resistance Strength (victim-side). To be comparable, both estimates must have the same scale within the event which gets estimated. A good proposal is to use the distribution of overall threat population for the scope of the event as a scale.
 - *Resistance Strength* is the rating of the defender. It defines how well the analysed system is protected against attacks. The scale for this should be the overall threat population to evaluate which portion of attacks the system can withstand.
 - *Threat Capability* is the rating of the attacker. It determines the capabilities, resources and knowledge of the assumed attackers compared against the overall threat population.
- **Impact:** The range of impacts an event can have, when it actually occurs (e.g. \$1000–\$2000, but most likely around \$1100).
- **Risk:** Represents a whole range of estimated outcomes for future events, together with their likelihoods. This is not a single value, but a distribution over probable outcomes. It is the result of the calculations and is most accurately visualized with a loss exceedance curve (LEC).

3.2 Using RISKEE

RISKEE can be embedded into the development process already very early on, at architectural phase or at design phase and can be refined during development as well as later on during runtime. We propose to use the Diamond model [4] as

a framework to define the attributes in order to judge frequency, vulnerability, and magnitude later on. While it would also be possible to judge them directly, it would not be as comprehensible as using a rigorous formal model like the Diamond model. If data is available from existing methods like SAHARA or FMVEA, it can be used to fill the Diamond attributes [9]. We use expert judgements to determine the attributes. The expert group should be composed of three to five people [10] from different domains [5] e.g. cyber security experts, infrastructure experts, domain/field experts, system analysts, or system architects. The judgements are combined via a linear opinion pool (arithmetic average) [29]. A risk tree is created by identifying the necessary steps for an attacker to achieve a goal and attributing them with frequency (how often does this attack occur), vulnerability¹ (how likely is the attack to be successful), and magnitude (what is the impact of an successful attack event). The nodes represent attack events and the edges resemble the order in which they can occur. The result is calculated via applying the RISKEE PROPAGATION ALGORITHM (Algorithm 1) and cumulated via a so called Loss-Exceedance-Curve (LEC). This curve depicts the probability of exceeding certain amounts of money over the whole range. With the LEC as basis, management can judge if the possible risks are tolerable, or still to high (which is called the risk appetite). Figure 3(b) in Sect. 3.5 shows an example for a loss-exceedance curve.

3.3 Process

To give a step-by-step guidance, here the complete process using the RISKEE method is described:

- Step 1: Create the attack graph.
- Step 2: Estimate the attributes for the events:
 - Entry Events: **Frequency**, (*Vulnerability, Magnitude*)
 - Intermediate Events: **Vulnerability**, (*Magnitude*)
 - Goal Events: **Magnitude**, (*Vulnerability*)
- Step 3: Calculate risk with the RISKEE Propagation Algorithm (see Algorithm 1).
- Step 4: Make decision based on the loss exceedance curve for the risk or enact further mitigation steps to reduce risk.

3.4 Calculation of Risk

The calculation of risk is done with the RISKEE PROPAGATION ALGORITHM (see Algorithm 1). All operations are done with probability distributions coming from the factor analysis of information risk FAIR analysis. It basically consists of the following steps:

¹ Vulnerability can also be judged indirectly in form of resistance strength and threat capability. These two estimations are subtracted by RISKEE to get the percentage of cases where an attack would be successful.

1. Split up the risk-graph into all distinct individual paths from all entry nodes to nodes with a defined loss magnitude (mostly goal nodes, but others could also have a defined magnitude).
2. For every path: take the attack frequencies of the input node and propagate it forward over the intermediate nodes (multiplied with the respective vulnerability) until the goal node is reached. Also accumulate any impacts on the way.
3. Calculate cumulative Risk in the goal node by multiplying the resulting frequency with the impact.
4. Apply this risk to all nodes and edges on the current path (sum up if they already contain existing risk-values).

Algorithm 1. RISKEE PROPAGATION ALGORITHM

```

1: procedure RISKEEPROPAGATE( $G$ ) ▷  $G \dots$  Risk Graph
2:   for all  $path \in Paths(G)$  do
3:      $frequency \leftarrow frequency_{entry}$ 
4:      $magnitude \leftarrow 0$ 
5:     for all  $node \in Nodes(path)$  do ▷ Propagate Forward
6:        $frequency \leftarrow frequency * vulnerability_{node}$ 
7:        $magnitude \leftarrow magnitude + magnitude_{node}$ 
8:     end for
9:      $risk \leftarrow frequency * magnitude$ 
10:    for all  $edge \in Edges(path)$  do ▷ Propagate Backwards to Edges
11:       $risk_{edge} \leftarrow risk_{edge} + risk$ 
12:    end for
13:    for all  $node \in Nodes(path)$  do ▷ Propagate Backwards to Nodes
14:       $risk_{node} \leftarrow risk_{node} + risk$ 
15:    end for
16:  end for
17: end procedure

```

3.5 Computations on Probability Distributions

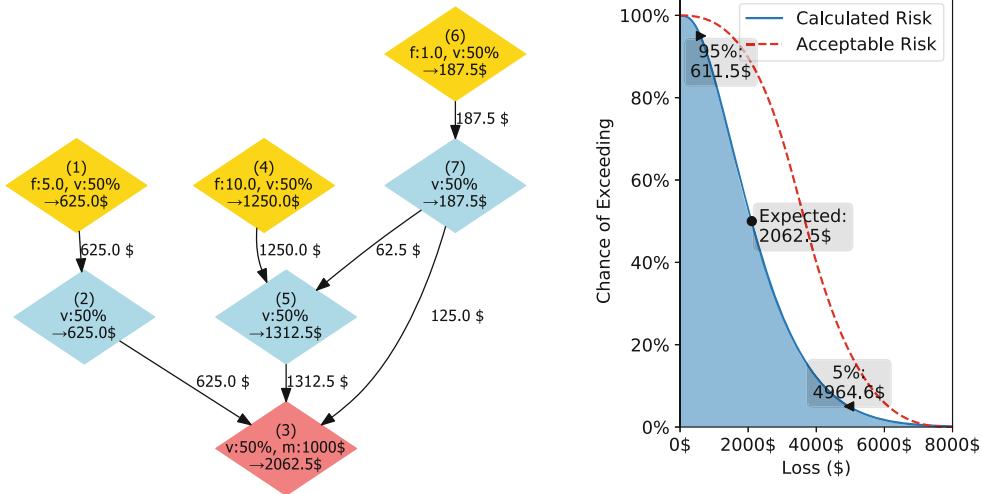
The RISKEE PROPAGATION ALGORITHM computes values based on probability distributions. We use monte-carlo simulation to calculate the mathematical operations on the probability distribution. This works by sampling many values from the distributions, executing the operations and in the end create a histogram over the results. We use PPS-sampling² via inverse-transformations of the cumulative probability density functions for unrelated values, and for related or conditional distributions we use simple random sampling. In both cases the calculation is finished by computing the histogram over the results and smoothing it with bounded kernel density estimation using gaussian kernels to get a continuous probability distribution function for further usage. We apply the smoothing to

² Probability-Proportional-Size-Sampling; A stratified sampling strategy.

avoid aggregation of quantification errors. In this way we can assume to have continuous functions for every further operation. For the PPS-sampling we use a fixed number of percentiles over the distributions (2500 percentiles), and for the random sampling we use an adaptive algorithm which stops on convergence ($\varepsilon < 0.05\%$), which happens mostly after about 20000 to 50000 samples.

Example: Here we feature an example for applying RISKEE on a simple attack-tree together with judgements of the input values and presenting the resulting risk. Figure 3 shows the risk tree and the result. In this example we showcase an attack tree consisting of seven events (enumerated from (1) to (7)). For easier demonstration it uses 50% vulnerability for all events, and only has one goal event with the magnitude of 1000\$. The entry events were event (1) with a frequency of 5 times/year, event (4) with a frequency of 10 times/year, and event (6) which occurs once per year. The resulting paths in this example graph are as follows:

- $5 \times [(1) \rightarrow (2) \rightarrow (3)] \times 1000\$,$ resulting in an expected risk of 625\$
- $10 \times [(4) \rightarrow (5) \rightarrow (3)] \times 1000\$,$ resulting in an expected risk of 1250\$
- $1 \times [(6) \rightarrow (7) \rightarrow (3)] \times 1000\$,$ resulting in an expected risk of 62.5\$
- $1 \times [(6) \rightarrow (7) \rightarrow (5) \rightarrow (3)] \times 1000\$,$ resulting in an expected risk of 125\$.



(a) An example for a risk tree, showing the different possible attack paths (f=frequency, v=vulnerability, m=magnitude, the resulting risk is displayed after the \rightarrow). (b) An example for a loss exceedance curve (LEC). It shows the probability of exceeding certain magnitudes of losses.

Fig. 3. Applied example for a risk assessment with RISKEE.

Summed up, the resulting expected risk 2062.5\$. By using RISKEE utilizing the power of probability distributions we gain even more knowledge than the

expected risk: We get the whole possibility space which can be presented as loss exceedance curve, shown in Fig. 3(b). It shows the probability of exceeding certain amounts of losses. To evaluate this, an overlay for the risk appetite is added which represent the acceptable risk. In this example we defined it as follows: with 70% chance we can afford to lose 3000\$, with 50% chance we accept loosing 4000\$, and with 10% we still can tolerate to loose 6000\$. By interpolating and smoothing between these fixed points we get a curve which can be easily compared to the calculated risks. In this example the risk curve is below the acceptable risk, therefore we can accept it.

The advantages of this approach are visible in Fig. 3(b). RISKEE delivers not only an expected value, but much more information in the form of the distribution of all possible outcomes. For example, the resulting graph states that the expected range of outcomes will be between around 600\$ and 5000\$ with 90% confidence (range between the 5% percentile and 95% percentile). Furthermore, we can see what the extreme cases even go well beyond 6000\$ (up until approximately 12400\$, but the graph is cut off due to space saving reasons in this paper).

3.6 Threats to Validity and Limitations

In cyber-security we often have to deal with rare but catastrophic events, which cannot easily be judged and predicted. Therefore estimations of vulnerability could be off by magnitudes. Also the hacks are often so focused to one specific combination of technologies that experts have really quite some difficulties of predicting them. If there is a obvious hole in the protection line, it has to be protected anyways, therefore the risk assessment is most useful for the hard to estimate events which are unknown and infrequent. Nevertheless, since the method can also model other types of risk, it is not limited to the specific field of cyber-security, but could also be applied to other fields where the vulnerability can be judged in a more reliable way. In the current form the method only supports the modelling of monetary loss, but other types of loss metrics can easily be added in the future. Regarding Scaling: The method does scale very badly in its current form. Adding more nodes increases the calculation time manifolds. This limitation will be tackled in future research via usage of dynamic programming, and stochastic optimizations. Currently we have no defined file format for export and import of data, and no bindings to other languages. These are features which are on the agenda for future work in order to make RISKEE compatible and usable from multiple locations and environments.

4 Conclusion and Future Work

In this paper we took a step further into the direction of unified integrated quantitative risk assessment for safety and security. We connected to our previous work on this topic and mentioned some of the limitations established methods have. The contribution of this paper is the RISKEE method which is a method

for risk assessment based on attack graphs and probability distributions. We described how to create such a graph and what the needed input values for determining the risk are. Furthermore, we showed the RISKEE PROPAGATION ALGORITHM to calculate risk by forward propagation of frequencies and backward propagation of risk. Finally, we discussed some aspects of computation with probability distributions, which we will follow on in future work. Also, in future work we want to investigate on mitigation possibilities and strategies, as well as enabling future predictions of risk by applying a decay on resistance over time and modelling dynamic evolving attackers.

References

1. Ammann, P., Wijesekera, D., Kaushik, S.: Scalable, graph-based network vulnerability analysis. In: Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS 2002, p. 217. ACM Press, Washington, DC (2002). <https://doi.org/10.1145/586110.586140>
2. Braband, J.: Risk assessment: as simple as possible (and no simpler). In: 1st IET International Conference on System Safety, vol. 2006, pp. 285–300. IEE, London (2006). <https://doi.org/10.1049/cp:20060229>
3. Braband, J.: Definition and analysis of a new risk priority number concept. In: Spitzer, C., Schmocker, U., Dang, V.N. (eds.) Probabilistic Safety Assessment and Management, pp. 2006–2011. Springer, London (2004). https://doi.org/10.1007/978-0-85729-410-4_322
4. Caltagirone, S., Pendergast, A., Betz, C.: The diamond model of intrusion analysis. Technical report. Center for Cyber Intelligence Analysis and Threat Research (2013)
5. Clemen, R.T., Winkler, R.L.: Combining probability distributions from experts in risk analysis. *Risk Anal.* **19**(2), 187–203 (1999)
6. Cox, A.L.: What’s wrong with risk matrices? *Risk Anal.* **28**(2), 497–512 (2008). <https://doi.org/10.1111/j.1539-6924.2008.01030.x>
7. Cox, A.L., Babayev, D., Huber, W.: Some limitations of qualitative risk rating systems. *Risk Anal.* **25**(3), 651–662 (2005)
8. Cox, A.L., Popken, D.A.: Some limitations of aggregate exposure metrics. *Risk Anal.* **27**(2), 439–445 (2007)
9. Dobaj, J., Schmittner, C., Krisper, M., Macher, G.: Towards quantitative integrated security and safety risk assessment. In: Proceedings of the DECSOS Workshop for SAFECOMP 2019 Conference on Computer Safety, Reliability, and Security, p. 14 (2018)
10. Ferrell, W.R.: Combining individual judgments. In: Wright, G. (ed.) Behavioral Decision Making, pp. 111–145. Springer, Boston (1985). https://doi.org/10.1007/978-1-4613-2391-4_6
11. Franklin, B.D., Shebl, N.A., Barber, N.: Failure mode and effects analysis: too little for too much? *BMJ Qual. Saf.* **21**(7), 607–611 (2012). <https://doi.org/10.1136/bmjqs-2011-000723>
12. Freund, J.: Measuring and Managing Information Risk: A FAIR Approach. Butterworth-Heinemann, Oxford (2015)
13. Huang, M.S.: An approach for improvement of risk priority number in FMEA. *DEStech Transactions on Computer Science and Engineering (itme)*, April 2017. <https://doi.org/10.12783/dtcse/itme2017/8010>

14. Hubbard, D.W.: The Failure of Risk Management: Why It's Broken and How to Fix It. Wiley, Hoboken (2009). oCLC: ocn268790760
15. IEC: IEC 60812: Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA) (2006)
16. ISO: ISO 26262 Road vehicles - Functional safety (2011)
17. Krisper, M., Dobaj, J., Macher, G.: Pitfalls, fallacies, and other problems in risk matrices using ordinal scales. Currently Under Review, p. 11 (2019, Submitted)
18. Macher, G., et al.: Integration of security in the development lifecycle of dependable automotive CPS (2017)
19. Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: SAHARA: a security-aware hazard and risk analysis method. In: Design, Automation & Test in Europe Conference & Exhibition (2015)
20. Messnarz, R., Sporer, H.: Functional safety case with FTA and FMEDA consistency approach. In: Larrucea, X., Santamaria, I., O'Connor, R.V., Messnarz, R. (eds.) EuroSPI 2018. CCIS, vol. 896, pp. 387–397. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-97925-0_32
21. Metropolis, N., Ulam, S.: The Monte Carlo method. *J. Am. Stat. Assoc.* **44**(247), 335–341 (1949). <https://doi.org/10.1080/01621459.1949.10483310>
22. Microsoft Corporation: Threat Modeling Tool. <https://docs.microsoft.com/en-us/azure/security/azure-security-threat-modeling-tool-threats>
23. Poolsappasit, N., Dewri, R., Ray, I.: Dynamic security risk management using Bayesian attack graphs. *IEEE Trans. Dependable Secure Comput.* **9**(1), 61–74 (2012). <https://doi.org/10.1109/TDSC.2011.34>
24. RiskAMP: The beta-PERT Distribution—RiskAMP (2010). <https://www.riskamp.com/beta-pert>
25. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (FMEA). In: Bondavalli, A., Di Giandomenico, F. (eds.) SAFECOMP 2014. LNCS, vol. 8666, pp. 310–325. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10506-2_21
26. Schneier, B.: Attack trees - modeling security threats. *Dr. Dobb's J.* **24**(12), 21–29 (1999). <http://www.schneier.com/attacktrees.pdf>
27. Sellappan, N., Palanikumar, K.: Modified prioritization methodology for risk priority number in failure mode and effects. *Analysis* **3**(4), 10 (2013)
28. Stevens, S.S.: On the Theory of Scales of Measurement (1946)
29. Stone, M.: The opinion pool. In: *Annals of Mathematical Statistics* (1961)
30. The Open Group: FAIR - ISO/IEC 27005 cookbook (c103) (2010)
31. The Open Group: Risk Analysis (O-RA), October 2013. <https://publications.opengroup.org/c13g>
32. The Open Group: Risk Taxonomy (O-RT) 2.0, October 2013. <https://publications.opengroup.org/c13k>
33. Thomas, P., Bratvold, R.B., Bickel, E.: The risk of using risk matrices. In: SPE Annual Technical Conference and Exhibition. Society of Petroleum Engineers, New Orleans, Louisiana, USA (2013). <https://doi.org/10.2118/166269-MS>
34. Tidwell, T., Larson, R., Fitch, K., Hale, J.: Modeling internet attacks. In: *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security* (2001)
35. Vesely, W.E., Goldberg, F.F., Roberts, N.H., Haasl, D.F.: *Fault Tree Handbook (NUREG-0492)*. U.S. Nuclear Regulatory Commission (1981)
36. Vose, D.: *Risk Analysis: A Quantitative Guide*, 3rd edn. Wiley, Hoboken (2008)
37. Xing, L., Amari, S.V.: Fault tree analysis. In: Misra, K.B. (ed.) *Handbook of Performability Engineering*, pp. 595–620. Springer, London (2008). https://doi.org/10.1007/978-1-84800-131-2_38

A.6 [P6] Patterns for Communicating Numerical Uncertainty

- Authors: Michael Krisper, Jürgen Dobaj, Georg Macher
- Year: 2019
- DOI: 10.1145/3361149.3361160
- Bibliography-Reference: [185]
- Presented and Discussed at EuroPLoP'19
- Published in EuroPLoP'19 conference proceedings (ACM, ICPS)

Summary This paper explores different ways how to communicate uncertainties to humans. This is described in pattern form, which includes the context, the problem (including the guiding forces), the solution idea, and finally, the consequences of applying the patterns. The problem with uncertainty communication and decision-making with uncertainty is that humans are biased, especially when it comes to probabilities and high uncertainty like in risk assessments. To avoid that, patterns in three categories of communication are discussed: graphically, numerically, and textual. The graphical category is about visualizing uncertainties on graphs and diagrams. The numerical category is about presenting numbers in a way that also includes the respective uncertainty. Textual representations of uncertainty are about descriptive words which give away a notion of the underlying uncertainty. While graphical and numerical categories contain patterns, the textual category contains antipatterns which should be avoided.

- **Patterns for graphical visualization of uncertainty:**
 - **Error Indicator:** Graphical indications or symbols which represent the uncertainty.
 - **Distribution Plots:** Show the distribution of possibilities around each drawn data value.
- **A Pattern for numerical representation of uncertainty:**
 - **Numbers with Uncertainties:** State the uncertainty in the form of, e.g., the standard deviation in addition to the numerical value. Use a special syntax for this which the reader can understand easily, e.g., using the \pm symbol or using parentheses.
- **(Anti-)Patterns for textual descriptions of uncertainty:**
 - **Words of Estimative Probability:** Words like *often*, *certainly*, or *unlikely*, are not suitable for stating the uncertainty because the interpretation depends on the context and the individual bias and cultural background of the reader.
 - **Numeric Hedge Words:** Words like “*almost x*”, “*at most x*”, “*about x*”, are more concrete than words of estimative probability, but still leave a room for interpretation which is bad and should therefore be avoided.
 - **Quantitative Comparisons:** Comparing probabilities to real events that are known to most of us, e.g., throwing dice.

My Contribution This paper was completely written and conceived by me. The contribution of the other authors was in the form of discussions, feedback for improvements, and corrigenda. Not mentioned as authors but also influencing the paper were the shepherd Lise Hrvatum and the workshop participants who gave feedback during the conference.

Copyright The version included in this dissertation is a permitted reprint of the definitive version, which can be accessed via the ACM Digital Library here:
<https://dl.acm.org/doi/10.1145/3361149.3361160>

Patterns for Communicating Numerical Uncertainty

Michael Krisper

michael.krisper@tugraz.at
Institute for Technical Informatics
Graz University of Technology
Graz, Austria

Jürgen Dobaj

juergen.dobaj@tugraz.at
Institute for Technical Informatics
Graz University of Technology
Graz, Austria

Georg Macher

georg.macher@tugraz.at
Institute for Technical Informatics
Graz University of Technology
Graz, Austria



ABSTRACT

Uncertainty is an inherent property of all measurements, statistics, or generally all communication involving numbers. Whenever numerical data is communicated, the uncertainty or confidence in this data should also be included. Neglecting it, or communicating it ambiguously, leads to misinterpretation and misunderstandings. There are some well-known and proven patterns to avoid such problems. In this paper, we present a collection of patterns for the communication of numerical uncertainty. These patterns revolve around three areas of applications: textual, numerical, and graphical. For numerical representations the pattern **NUMBERS WITH UNCERTAINTIES** is shown. For textual descriptions **WORDS OF ESTIMATIVE PROBABILITY**, **NUMERIC HEDGE WORDS** and **QUANTITATIVE COMPARISONS** are explained, and for graphical visualization **ERROR INDICATOR** and **DISTRIBUTION PLOTS** are described. The paper is targeted towards communicators, visualizers, reporters, as well as developers, engineers, and researchers of solutions for problems which involve uncertainty.

CCS CONCEPTS

• **Human-centered computing** → **Visualization application domains**; • **Computing methodologies** → **Uncertainty quantification**; • **Software and its engineering** → **Design patterns**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EuroPLOP '19, July 3–7, 2019, Irsee, Germany

© 2019 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 978-1-4503-6206-1/19/07...\$15.00

<https://doi.org/10.1145/3361149.3361160>

KEYWORDS

communication, uncertainty, error, ambiguity, confidence, visualization, probability

ACM Reference Format:

Michael Krisper, Jürgen Dobaj, and Georg Macher. 2019. Patterns for Communicating Numerical Uncertainty. In *24th European Conference on Pattern Languages of Programs (EuroPLOP '19)*, July 3–7, 2019, Irsee, Germany. ACM, New York, NY, USA, 15 pages. <https://doi.org/10.1145/3361149.3361160>

1 INTRODUCTION

Uncertainty and risk play a huge role in our everyday life. Every decision and judgment is accompanied by a degree of uncertainty and followed by a risk assessment for an estimation of the probable consequences. We are so accustomed to the uncertainty that we usually do not even recognize it explicitly, but have a well-trained gut feeling about how risky or uncertain a situation is. This applies especially to everyday situations where we already have much experience, but now and then, we stumble upon situations where our gut feeling is wrong and biased [23]. Things like making future predictions and estimations, statistics, and probabilities are tough for humans to grasp. Furthermore, we are highly susceptible to biased communication with exaggerations, understatements, or fake news, which are conveying wrong information and neglecting uncertainty and talking risks down or extremely overemphasize them. The problem of communicating information from a sender to a receiver is challenging and induces many problems. This may be intentional or non-intentional and can have severe effects.

1.1 Sender versus Receiver of Information

The general problem of communication is how information and messages are transmitted between a sender and a receiver. Does the receiver understand the intended message, or is the information somehow distorted and misinterpreted? Due to different cultural

backgrounds and human bias, there can be no perfect information channel in this regard, but at least we can try to avoid common errors and apply patterns and principles which have proven to make communication more understandable and informative.

To give an illustrative example: In natural language, we have phrases and words like “very common”, “with high confidence”, or “highly likely”, “probably” and so on. These phrases obfuscate the actual uncertainty of the data [9, 48]. The meaning of such words of estimative probability is highly subjective and can lead to misinterpretations. Just think of the term “very often” - what number of events does this indicate? Is it already very often, when you check your emails ten times per day? For some people, it is not, but for many, it is. Now, think of the same number in another context: Is it very often if you shower ten times a day? For the majority of people, it is. Depending on the context and the personal and cultural background, words of estimative probability could have a completely different meaning.

1.2 Uncertainty, Ambiguity, Ignorance, Error

In measurement theory, there is a distinction between aleatoric and epistemic uncertainty. Aleatoric uncertainty is the statistical uncertainty that is unavoidable, but at least is known and can be modeled and calculated. Epistemic is the systematic uncertainty that is not known or cannot be fully modeled [26, 28]. These two relate to models and numbers, but there is also a third variant of uncertainty: Ambiguity. Especially in natural languages (spoken or written), the meaning of words is very ambiguous and can differ greatly depending on the context. Also, visualization can be ambiguous and misleading e.g., when the visual representations do not correlate to the underlying numbers or when they are perspectively distorted. Tufte called this the “lie factor” in visualizations [43]:

$$\text{Lie Factor} = \frac{\text{size of effect shown in graphic}}{\text{size of effect in data}} \quad (1)$$

A “good” lie factor is between 1.05 and 0.95 - outside of this range, the visualization differs too much from the data - it “lies” [43].

There is, of course, another cause of uncertainty: errors and faults. The reasons are e.g., mishaps, wrong assumptions, and models, or faulty executions. Humans make errors, and even the most accurate representation of uncertainty is worthless if it is faulty or the model is entirely wrong.

1.3 General Problem Statement

Think of the following problem situations which may occur in your daily life: You have a specific idea in mind and data supporting it, and you want to communicate it to your colleagues and friends. First, you try to describe it to them using words. In spoken and written language, we tend to use hedge-words like “good”, “high certainty”, “low confidence”, “very often” and these may convey a completely wrong message because of their vagueness. So you try to use numbers, but they are difficult to interpret - only presenting averages may be easier but hides away your uncertainty. Numbers often are understood as facts with very high confidence. Finally, you try to use diagrams and graphs to demonstrate trends and comparisons, but visualize the uncertainty wrong, tricking others into thinking that the data is much more precise than it is. All three

ways (textual, numerical, and graphical) have some pitfalls. In this paper, we show some of these pitfalls and some patterns to avoid them.

1.4 Structure

In the remainder of this paper, we will give some motivational examples, and afterward, we present six patterns, including some variants grouped into three areas of communication - graphical, numeric, and textual representations:

- **Graphical Representations:** Representing information in a graphical way as a diagram, or visualization. Use this if you have large amounts of data that you want to present concisely and intuitively. This should always be combined with textual or numeric representations. Patterns:
 - **ERROR INDICATOR:** A pattern for showing the uncertainty in data visually.
 - **DISTRIBUTION PLOTS:** A pattern for visualizing the distribution of numerical data.
- **Numerical Representations:** Representing information in the form of numbers like data values, data tables, measurements, or descriptive statistics. Use this as support for the written argument in textual representations. Also, use this if you want your communication as precise and unambiguous as possible. Patterns:
 - **NUMBERS WITH UNCERTAINTIES:** A pattern for stating the actual numbers with their respective uncertainties appropriately and reasonably.
- **Textual Representations:** Representing information in text form as e.g., an article or in a blog post. Use this if you want to build up a written argument based on some data. Patterns:
 - **WORDS OF ESTIMATIVE PROBABILITY** (Anti-Pattern). Words for probabilities or frequencies which are ambiguous and highly subjective, e.g., often, likely, doubtful, probably.
 - **NUMERIC HEDGE WORDS** (Anti-Pattern). Words which emphasize, understate, or change the importance of some statement e.g., highly, very, significantly, hardly.
 - **QUANTITATIVE COMPARISONS.** Paraphrased substitutes for actual probabilities and frequencies, which are not ambiguous, but still intuitively understandable, like e.g., 9 out of 10 people, on every 3rd usage, once a week.

The patterns are described using canonical pattern structure: name, summary, context, problem, forces, solution, consequences, and known-uses. The solution also includes variants and examples, and the consequences, benefits, drawbacks, and liabilities of the pattern are discussed. For the anti-patterns, a much simpler form was chosen, just consisting of the problem description and the consequences.

2 MOTIVATION

In the following sections, some examples of real-life situations will be explained to give motivation for the paper.

Media and Data Journalism. Journalists present statistics that are measured via polls, surveys, or is coming from the public census.

How this data is presented may strongly influence public opinion. Especially in elections this plays a role, when presenting e.g. election forecasts or voter sentiment analysis [8, 12]. A recent example of this is the presidential election in the USA in 2016 between Trump and Clinton. Here, many predictions were presented indicating that Clinton would win, but the outcome was the complete opposite, as we all know now. Advertisement and news bubbles influenced people on Facebook and vague predictions on TV and newspapers [12, 33]. Showing the uncertainty in the data would have helped the people recognizing that the predictions are not that given, and a win is still possible for both sides. The wish for showing precise data, high confidence, and probabilities deceived people in thinking that the result is already fixed. The wish for certainty in data is a bait that media and journalists exploit very often in the form of catchy headlines. Every time a new cure for cancer, Alzheimers, or obesity is found and presented in the media, this should be taken with care and a high sense of realism. Media is hiding behind pompous and exaggerated headlines and click-baits in order to get attention, clicks, and likes from people.

2.0.1 US Election 2016: Trump vs. Clinton. In 2016, just before the presidential elections in the US, media and newspapers made several user surveys about the outcome and tried to predict the results. Many of these predictions had high statistical uncertainty, but this was not appropriately communicated to the general public. Only a few reporters like e.g., Nate Silver, showed the predictions with their actual uncertainty [12, 33].

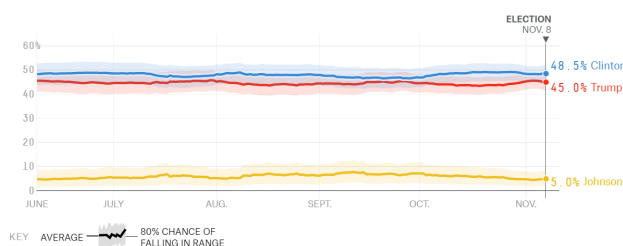


Figure 1: The prediction of election results by the FiveThirtyEight 2016 Election Forecast. It can be seen that the result is not very certain and could turn out either way. Figure taken from [39].

2.0.2 Catalan independence in Spain. For many years now, there is a debate about whether Catalonia should be an independent country or not. There are regular user surveys about this, and again the voters are very indecisive, fluctuating between leaving or staying with Spain. Every time the sentiment changes, media is reporting this as a catchy headline, but actually, the uncertainty is quite huge in these surveys. Figure 2 shows this fluctuations. A famous example was back in 2014 when, for the first time in many years, more people wanted to stay with Spain than leave. 45.3% of the people wanted to stay, 44.5% wanted to leave. The Spanish newspaper El Pais reported this as “Catalan public opinion swings towards ‘no’ for independence, says survey” [38]. The problem was that the survey had only a small sample size of 1100 people, and therefore the statistical uncertainty was quite large ($\pm 2.95\%$) [4].

With such high uncertainty, the 0.8% difference between the groups was not significant, and therefore it can not be concluded that there is any “swing towards” one side or the other - the survey results could just have happened due to statistical randomness. Still it caused quite an upheaval amongst the people and politicians.

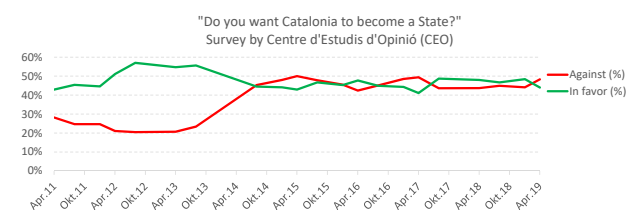


Figure 2: The voter sentiments for Catalan independence over the last few years. From 2014 onward there is no clear majority anymore, and the uncertainty is not depicted - not a good basis for reasoning. Survey was done by the Catalan Center for Opinion Studies (CEO) [6], Graph is redrawn from data by [49].

Brexit Referendum 2016. The UK held a referendum in 2016 on whether to leave the European Union. In this referendum, 52% of the voters decided to leave the EU, but this result was heavily debated due to bias and distortion of the votes. In total, it was a very close call for such an influential decision of a country. Additionally, the media reported the consequences in a very biased way, which led people to believe that leaving may be better for the country and that the consequences are not that severe [36]. Figure 3 shows the sentiment over the years.

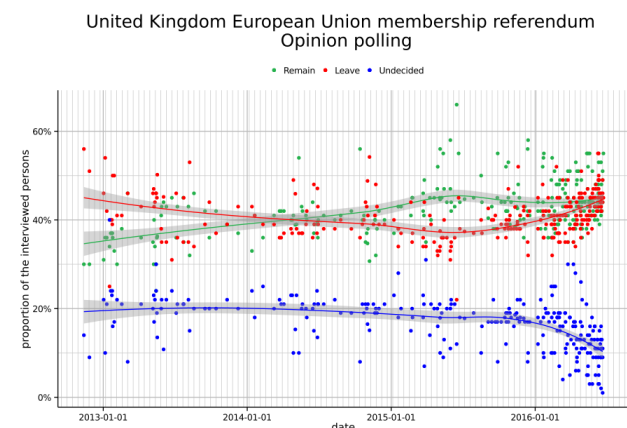


Figure 3: The UK Brexit sentiments over time. This graph shows the individual samples, as well as the average and the respective uncertainty. By seeing this, the reader can decide how much the data can be trusted, and if important decisions for a whole country should be made based on that. Figure taken from [42].

2.1 Weather Forecasts and Climate Reports

Have you ever wondered why predictions by the local weather forecast are so often wrong or inaccurate? For a few days into the future, they are quite correct, but looking at the 14-day forecast is just a blurry vision. Why is it that we still trust the weather forecast, although they are wrong so often? The problem here is: Science is often more inaccurate than we want to believe. People do not want to see vague, uncertain, and doubtful data (called bias of uncertainty aversion or risk aversion) [4, 23]. That may be the reason why TV broadcasters eschew the uncertainty and present it to their best effort knowledge. In 2018 BBC admitted to “get climate change coverage wrong too often” and send out guidance to all authors how to report the climate more accurately [5, 15]. Alberto Cairo pointed out that it is rather difficult to draw a weather forecast, which includes all uncertainty but still is trustworthy enough [4]. Figure 5 shows some examples by Cairo, and Figure 6 shows a real-life example of a hurricane forecast for the hurricane Lorenzo in September 2019. The NOAA wrote the following about Lorenzo: “After Wednesday, Lorenzo will likely continue north, moving west of the British Isles and Ireland towards Iceland. However, the potential track beyond Thursday remains uncertain.” [34], showing again how uncertain our predictions still are.

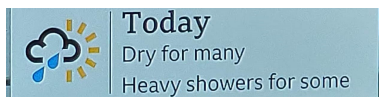


Figure 4: The weather forecast for 29. August 2019 in the Argyll and Bute Area (Scotland), demonstrating very high uncertainty.



(a) A common, but deceitful version of visualizing the path of a hurricane. The error indicator shows just the 66% confidence interval.

(b) A very deceitful variant of 5(a) which uses a hurricane pictograph, conceiving the reader to think that the hurricane only moves along the shown path but gets bigger.



(c) A more accurate version, which shows different paths the hurricane could take and indicating the confidence visually with transparency and line thickness.

(d) More realistic variant of 5(c) showing possible paths and sizes of the hurricane using pictographs and transparency, but is a little bit overloaded and might induce severe fears in people.

Figure 5: Examples of visualizations for the path of a hurricane. Illustrations by Alberto Cairo, taken from [4].



Figure 6: Real Life Example: 5 day prediction for the probability of winds in tropical-storm-strength along the path of hurricane Lorenzo on 29. September 2019 by the National Oceanic and Atmospheric Administration (NOAA) [34]. Should Ireland be evacuated or not?

3 GRAPHICAL REPRESENTATIONS OF UNCERTAINTY

One picture is worth a thousand words.

F. R. Barnard, 1921

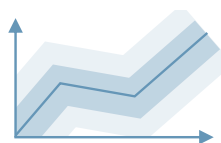
Graphical representations of data in form of diagrams, graphs, charts, and infographics are common in all kinds of media. People understand and digest graphical information much faster, easier and with less effort than written representations or tables with numbers. This makes it ideal to present huge amounts of data points, or to make relations and proportions clear in a quite intuitive way for the reader. Anyhow, this was not always the case. Before the golden age of information visualization in 1820 [10], the majority of people was not able to read graphs and diagrams because they simply never have seen one and didn't learn how to interpret them. Fortunately, nowadays the majority is trained in graphical literacy and accustomed to read diagrams like barcharts, linecharts, and other graphical representations of data. Still, it is a difficult task to design visualizations in an aesthetic and informative way, which reaches the reader. For further reference we lead the reader to some of the well known books about information visualization by Colin Ware [47], Alberto Cairo [3] and Edward Tufte [43].

In this section we present the following patterns:

- **ERROR INDICATOR:** A pattern for showing the uncertainty in data visually.
- **DISTRIBUTION PLOTS:** Patterns which visualize the distribution of numerical data.

All patterns in this section fall under the same context of usage: We have data which should be communicated to a human in a graphical way in order to convey some message. The exact shape and kind of data varies, and the data may contain varying degrees of uncertainty. Typically this is done with graphs and diagrams, visualizing the underlying data points. The presentation media is mostly restricted to 2d-surfaces (e.g. computer screens, smartphones, images, or printed paper).

3.1 Pattern: ERROR INDICATOR



An *ERROR INDICATOR* shows the uncertainty of the data visually with an overlay of lines or areas, which represent the range of probable values around the average.

3.1.1 Context. We have data that is arranged along a continuous ratio axis and has some uncertainty. We want to communicate this data to the reader visually.

3.1.2 Problem. The data we want to show contains uncertainty and errors, and we want to convey this somehow to the reader. Only showing the average values conveys a wrong image of certainty, which could lead to wrong decisions.

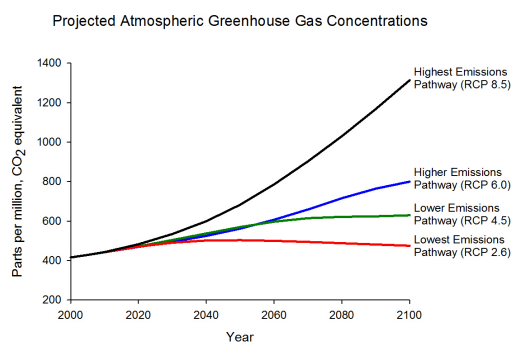


Figure 7: Problem: This graph shows predictions of CO₂ in the atmosphere. Drawing the data only by using lines make it seem very precise, but that is not true. Especially in climate forecasts there is huge uncertainty. Figure taken from [46].

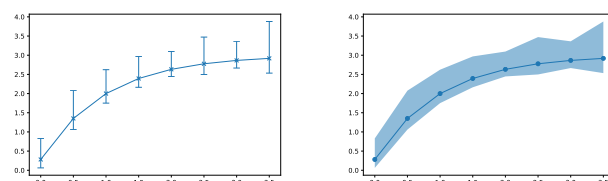
How can we appropriately communicate the degree of uncertainty to the reader in a graphical way, like in diagrams, graphs, or charts?

3.1.3 Forces:

- (F1) Showing uncertainty information in a graph would provide more information, but it makes the graph more difficult to understand. On the other side, neglecting uncertainty gives a completely wrong impression of certainty and confidence.
- (F2) Every data point could have a different amount of uncertainty. This respective error of each data point should be visible in the graph, but without confusing the reader. The limits could even be asymmetric around the mean and could also be different for every dimension of the graph (e.g., not only on the x-axis or y-axis but maybe both).

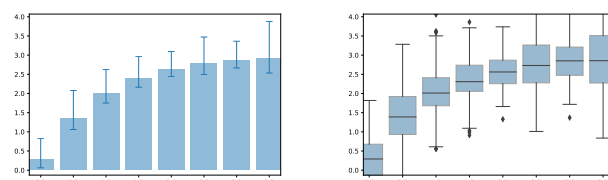
3.1.4 Solution. Draw an overlay on the visualization which indicates the size of the uncertainty or error. This could be done via lines (see Fig. 8(a) and Fig. 8(c)), or areas around the data points which are sized in correlation to the underlying error (see Fig. 8(b) and Fig. 8(d)). These error indicators should be drawn on the same position as the data point, but thinner, more transparent or a little brighter/dimmer in order to be distinguishable.

Regarding the kind of error, there are multiple possible options to choose from (see 3.1.8 for a list of different commonly used error types). While the most common is the standard-deviation from the mean, they all have their respective use-case, and therefore it is crucial always to describe which error is shown in the graph so that the reader can interpret it correctly.



(a) A linechart with error bars, often used for showing standard deviations or standard errors.

(b) A linechart with error area, often used for showing the prediction or confidence interval.



(c) A barchart with error bars, often used for showing standard deviations or standard errors.

(d) A boxplot showing the error indicators, but also giving a rough image of the distribution.

Figure 8: Examples of plots with ERROR INDICATOR

3.1.5 Problem Resolved. In the next figure, a graph is shown which looks quite similar to the one in the problem section, but this time an error indicator in the form of a bright area is drawn around the prediction lines representing the uncertainty. With this additional information, the reader can directly see how much confidence one has in the predictions and make informed decisions.

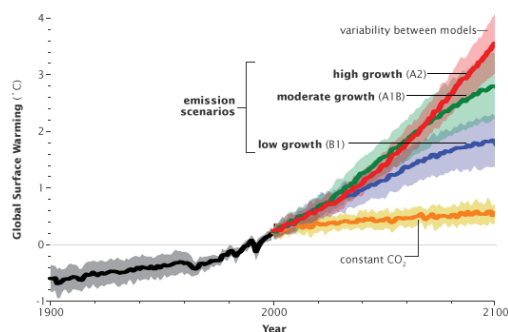


Figure 9: Another visualization of climate change. In this case the global warming is predicted and *ERROR INDICATOR* is used to depict the uncertainty of the predictions. Figure taken from the 2007 IPCC report about climate change [40].

3.1.6 Consequences. After applying the pattern, we see indications of errors in our visual representations. While this solves our primary

problem, it may come at a cost which is discussed here in the consequences.

+ *Benefits*

- + The error is shown subtly so that the reader is not disturbed by it.
- + The error shown in the graph corresponds to the error in the underlying data.
- + The reader is informed about the error and can reason and make better-informed conclusions based on that.

- *Drawbacks*

- Showing the errors still adds more elements to a visualization and may overload it.
- Readers may not understand the meaning of the shown error (e.g. standard deviation vs. standard error, confidence vs. prediction), and could misinterpret them as actual minimum and maximum. Clarify this in the legend or graph description.

⚠ *Liabilities*

- ⚠ Make sure the error indicators do not clutter the whole visualization.
- ⚠ Make sure to explicitly state the kind of error you are showing in the graph.
- ⚠ Combine the graphical representation with other forms of communication, like textual or numeric. This helps to avoid misinterpretations.

3.1.7 *Known-Uses.* ERROR INDICATORS are widespread in graphical representations for all kinds of media like newspapers, TV, books, posters, papers in public, in industry, and of course, also in academia. To name just some examples: Climate and Weather Reports [4, 20], Academic Research Papers [41].

3.1.8 *Types of Error-Intervals.* There are several different types of error intervals showing different kinds of uncertainties. While the standard deviation and the standard error are the most common, others like prediction or confidence intervals are also used, especially in prediction models. Quantiles and percentiles are used mostly for boxplots.

- **Standard-Deviation:** Represent the average deviation from the mean and is commonly called σ .
- **Variance.** The variance is the squared standard-deviation, most commonly written as σ^2 .
- **Standard-Error:** Often also called the standard error of the mean. It signifies how well the mean represents the mean of the whole population. The more samples you have, the smaller the standard-error gets.
- **Confidence-Interval:** The interval bounds where the real value of some estimation may lie with a given probability e.g., the real mean of a data set lies is between 4.5 and 4.8 with 90% probability (and 10% probability, that it may be outside of this range).
- **Prediction-Interval:** A prediction where future values are likely to fall into, most of the time. Depends on regression models, e.g., our model can predict that in half the trials, more than 95% of the values will be within 4.2 and 5.2 (in the other half, less than 95% will be within this range). Since

the prediction interval also accounts for the variance of the data, it is wider than the confidence interval.

- **Tolerance-Interval:** A range of predicted values, like the prediction-interval but with higher probability e.g. our model can predict that in 80% of the trials more than 95% of the values will be within 4.1 and 5.4 (and in the other 20% of the trials, less than 95% will be within this range). Since the tolerance interval demands a higher probability of being correct, it has a broader range than the prediction-interval.
- **Min-Max Interval:** The simple minimum and maximum of all measured values. This could be a huge range.
- **Quantiles / Percentiles:** Quantiles (or percentiles) represent a given portion of the population e.g., the 50% percentile (also called median) is the value that splits the population into two halves. When you divide the population into four parts, you get quartiles: The 1st quartile represents the value at the 25% percentile, the 3rd quartile represents the 75% percentile.

3.2 Variant: BOXPLOT

A **BOXPLOT** indicates the errors as well as the distribution of data by showing the quartiles, the median, the interquartile range, and potential outliers.



A boxplot is an advanced variant of an error indicator because it also tells something about the distribution of the data. Therefore it could be seen as the combination of an ERROR INDICATOR and a DISTRIBUTION PLOT. The boxplot was first described by Tukey [44].

A boxplot consists of several graphical components: the median line, the quartile box, the whiskers, and outliers. The median line signifies the position of the median in the data (the 50% percentile). The quartile box is a box drawn around the lower quartile (25% percentile) and the upper quartile (75% percentile). The whiskers signify the range of values and span 1.5 times the interquartile range (IQR = 75% percentile - 25% percentile) from the median (but not longer than the absolute maximum or minimum of the data). All values which are outside of this range are called outliers and visualized with special markers e.g., crosses or dots. Some variants also draw the mean value in the form of a small point or star, to give even more information about the distribution.

The main benefit of the boxplot is its simplicity: By illustrating a five-number summary, it shows the distribution of the data simply and recognizably. A boxplot is useful for values that have only one mode (one main center of density). If a dataset is multimodal, the boxplot is not appropriate.

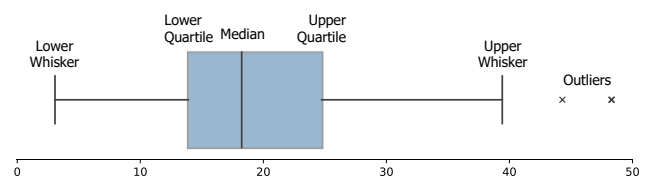


Figure 10: Example of a boxplot, showing median (50% percentile), interquartile range (IQR: 25% to 75% percentile), whiskers (Quantiles $\pm 1.5 \times$ IQR), and outliers (outside IQR).

3.3 Pattern: DISTRIBUTION PLOT



A *DISTRIBUTION PLOT* shows the distribution of numerical data by drawing the counts or frequencies of values over the whole range. They visualize density, alignment, and concentration of data. The more frequent a value is, the more visual emphasis it gets.

3.3.1 Context. We have data that should be graphically communicated to a human in order to convey some message. The exact shape and kind of data vary, and the data may contain varying degrees of uncertainty. Typically this is done with graphs and diagrams, visualizing the underlying data points. The presentation media is mostly restricted to 2d-surfaces (e.g., computer screens, smartphones, images, or printed paper). Displaying only the mean and its uncertainty can be misleading when data is skewed, asymmetric, and multi-modal (has multiple regions which are highly likely).

3.3.2 Problem. When showing a data set graphically, it can be quite difficult to decide how much detail is necessary in order to convey as much information as possible, without overwhelming the reader with too much data. The underlying distribution of data values may often be skewed, unevenly partitioned, and may have multiple centers of mass. For arbitrary data sets, this cannot be known beforehand, and therefore such information should be shown to the user. How can this be communicated clearly?

3.3.3 Forces:

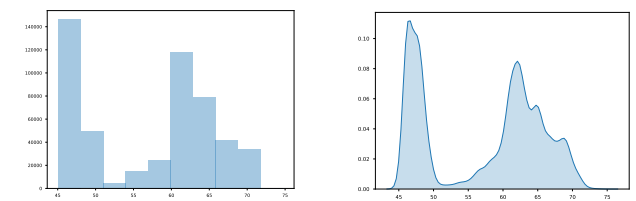
- (F1) All important properties of the distribution should be made visible (e.g., skewness, kurtosis, centers of distributions), but still, the visualization should not be overloaded with too much clutter and symbols.
- (F2) Sometimes, data series consist of really many data points (thousands, millions, or even more) that cannot be displayed individually. Still, we want to visualize the overall distribution of the data in order to make decisions based on that.
- (F3) In some cases, there may also be many data series to visualize (not only many data points). Showing the individual distributions in such cases would completely overload the visualization.

3.3.4 Solution. Show the data in the form of discrete or continuous probability distributions. Discrete types (Histograms, see Fig. 11(a)) divide the whole value range in a fixed number of bins and represent the count of values that fall into the respective bin. Sometimes these absolute counts are normalized not to represent counts anymore, but probabilities. A continuous type could either be parametric or non-parametric. Parametric continuous models are created by adjusting parameters of a specific known function to find the best fit (e.g., the normal distribution, or the beta distribution). Non-parametric distributions are constructed by applying a generic smaller distribution (called kernel) to the bins of a histogram and then summing them all up and normalizing them. This is called kernel density estimation and creates a smooth and continuous histogram over the whole data range. In either case the result is a distribution function which represents the underlying data, and can

be visualized in a graph like shown in Figures 12(a), 12(b), 12(c), and 12(d). Such distribution functions have interesting properties: They show the frequencies or likelihoods of the underlying values over the whole value range. Therefore, if the data is unevenly distributed, or asymmetric, or skewed in another way, this can be immediately seen. Furthermore, it is possible to have multiple dense and sparse areas.

3.3.5 Variant: HISTOGRAM. A histogram (see Fig. 11(a)) aggregates the data points into bins or buckets and visualizes them as bar charts. One axis shows the range of values, and the other its counts or likelihoods (if it is normalized). Showing the real counts of data points falling into a bin gives the reader a sense of the absolute numbers. If the area is normalized to 1, the bins represent the likelihoods for the respective bins. The amount of bucket can change the gestalt of the whole visualization, depending on how many data points fall in which bucket. This can change the whole meaning of a distribution and is, therefore, a critical and sensitive parameter.

3.3.6 Variant: PROBABILITY DENSITY PLOT. A probability density plot (see Fig. 11(b)) shows a continuous model of the distribution for the underlying data, which looks smooth, but the smoothing could introduce probabilities for data, which are not supported in the underlying source. However, it allows a sophisticated analysis of the probabilities because of the used smooth probability density function. It can be created using kernel density estimation or fitting the parameters of a known function to the data. It is visualized using a line graph, or an area chart, with one axis showing the range of values, and the other the respective likelihoods.



(a) A histogram, showing discrete bars representing the accumulated slices of the frequency or probability for occurrence of a value over a whole range.

(b) A probability density plot showing a continuous representation for the probability of occurrence of a value over a whole range.

Figure 11: Examples of two basic variants of a DISTRIBUTION PLOTS: the histogram and the probability density plot.

3.3.7 Variant: VIOLINPLOT. A violin plot is created the same way a probability density plot, but mirrored along the axis (so it looks like a violin) and displayed side by side with other series of data (see Fig. 12(a)). This shows the distributions of multiple series in one graph, which is very informative and precise but could overwhelm the reader easily.

3.3.8 Variant: SCENARIO PLOT. A scenario plot (see Fig. 12(b)) shows randomly chosen samples from the underlying data. By visualizing many possible examples in the graph, the underlying distribution gets visible via the density of the lines or points. Very often, such graphs use transparency to even better distinguish between the

widespread scenarios and rare scenarios. This kind of visualization gives a way of natural interpretation of the underlying data. In such a way, the reader can intuitively see what the most likely outcomes may be, and where the dense areas of the data are. The biggest drawback is that, unless many random samples are drawn, the shown scenarios only give a very rough image of the underlying data.

3.3.9 Variant: STRIPLOT and SWARMPLOT. Strip plots and Swarmplots show the distribution by drawing discrete points representing fractions of the underlying data (see Fig. 12(c) and 12(d)). Areas with more points represent higher density, while areas with fewer points represent sparse data. This allows intuitive, natural interpretation of the data because the reader can immediately see where most of the data points are, and knows that here also the most of the underlying data is concentrated. While these visualizations are very intuitive, they are the least precise due to the quantization of the bins and the points.

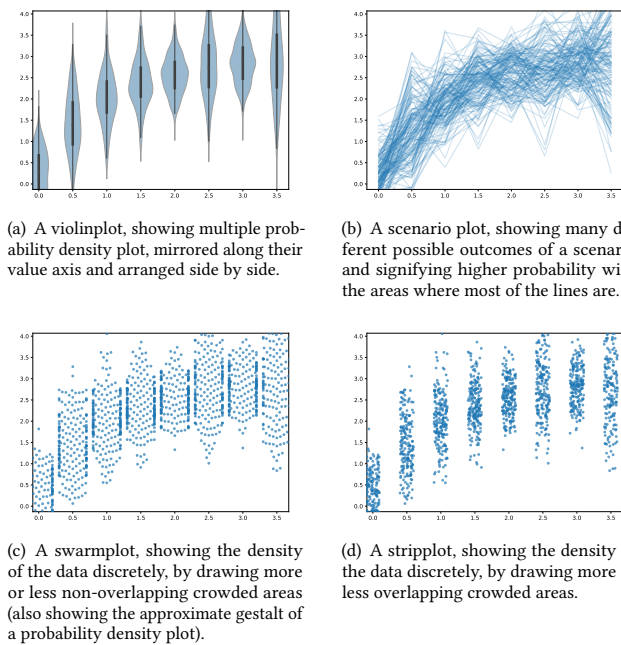


Figure 12: Examples of advanced variants of DISTRIBUTION PLOTS: violin plot, scenario plot, swarm plot, and strip plot. All depicted visualizations differently represent the same underlying dataset.

3.3.10 Consequences. Showing the distributions of data has some consequences which are discussed here:

+ Benefits

- + The reader can see the actual distribution of the data. Not just the aggregated moments.
- + The reader can clearly distinguish between high-density areas and low-density areas.

- + The reader is not overwhelmed by too much data (in comparison to showing all data points individually)
- + Uneven, asymmetric as well as multi-modal distributions can be spotted with ease.

- Drawbacks

- It is difficult for humans to understand the actual probabilities in probability density plots. Histograms with fewer bins are more suitable for such situations because they at least show actual aggregated values that are more easily recognizable.
- Sometimes, people are overwhelmed with seeing the whole distribution of values. In such cases, using a boxplot or just expressing the uncertainty via an error indicator would be more helpful.

⚠ Liabilities

- ⚠ For plots that use bins and quantized visualization like Histogram, Stripplot, and Swarmplot, make sure that you use an appropriate amount of bins and point sizes in order to convey a truthful message to the reader.
- ⚠ Avoid drawing too many data series into one distribution plot because this can get messy and convoluted very fast.

Known-Uses. Distribution plot are very common and used in many publication, for example histograms are used in population pyramids to show the distribution of people in respect to their age [35, 45], or in photography to visualize the exposure or color values. Figure 13 shows a histogram from the area of bluetooth communications. Examples for violinplots can be found in [32] and for probability density plots in [2], and scenario plots in [7]. Figure 14 shows a real-life scenario plot for climate change models.

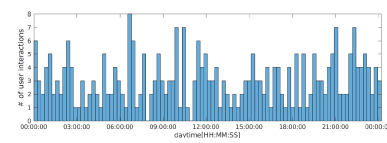


Figure 13: A histogram, showing the number of interactions over time for bluetooth communication. Graph taken from [37].

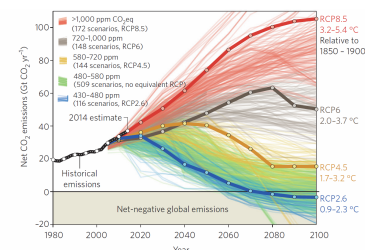


Figure 14: A scenario plot, showing the outcomes of different scenarios for CO₂ emissions over the next 100 years. Figure taken from [11].

4 NUMERICAL REPRESENTATION OF UNCERTAINTY

$$G = (6.67408 \pm 0.00031) \cdot 10^{-11} \frac{m^3}{kg s^2}$$

Best known approximation for the Newtonian constant of gravitation (as of 2016) [29].

In this section, we show a pattern that resolves the problem of stating the uncertainty in data: NUMBERS WITH UNCERTAINTY. While in the other sections, the numbers were translated into textual or visual form, here, the numbers and its respective uncertainty are stated explicitly in numeric form.

4.1 Pattern: NUMBERS WITH UNCERTAINTY

10±1

State the range of uncertainty when communicating a number. Use special syntax to describe the range and also make the type of uncertainty explicit.

4.1.1 Context. You want to explicitly communicate numbers which have inherent uncertainty.

4.1.2 Problem. How can numeric values and their according uncertainty be stated reasonably? Numbers are often stated as ambiguous words of frequency, estimative words of probability, or simplified comparisons. Their actual meaning is strongly dependent on context and is highly subjective. Also, the confidence in the data is often only given vaguely with numeric hedge words like e.g., *absolutely sure*, *around*, *above*, or *approximately*. Visualizations can express many numbers and facts in one picture, but they are imprecise. Only actual numbers can precisely express facts and their corresponding uncertainties.

4.1.3 Forces:

- (F1) Convey more information about the uncertainty, but do not overwhelm the reader with long textual descriptions. The notation should be concise and intuitive.
- (F2) The syntax should not be too complicated. It should connect to a common understanding of mathematics and notations, like adding or subtracting.
- (F3) If the number has a physical unit attached to it, this physical unit should also be applicable to the uncertainty.

4.1.4 Solution. Communicate the data explicitly in a numeric form and also state the respective uncertainty. First, write the actual number followed by the uncertainty and then optionally followed by a physical unit. Separate the number and its uncertainty with “±”, or enclose the uncertainty in parenthesis (which is preferred by the standard). Chose an appropriate type of uncertainty (e.g., the standard deviation or the standard error). Round the numbers (and its uncertainties) to a reasonable number of significant digits. It does not make sense to state a number with ten decimal digits when the uncertainty is ±0.1. Even if the uncertainty is not stated explicitly, the numbers should be rounded to a reasonable number of significant digits, not to give a wrong impression of confidence.

Examples for a weight measurement [19]:

- 1.2147kg with combined uncertainty of 0.0035kg
- 1.2147(35)kg
- 1.2147(0.0035)kg

- 1.2147kg ± 0.0035kg
- (1.2147 ± 0.0035)kg

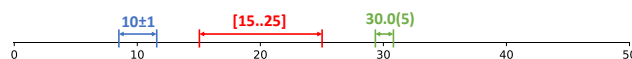


Figure 15: Examples for numbers with uncertainty drawn along a number axis (just for visualization; the graph is not part of the pattern).

Significant Digits and Rounding. The guide to the expression of uncertainty in measurements [21] defines how many significant digits are important for a value and, therefore, also should be stated. The number depends upon the uncertainty in the data. The rules are as follows:

(explained with the example: $987.654321 \pm 1.23456789$)

- (1) Round the uncertainty to 2 significant digits:
 $1.23456789 \rightarrow 1.2$
- (2) Round the number to the same precision as the least significant digit of the rounded uncertainty:
 $987.654321 \rightarrow 987.6$
- (3) Finally combine the number and the uncertainty:
 $987.6(1.2)$ or
 987.6 ± 1.2

uncertainty. There are different variants of what type of error can be used. Therefore, if the type is not clear from the context of the number, it should be stated explicitly which one is used. A good default is the combined standard deviation for the 1σ distance (65% confidence), but very often also the standard deviation for 90% or 95% confidence is used. If nothing is given, uncertainty values should be taken very carefully because these ranges could be very different depending on their type and confidence. Section 3.1.8 shows different variants of errors like standard deviation, standard error, prediction interval, or tolerance interval.

Number Format. There are different formats to represent numbers. The most common is normal decimal notation like 1.23, 5, 0.003, -5.7 . Another possible notation is scientific notation which looks like the following: 1.8×10^3 , 4.55×10^{-5} , $1.23 \cdot 10^{22}$. Here the number is normalized to one digit before the comma and multiplied by a decimal factor.

Very often, numbers are accompanied by some physical units, and for such, the SI system should be used [1]. After the number, the unit abbreviation should be stated (e.g., kg, m, s for Kilogram, Meter, or Second). To avoid writing many zeroes, the SI-prefixes like mega, giga, kilo, dezi, milli should be used e.g., 1MW (one megawatt), 23.5mm (millimeter).

Variant: Relative Uncertainty.

Relative Uncertainties represent the uncertainty as a percentage of the value. Use this if you want to show the magnitude of error compared to the base value. Avoid usage when the value is around zero, or very small compared to the uncertainty.

Relative uncertainties represent the uncertainty value as a percentage of a given base number. This allows for fast recognition of the magnitude of uncertainty in comparison to its number. The

default way is to divide the absolute uncertainty by the measured value, but sometimes also another defined value is used as 100% and taken as the base because this avoids the problem of relative uncertainties getting huge in the vicinity of zeroes. For multiplication and division, the error propagation is easier calculated with relative uncertainties.

- $257.3 \pm 2.6\%$
- $100.02147g \pm 7\%$

Variant: Interval Arithmetic.

Interval Arithmetic represents the range of values without stating the most expected number. Use this if the values inside have uniformly distributed probability, or if you want to state just the limits of some value.

Another way of depicting uncertainty is by stating an interval or even a probability distribution for the data. For example, by stating the min and max level, it can be safely assumed that the true value is somewhere in between. However, just giving the range does not state anything about the distribution of values in between, which could be e.g., uniform, normal, or even be a completely arbitrary distribution. This is why the semantics of the range, as well as its distribution, should be defined somewhere that the reader can interpret it correctly. It should also be stated if the range is inclusive or exclusive (mathematically, one could use brackets “[”]” for inclusive ranges and parentheses “()” or exclusive ranges [18]). Examples:

- The values are ranging from 10.2 to 10.7 (95% confidence).
- Values are between 10.2 and 10.7 with 95% confidence.
- [10.2, 10.7] (closed interval, includes limits)
- (10.2, 10.7) (open interval, excludes limits)
- [45, ..., 55] (also very common, but ambiguous: range could be interpreted as integer values only)
- [45 – 55] (ambiguous: could be interpreted as subtraction)

4.1.5 Consequences.

+ Benefits

- + Numbers with Uncertainty are not ambiguous and cannot be misjudged like hedge words.
- + Numbers with Uncertainties can be used for further calculations.
- + Stating the value for the uncertainty and the type of uncertainty helps to interpret the magnitude of the error unambiguously.

- Drawbacks

- To be able to write the numbers with uncertainties, the writer, of course, needs to measure, calculate, or acquire the uncertainties before. This is an unavoidable overhead.
- Stating the explicit numbers with uncertainties may give away more information than is wanted by the writer. Sometimes you want to stay ambiguous (although the authors do not recommend this).
- Large amounts of numbers scare most readers. Shift data tables to the appendix, share a link to the source materials or visualizing the data in a diagram.

- Relative errors can be misunderstood and misleading. What is 100%? The value itself, or some calibration value from the sensor. State this somewhere in an accompanying text.
- Relative errors get huge in the vicinity of zero and are infinite at precisely zero. On the other hand, they diminish for huge numbers if the error is small.

⚠ Liabilities

- ⚠ Stick to the same number format (be consistent).
- ⚠ Round the numbers to the appropriate significant digits.
- ⚠ State the kind of uncertainty you are using for the numbers. Is it the standard deviation or the standard error? Which confidence value are you using?
- ⚠ Using normal decimal notation is sometimes difficult to read. Especially huge or small numbers may be difficult to understand and often cannot be imagined by the average reader. State huge or tiny numbers using numeric words for the decimal exponent e.g., million, billion, thousandth, millionth, or the respective SI prefixes in the physical units like mega, giga, micro, nano.
- ⚠ Using scientific notation is not very common among the general public. An ordinary reader may not understand this. Therefore, use decimal notation whenever possible.

5 TEXTUAL REPRESENTATIONS OF UNCERTAINTY

I don't know half of you half as well as I should like; and I like less than half of you half as well as you deserve.

Bilbo Baggins in *Lord of the Rings*; J. R. R. Tolkien, 1954

In spoken and written language, we use textual representations of uncertainty. With that, we mean words like often, sometimes, high confidence, very good, and so on. These so-called WORDS OF ESTIMATIVE PROBABILITY and NUMERIC HEDGE WORDS are used very commonly in textual media like newspapers, books, and articles, but of course also in spoken language. The usage of such words could completely change the meaning of a statement by either exaggerating or understating the meaning.

The following patterns will be discussed in this section:

- **Anti-Pattern: WORDS OF ESTIMATIVE PROBABILITY.** Words for probabilities or frequencies which are ambiguous and highly subjective, e.g., often, likely, doubtful, probably.
- **Anti-Pattern: HEDGE WORDS.** Words which emphasize, understate, or change the importance of some statement e.g., highly, very, significantly, hardly.
- **QUANTITATIVE COMPARISONS.** Paraphrased substitutes for actual probabilities and frequencies, which are not ambiguous, but still intuitively understandable, like e.g., 9 out of 10 people, on every 3rd usage, once a week.

WORDS OF ESTIMATIVE PROBABILITY is an anti-pattern that is solved by QUANTITATIVE COMPARISONS described in this section, but also by the pattern EXPLICIT NUMBERS and EXPRESSION OF UNCERTAINTY, described in the next section.

Table 1: The Odds Table by Sherman Kent, created 1964, but released to the public in 2012 [24].

Term	Odds
Certain	100%
Almost certain	93% ± about 6%
Probable	75% ± about 12%
Chances about even	50% ± about 10%
Probably not	30% ± about 10%
Almost certainly not	7% ± about 5%
Impossible	0%

Table 2: Likelihood Scale from Guidance Notes for Lead Authors of the IPCC Fifth Assessment Report on Consistent Treatment of Uncertainties [17].

Term	Likelihood of the outcome
Virtually certain	99-100% probability
Very likely	90-100% probability
Likely	66-100% probability
About as likely as not	33-66% probability
Unlikely	0-33% probability
Very unlikely	0-10% probability
Exceptionally unlikely	0-1% probability

5.1 Anti-Pattern: WORDS OF ESTIMATIVE PROBABILITY

*“Probably”
“Certainly”
“Likely”
%?
WORDS OF ESTIMATIVE PROBABILITY are commonly used words to signify a vague notion of likelihood or frequency of an event.*

5.1.1 Problem. In common language we are accustomed to use and interpret words for frequencies like *very often*, *seldom* or *rarely*. In spoken language, we can conclude the actual frequency from the context and worldview of the person we are talking to, but still, it often leads to misunderstanding and conflict. For written text, it is even more difficult because you often do not know the context and worldview of the writer. It was shown that even trained officers still interpret them in a wide range of possibilities (depicted Figure 16) [14]. Think of the term *likely* - what percentage of likelihood would that represent, 70%, 80%, or 50%? Is an event *likely* to occur, if it occurs once a week, or once every hour? Or what do you think about, if an event is *extremely rare*? Would that be once a year, or once in a lifetime? Recently scientists from Italy have observed a real extremely rare event, in fact, one of the rarest events in the universe, on their search for dark matter: They detected the decay of a xenon-124 atom, which has a half-life of $(1.8 \pm 0.6) \times 10^{22}$ years [50]. That is what they see as *extremely rare* in the context of their research. Depending on the context, the subjective biases, and cultural background, words of estimative probability could have a completely different meaning.

There have been approaches to standardize such words to bring order to the chaos of the ambiguity of natural language. In his article from 1964, Sherman Kent was among the first to describe

the words of estimative probability, and he assigned ranges of probabilities to them (see Table 1) [24]. Unfortunately, this article was under disclosure until 2012. In the meanwhile, also others tried to define scales for estimative words, and we ended up in a jungle of different scales again [17, 25, 27, 30, 31, 48]. However, one of the most commonly known and used is the scale by the IPCC (intergovernmental panel on climate change). Table 2 shows the likelihood taken from the *Guidance Notes for Lead Authors of the IPCC Fifth Assessment Report on Consistent Treatment of Uncertainties* [17]. Most approaches define a range of numerical probabilities to the terms in natural language. Sometimes these ranges overlap, sometimes these have some gaps in between. In the end, still, a huge range of interpretation is possible, even if we accept and use such definitions.

Still, such approaches are destined to fail, because humans are unconsciously influenced by their biases and interpret words of probability differently, even if they know the exact scale beforehand [14]. This was proven by Heuer et al. by asking trained NATO officers for their interpretation of probability phrases. Though they were accustomed to standard scales, they still interpreted the phrases differently (and inconsistently) when used in real questions [14, 16]. Figure 16 shows the results and wide ranges of probabilities). The consequence is, that if even trained officers are so inconsistent in their ratings, how could one assume that in different situations, it would work better?

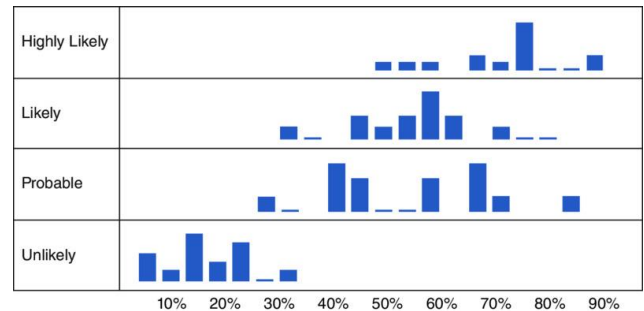


Figure 16: NATO Officers' Interpretations of Probability Phrases [14] (taken from [16]).

It is not our intention to redefine or combine the existing scales; we just repeated them as illustrative examples to show the ambiguity. Instead, we propose to avoid such words altogether, and use EXPLICIT NUMBERS and QUANTITATIVE COMPARISONS instead.

5.1.2 Consequences. The usage of WORDS OF ESTIMATIVE PROBABILITIES has several consequences, and most of them are not desirable. Nevertheless, they are so common in natural language that people do not see the inherent drawbacks or never dared to question and think about them.

- + Very common and simple: They are natural to use and fast to write. They are well known and used in natural language.
- Wrong impression of common understanding: The reader seems to understand the meaning of the words, but in reality, the interpretation may be completely different from what the writer wanted to convey.

- Human bias: All humans have their background and framing, and therefore also interpret words of estimative probabilities different. Even when the words are exactly defined, we tend to interpret them differently (unconsciously). People estimate extreme events with subjectively higher or lower probability than it is [16, 23].
- ⚠️ Ambiguity: They allow for a wide range of interpretations, which could lead to severe problems in understanding. Nevertheless, sometimes such ambiguity can be accepted or is even intended.

5.2 Anti-Pattern: NUMERIC HEDGE WORDS

NUMERIC HEDGE WORDS are used to describe approximate ranges or vague proximities around numbers.

5.2.1 *Problem.* Words like e.g. *almost, exactly, about, at least*, have ambiguous meaning [9]. Similar to the word of estimative probability, these words have a different meaning in different contexts and are prone to human bias [13, 22]. The problem here is that people have many different definitions of precision and scaling in their thinking. “Below 9” could mean 8 or 7, but it also could mean 8.9 or 8.5, depending on the subjective measure of precision. Ferson et al. [9] tried to define specific semantics for some of the most common hedge words and succeeded, but their definitions are not very widely known. See Figure 17 for an example of their results. They distinguished between symmetric hedge words (e.g., about, around, approximately, roughly), asymmetric hedge words (e.g., below, over, at least, almost) and ranges (e.g., between, within) and tried to find exact definitions for them by taking thousands of samples in anonymous user surveys. Afterward, they defined the boundaries based on the survey results. Table 3 shows their results. The only problem lies still in the exact definition of precision and the impreciseness of natural language. Sometimes such hedge words are used inconsistently, and the exact definition is overruled (people sometimes exaggerate or understate).

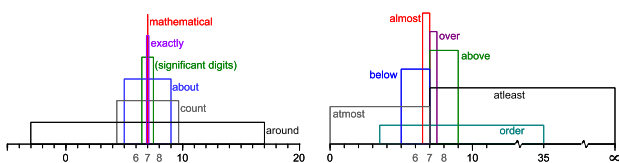


Figure 17: Example ranges for symmetric and asymmetric numeric hedge words around the number 7 (taken from [9]).

5.2.2 *Consequences.* NUMERIC HEDGE WORDS are used often in natural language, although they imply huge ambiguities as a consequence. Therefore the consequences are quite similar to the WORDS OF ESTIMATIVE PROBABILITY.

- + Very common and simple: They are fast to write, seem easy to understand, and are used in natural language.
- + To give precise single point estimates could be very difficult. Hedge words can at least signify that there is uncertainty in the numbers.

- Wrong impression of common understanding: The reader seems to understand the meaning of the words, but in reality, the interpretation may be completely different from what the writer wants to convey. All humans have their background and framing, and therefore also interpret hedge words differently [9].
- Standardization still uncommon: There is no exact definition of the words. Attempts like [9] are still uncommon and not widely known. Still, there is the question if people would stick to it, even if some standard body defined it (as can be seen in WORDS OF ESTIMATIVE PROBABILITIES, Figure 16).
- ⚠️ Ambiguity: They allow for a wide range of interpretations, which could lead to severe problems in understanding. Be aware of the ambiguity! Nevertheless, sometimes, this can be accepted or is even intended.

Table 3: Hedge words and their mathematical interpretation by Ferson et al. 2015 [9]. x is the hedged number, d signifies the precision e.g. the decimal place of the last significant digit of x .

Hedge word	Interpretation
mathematical x	x
exactly x	$x \pm 10^{-d-1}$
x	$x \pm 0.5 \times 10^{-d}$
about x	$x \pm 2 \times 10^{-d}$
around x	$x \pm 10^{-d+1}$
count x	$x \pm \sqrt{x}$
almost x	$[x - 0.5 \times 10^{-d}, x]$
over x	$[x, x + 0.5 \times 10^{-d}]$
below x	$[x - 2 \times 10^{-d}, x]$
above x	$[x, x + 2 \times 10^{-d}]$
at most x	$[(-\infty 0), x]$
at least x	$[x, \infty]$
in the order of x	$[0.5x, 5x]$
between x and y	$[x, y]$

5.3 Pattern: QUANTITATIVE COMPARISONS



Describe odds and probabilities in relative comparison to real-life cases, so that humans can relate to them and get an intuitive understanding of the underlying likelihoods.

5.3.1 *Context.* You want to communicate some data of percentages or frequencies in textual form.

5.3.2 *Problem.* How can percentages or frequencies be stated, while still maintaining an easy to understand level of comprehension? In textual descriptions of data (especially of statistical data), very often probabilities and likelihoods are used. While just stating the numbers is precise, but could still lead to misunderstandings because humans are terrible at understanding odds and probabilities because they are very biased when it comes to likelihoods of events (e.g., confirmation bias, availability bias, outcome bias). Also, stating a number implies precision until the least significant digit. Stating a fraction or real-life comparison implies higher uncertainty

because we are used to the fact that in real-life events are not that certain.

5.3.3 Forces:

- (F1) The reader should get a rough idea of the odds and frequencies for some event. Exact numbers are too detailed, and words of estimative probability would be too ambiguous.
- (F2) Humans are bad at interpreting numbers for probabilities. We want to state comparisons that are easy to understand but still exact enough and unambiguous so that the message is conveyed in the right way.
- (F3) Stating exact numbers with uncertainties would hinder the flow of reading.
- (F4) We want to have a phrase for describing different scales:
- Probabilities, Likelihood, Odds
 - Frequencies, Occurrences, Amounts
 - Exposures, Durations

5.3.4 Solution. The solution consists of three steps: First, try to express your percentage as a fraction. Fractions are more easily recognized than percentage numbers or ratios e.g., 16.7% gets 1/6, 50% gets 1/2. The next step is to use descriptive language like: in 1 out of 6 cases, or every second time. The third and last step is to find a real-life scenario where this fraction applies to e.g., getting a 6 in a dice throw, or succeeding on every second try. The last part is the most difficult because it depends on subjective frames and scales. If your time scale is days, then once in a week refers to the fraction 1/7, but when thinking in hours of a working week, it would mean 1/40 (which could also differ from company to company).

Solution-Steps:

- (1) Convert the ratio number into a simple fraction (can be approximate, rounding is allowed): x/y
- (2) Express the fraction in terms like x out of y , x per y
- (3) Try to find a real-life comparison and time frame: winning the lottery; once a week; every time you blink; every hour.

Use specific phrases to establish a quantitative comparison to the actual numbers for probabilities, frequencies, or exposures that the reader can relate to from his or her real-life experience. In the following sections, examples for such words are given, but it must be clear that these words should also be chosen with care for the cultural context of the reader. More about that will be discussed in the consequences.

Probabilities, Likelihoods, Odds. Probabilities and likelihoods can be rewritten as odds, which already makes them easier to understand. They also can be compared with real-life objects which represent random events like dice, lottery, or casino games. Another possibility is to convert the probability into a frequency by adapting it to a time frame of everyday life situations: e.g., once in a lifetime, every time you fly with a plane, once every second.

Examples:

Frequencies, Occurrences, Amounts. Frequencies or occurrences state how often something occurs. They can be compared with a fixed time frame like e.g., days in a year. Examples (with days as the time scale):

- 1/7: once a week.
- 1/365: once a year.

Table 4: Percentages and their relative comparison.

Percentage	Fraction	Comparison
0.1%	1/1000	1‰; One in a thousand.
1%	1/100	One in a hundred.
3%	1/36	Getting 6 twice from 2 dice throws. Betting on a number in roulette.
5%	1/20	
10%	1/10	10 out of 100.
15%	1/7	
16.7%	1/6	Getting a 6 in a dice throw.
20%	1/5	Every fifth time.
25%	1/4	Every fourth time; Quarter.
33%	1/3	Every third time.
50%	1/2	Every second time; Half of times; Coin flip.
66%	2/3	In two thirds of cases. 1 σ confidence interval
75%	3/4	In 3 out of 4 cases.
90%	9/10	In 9 out of 10 cases.
95%	19/20	Only 1 failure in 20 cases 2 σ confidence interval
99%	99/100	Only 1 failure in 100 cases
99.9%	999/1000	Only 1 failure in 1000 cases

- 1/25000: once in a lifetime (70 years * 365 days)
- 1/2: every second day.

Exposures, Duration. Exposures depict a fraction of how long some event takes place. It can be described using fixed time frames or by relating to real-life examples.

Examples:

- Exercise 30 minutes per day.
- Practice for 1 hour, twice a week.
- Radioactive exposure during an average airplane flight.
- UV-light exposure during an afternoon in the swimming pool.

5.3.5 Consequences. Here the consequences are described.

+ **Benefits**

- + Odds that are very common are likely to be easily understood by the reader.
- + Stating comparison instead of just pure numbers make the text more lively and easier to read.
- + People who have problems with percentages may at least understand a real-life comparison.

- **Drawbacks**

- The cultural context and human bias come into play. Humans tend to under- or overestimate their luck in casino games, dice rolls. Also, comparison to car drives, airplane flights, bus rides, or going to the cinema may be off, since not everyone is doing that with the same frequency and could, therefore, misinterpret the numbers.

- Making comparison prolongs the text more and could, therefore, be more difficult to understand. Also, unrelated comparisons increase the semantic energy needed to understand a sentence, which also makes it more difficult to read. Sometimes just stating the numbers would be easier.

⚠ Liabilities

- ⚠ Choose a comparison only if stating the number directly, would hinder the readability of a sentence.
- ⚠ Choose a comparison that is appropriate for the context and bias of the reader.
- ⚠ Often, the exact number has no simple comparison - here you have to pick between preciseness and legibility. State the exact number if there is not an appropriate comparison. Do not force yourself always to use quantitative comparison!

6 CONCLUSION

In this paper, several patterns of representing uncertainty were discussed for the areas of graphical, textual, and numeric applications. They can be used for the communication of uncertainties and errors to the reader. The most important thing here was that the way of representing the uncertainty should be appropriate to the receiver in order to convey the message in a meaningful and intended way. In order to avoid misinterpretations, it is reasonable to combine several methods and state the described kind of errors explicitly e.g., combine visual and numeric representations and state that the error is shown as standard deviation with 65% confidence (1 σ distance in a normal distribution).

Especially take care of textual representations because they give the highest room for interpretation. Visual representations can be interpreted and understood the fastest, but are also not very exact when it comes to the numbers. Numeric representations are the most exact but may be difficult to understand.

ACKNOWLEDGMENTS

I want to thank Lise Hrvatum for being my shepherd during the shepherding for the EuroPLoP 2019. Her nearly infinite patience and reasonable feedback during the shepherding helped me sorting out my thoughts to increase the understandability and readability of the paper. Additionally, I want to thank all the workshop members of the writers' workshop, which gave me invaluable feedback and many insights on different perspectives.

REFERENCES

- [1] BIPM. 2006. *The International System of Units (SI)* (8th ed.). 186 pages.
- [2] James Bornholt, Todd Mytkowicz, and Kathryn S. McKinley. 2014. Uncertain<T>: A First-order Type for Uncertain Data. In *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '14)*. ACM, 51–66. <https://doi.org/10.1145/2541940.2541958>
- [3] Alberto Cairo. 2016. *The truthful art: data, charts, and maps for communication*. New Riders. OCLC: ocn941982960.
- [4] Alberto Cairo. 2017. Uncertainty and graphicacy How should statisticians, journalists, and designers reveal uncertainty in graphics for public consumption? (2017), 12.
- [5] Damian Carrington. 2018. BBC admits 'we get climate change coverage wrong too often'. *The Guardian* (Sept. 2018). <https://www.theguardian.com/environment/2018/sep/07/bbc-we-get-climate-change-coverage-wrong-too-often>
- [6] Centre d'Estudis d'Opinió. 2019. Centre d'Estudis d'Opinió. <http://ceo.gencat.cat/ca/inici>
- [7] Christopher Pease. 2018. An Overview of Monte Carlo Methods – Towards Data Science. <https://towardsdatascience.com/an-overview-of-monte-carlo-methods-675384eb1694>
- [8] Harry Crane and Ryan Martin. 2017-04-04. Rethinking probabilistic prediction in the wake of the 2016 U.S. presidential election. (2017-04-04). arXiv:1704.01171 <http://arxiv.org/abs/1704.01171>
- [9] Scott Ferson, Jason O'Rawe, Andrei Antonenko, Jack Siegrist, James Mickley, Christian C. Luhmann, Kari Sentz, and Adam M. Finkel. 2015-02. Natural language of uncertainty: numeric hedge words. 57 (2015-02), 19–39. <https://doi.org/10.1016/j.ijar.2014.11.003>
- [10] Michael Friendly. 2008. The Golden Age of Statistical Graphics. *Statist. Sci.* 23, 4 (Nov. 2008), 502–535. <https://doi.org/10.1214/08-STS268>
- [11] Sabine Fuss, Josep G. Canadell, Glen P. Peters, Massimo Tavoni, Robbie M. Andrew, Philippe Ciais, Robert B. Jackson, Chris D. Jones, Florian Kraxner, Nebojsa Nakicenovic, Corinne Le Quéré, Michael R. Raupach, Ayyoob Sharif, Pete Smith, and Yoshiki Yamagata. 2014. Betting on negative emissions. *Nature Climate Change* 4 (Sept. 2014), 850. <https://doi.org/10.1038/nclimate2392>
- [12] Andrew Gelman and Julia Azari. 2017. 19 Things We Learned from the 2016 Election. *Statistics and Public Policy* 4, 1 (2017), 1–10. <https://doi.org/10.1080/2330443X.2017.1356775>
- [13] Thomas Gilovich, Dale Griffin, and Daniel Kahneman. 2002. *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge University Press, Cambridge, U.K.; New York.
- [14] Richards J. Heuer. 1999. *Psychology of intelligence analysis*. Center for the Study of Intelligence, Central Intelligence Agency, Washington, D.C.
- [15] Leo Hickman. 2018. Exclusive: BBC issues internal guidance on how to report climate change. <https://www.carbonbrief.org/exclusive-bbc-issues-internal-guidance-on-how-to-report-climate-change>
- [16] Douglas W. Hubbard and Richard Seiersen. 2016. *How to measure anything in cybersecurity risk*. Wiley, Hoboken. 00000.
- [17] IPCC. 2014. Guidance Notes for Lead Authors of the IPCC Fifth Assessment Report on Consistent Treatment of Uncertainties.
- [18] ISO. 2019. ISO 80000-2:2019. <http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/49/64973.html>
- [19] JCGM. 2008-09. JCGM 100:2008 Evaluation of measurement data Guide to the expression of uncertainty in measurement. <https://www.bipm.org/en/publications/guides/gum.html>
- [20] Jen Christiansen. 2017. *Visualizing Uncertain Weather - Scientific American Blog Network.pdf*. <https://blogs.scientificamerican.com/sa-visual/visualizing-uncertain-weather>
- [21] JCGM. 2009. JCGM 104:2009 Evaluation of Measurement Data - An introduction to the "Guide to the expression of uncertainty in measurement" and related documents. <https://www.bipm.org/en/publications/guides/gum.html>
- [22] Amos Tversky and Daniel Kahneman. 1971. BELIEF IN THE LAW OF SMALL NUMBERS. (1971), 6.
- [23] Daniel Kahneman and A Tversky. 1972. Subjective probability: A judgement of representativeness. *Cognitive psychology* 3, 3 (1972), 430–454. <http://datacolada.org/wp-content/uploads/2014/08/Kahneman-Tversky-1972.pdf>
- [24] Sherman Kent. 2012. Words of Estimative Probability. <https://www.cia.gov/library/center-for-the-study-of-intelligence/csi-publications/books-and-monographs/sherman-kent-and-the-board-of-national-estimates-collected-essays/6words.html>
- [25] Rachel F Kesselman. 2008. Verbal Probability Expressions in National Intelligence Estimates: A Comprehensive Analysis of Trends from the Fifties through Post 9/11.
- [26] Armen Der Kiureghian and Ove Ditlevsen. 2009. Aleatory or epistemic? Does it matter? *Structural Safety* 31, 2 (March 2009), 105–112. <https://doi.org/10.1016/j.strusafe.2008.06.020>
- [27] P Klopogge, J. van der Sluijs, and A Wardekker. 2007. *Uncertainty communication: issues and good practice*. Utrecht University, Utrecht. <http://edepot.wur.nl/62349> OCLC: 1012567140.
- [28] Hermann G. Matthies. 2007. QUANTIFYING UNCERTAINTY: MODERN COMPUTATIONAL REPRESENTATION OF PROBABILITY AND APPLICATIONS. In *Extreme Man-Made and Natural Hazards in Dynamics of Structures*, Adnan Ibrahimbegovic and Ivica Kozar (Eds.). Springer Netherlands, Dordrecht, 105–135. https://doi.org/10.1007/978-1-4020-5656-7_4
- [29] Peter J. Mohr, David B. Newell, and Barry N. Taylor. 2016. CODATA Recommended Values of the Fundamental Physical Constants: 2014. *Journal of Physical and Chemical Reference Data* 45, 4 (Dec. 2016), 043102. <https://doi.org/10.1063/1.4954402>
- [30] M. Granger Morgan. 2003-03. Characterizing and Dealing With Uncertainty: Insights from the Integrated Assessment of Climate Change. 4, 1 (2003-03), 46–55. <https://doi.org/10.1076/iaij.4.1.46.16464>
- [31] Richard H Moss and Stephen H Schneider. 2000. UNCERTAINTIES IN THE IPCC TAR: Recommendations To Lead Authors For More Consistent Assessment and Reporting. (2000), 19.
- [32] Todd Mytkowicz, Amer Diwan, Matthias Hauswirth, and Peter F Sweeney. 2009. Producing Wrong Data Without Doing Anything Obviously Wrong! (2009), 12.

- [33] Nate Silver. 2017. *The Real Story Of 2016*.
- [34] National Oceanic and Atmospheric Administration. 2019. Hurricane Lorenzo remains a very powerful system – hurricane watch for dangerous winds and major waves has been issued for the Azores. <http://www.severe-weather.eu/tropical-weather/hurricane-lorenzo-remains-a-very-powerful-system-hurricane-watch-for-dangerous-winds-and-major-waves-has-been-issued-for-the-azores/>
- [35] Populations. 2019. Population Pyramids of the World from 1950 to 2100. <https://www.populationpyramid.net/world/2017/>
- [36] Jackson Rawlings. 2018. The 17 Cognitive Biases That Explain Brexit. <https://medium.com/the-politicalists/the-17-cognitive-biases-that-explain-brexite-894ec10e03b8>
- [37] Tobias Renzler, Michael Spörk, Carlo Alberto Boano, and Kay Römer. 2018. Improving the efficiency and responsiveness of smart objects using adaptive BLE device discovery. In *Proceedings of the 4th ACM MobiHoc Workshop on Experiences with the Design and Implementation of Smart Objects - SMARTOBJECTS '18*. ACM Press, Los Angeles, California, 1–10. <https://doi.org/10.1145/3213299.3213306>
- [38] Pere Ríos. 2014. Catalan public opinion swings toward “no” for independence, says survey. *El País* (Dec. 2014). https://elpais.com/elpais/2014/12/19/inenglish/1419000488_941616.html
- [39] Nate Silver. 2016. 2016 Election Forecast. <https://projects.fivethirtyeight.com/2016-election-forecast/>
- [40] Susan Solomon, Intergovernmental Panel on Climate Change, and Intergovernmental Panel on Climate Change (Eds.). 2007. *Climate change 2007: the physical science basis: contribution of Working Group I to the Fourth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, Cambridge ; New York. OCLC: ocn132298563.
- [41] Michael Spörk, Carlo Alberto Boano, Marco Zimmerling, and Kay Römer. 2017. BLEach: Exploiting the Full Potential of IPv6 over BLE in Constrained Embedded IoT Devices. In *Proceedings of the 15th ACM Conference on Embedded Network Sensor Systems - SenSys '17*. ACM Press, Delft, Netherlands, 1–14. <https://doi.org/10.1145/3131672.3131687>
- [42] T.seppelt. 2015. Opinion polling for the United Kingdom European Union membership referendum. https://commons.wikimedia.org/wiki/File:UK_EU_referendum_polling.svg
- [43] Edward R. Tufte. 2001. *Visual Display of Quantitative Information* (2 ed.). Bertrams, Cheshire, Conn.
- [44] John Wilder Tukey. 1977. *Exploratory data analysis*. Addison-Wesley Pub. Co, Reading, Mass.
- [45] United Nations. 2019. World Population Prospects - Population Division - United Nations. <https://population.un.org/wpp/Graphs/DemographicProfiles/>
- [46] OAR US EPA. [n.d.]. Future of Climate Change. [/climate-change-science/future-climate-change](https://climate-change-science/future-climate-change)
- [47] Colin Ware. 2010. *Visual Thinking: for Design* (1 ed.). Morgan Kaufmann.
- [48] Charles Weiss. 2006-10. Can there be science-based precaution? 1, 1 (2006-10), 014003. <https://doi.org/10.1088/1748-9326/1/1/014003>
- [49] Wikipedia. 2019. Catalan independence movement. https://en.wikipedia.org/w/index.php?title=Catalan_independence_movement&oldid=917342498 Page Version ID: 917342498.
- [50] XENON Collaboration. 2019. Observation of two-neutrino double electron capture in ^{124}Xe with XENON1T. *Nature* 568, 7753 (April 2019), 532–535. <https://doi.org/10.1038/s41586-019-1124-4>

A.7 [P7] Assessing Risk Estimations for Cyber-Security Using Expert Judgment

- Authors: **Michael Krisper**, Jürgen Dobaj, Georg Macher
- Year: 2020
- DOI: 10.1007/978-3-030-56441-4_9
- Bibliography-Reference: [162]
- Presented at EuroSPI 2020 conference in Düsseldorf, Germany.
- Published in the EuroSPI 2020 conference proceedings (Springer, CCIS, volume 1251).

Summary This paper presents a feasibility study for cybersecurity assessment using the RISKEE method and structured expert judgment. It elaborates on a user study that was done at the EuroSPI 19 conference and involved several experts in judging the risk of a specific cyber-security attack. The experts assessed a known attack on the passive keyless entry systems (PKES) of Tesla Model S cars. This was done in three phases: First, the experts answered calibration questions used to assess their judgment capabilities. Afterwards, the experts judged the cyber-security risk of the actual case twice: The first round was individual judgment, and the second round was done after discussions and knowledge exchange among the experts. The key findings of this user study were:

- Doing a calibration round reveals the judgment capabilities of individual experts and allows for weighting them accordingly.
- Most of the experts in this study were bad regarding their judgment accuracy (calibration score), which was surprising. Without a calibration round, we would have never expected this.
- In the second round, most of the attributes were judged higher, which, in the end, resulted in doubling the calculated risk compared to the first round. In this case, the discussion increased the risk sensitivity of the experts, which led to a higher estimation of the risks.
- RISKEE can be used on risk graphs, and the calculation of risks by forward and backward propagation works in a real case scenario.

This paper was important for showing that the applied method for structured expert judgment used in RISKEE was feasible. It also coincides with previous studies done by other researchers, which showed that the majority of experts (around $\frac{2}{3}$) have bad judgment capabilities, and their judgments should be rejected to increase the quality of the results. We did not take this route since, in our case, it would have meant that only two or three out of the 17 experts would have been counted, which we deemed as being too selective. Therefore we did not sieve them out but just weighted them accordingly to their calibration performance.

My Contribution This paper was completely written and conceived by me. The contribution of the other authors was in the form of discussions, feedback for improvements, and corrigenda, as well as assistance during the feasibility study at the conference.

Copyright ©2020 Springer Nature Switzerland AG. The version included in this dissertation is a permitted reprint of the published version from Proceedings of the European & Asian System, Software & Service Process Improvement & Innovation (EuroAsiaSPI), September 2019, which can be accessed via SpringerLink: https://link.springer.com/chapter/10.1007/978-3-030-56441-4_9

Assessing Risk Estimations for Cyber-Security Using Expert Judgment

Michael Krisper^(), Jürgen Dobaj, and Georg Macher

Graz University of Technology, 8010 Graz, Austria
{michael.krisper,juergen.dobaj,georg.macher}@tugraz.at

Abstract. In this paper, we show a use-case of structured expert judgment to assess the risk of a cyber-security attack. We showcase the process of eliciting unknown and uncertain values using multiple experts and combining these judgments by weighing the experts based on their performance. The performance of an expert is assessed using the information and calibration score calculated from the judgments of calibration questions. The judgments are stated with three-points estimates of minimum, most likely, and maximum value, which serve as input for the PERT probability distribution. For the use-case, the input values frequency, vulnerability, and impact were asked. The combined results are propagated along an attack path to calculate the risk of a cyber-security attack. This was done using RISKEE, a tool for assessing risk in cyber-security and implementing the combination of expert judgments and propagation of the values in an attack-tree. It uses an attack graph to model the attack paths and applies probability distributions for the input values to consider the uncertainty of predictions and expert judgments. We also describe experiences and lessons-learned for conducting an expert elicitation to acquire input values for estimating risks in cyber-security.

Keywords: Risk assessment · Expert elicitation · Cyber-security · Expert judgment · Probabilistic methods

1 Introduction

Assessing the risk of cyber-security becomes more and more important nowadays. With the rise of IoT (Internet of Things) and the evergrowing number of devices connected to the internet, the possibility of hacker attacks is ever increasing. Smart housing, connected vehicles, and the general trend of all devices being able to communicate via wireless technologies (like WiFi, Bluetooth, Zigbee), increase the complexity tremendously. Besides being convenient and useful to the user and operators, this high connectivity at the same time widens the attack surface and invites hackers to use exploits in order to gain control over the devices. With such control, hackers could steal data, money, and identities, install ransomware,

deploy botnets, or even destroy hardware. This goes as far as committing terrorist attacks if they target critical infrastructures like the power grid, medical infrastructures, public transportation, or governmental institutions.

The problem is that risk for cyber-security is immensely difficult to estimate and even harder to predict for the future [25, 42]. In safety, experts can extrapolate from past events and apply statistical methods to calculate future risk, cyber-security risk is behaving more chaotic and, therefore, can not be easily modeled [12]. The last resort, after not being able to rely on historical or reliability data, is to fall back to expert judgment. However, this also has its flaws. Humans have cognitive biases [18, 33, 34] and especially regarding probabilities we are really bad in estimating [23, 24]. It was even shown that the majority of experts have terrible skills in giving quantitative judgments [3]. The topic of expert elicitation and how to get the best out of the combination of several expert judgments in the form of opinion pools is discussed for many years now [7, 10]. We tackle this topic by combining several techniques from different domains and applying it to the cyber-security domain. We can calculate the risk of attacks by using attack trees and combining them with probability distributions and uncertainty propagation for the properties of attack frequency, attack probability, and impact. Furthermore, we apply structured expert judgment using the IDEA protocol [19, 20] having two rounds of expert elicitation and doing performance-based weighting based on previously asked calibration questions. By doing so, we get a realistic image of risk, based on the answers of the best experts and can make better-informed decisions based on that.

2 Background and Related Work

Since the upcoming of the first computers, cyber-security played an important role [47]. Bruce Schneier was one of the cyber-security pioneers coining the term attack trees [46]. From this idea on the method of Attack Tree Analysis (ATA) [2, 48] was developed and from that several other ways to describe multi-step cyber attacks were invented, like the Kill Chain [26] or Bayesian Attack Graphs [43]. In our work, we build upon this technique of modeling attacks in an attack graph by describing every step an attacker has to undertake to reach a goal. Furthermore, we combine that with quantifiable risk values like the factor analysis of information risk (FAIR) [17] does. For the evaluation of the survey in this paper, we developed and used the tool RISKEE [36] (coined from the fundamental structure of our approach: the risk tree). This is an ongoing approach to create a method for integrating safety and security risk in a holistic way [13].

Still open was the question of how to get the actual input data. Although there are many existing databases about security vulnerabilities like the Common Vulnerability Scoring System (CVSS) [15, 39] or Common Criteria, and there are approaches on extracting the attack probabilities from them [21], it is still difficult to apply them to whole systems and calculate the risk for multistep attacks [25]. There are many standards for risk assessment, which we looked into: NIST SP-800 [32], NIST CSF [41], ISO/IEC 27000 [29, 31], ISO 31000 [30],

EVITA [44], STRIDE [22,35], COBIT [28], OCTAVE [1], SAHARA [38], and FMVEA [45]. They model risk mainly in a qualitative way using ordinal or semi-quantitative scales, which was not sufficient for our purposes. Cox et al. proved that using such scales can lead to severe problems and has many pitfalls [9,10]. That is why we fell back to the last resort: Expert judgment. We found a robust method in Structured Expert Judgment (SEJ) [7,8] by Roger Cooke from TU Delft. It works by measuring the judgment performance of experts by asking them calibration questions and calculating a calibration score (how well did the expert catch the correct value) and information score (how big or small is the uncertainty in the judgments). These scores define the weight an expert should get for the combination into a linear opinion pool.

3 Method

In this section, we describe our applied method for expert elicitation. It is based on structured expert judgment (SEJ) [7] and the IDEA elicitation procedure [19,20]. IDEA stands for Investigate, Discuss, Estimate, and Aggregate and uses the basic principles of the Delphi method [37]. In this method, probability-based estimations and calibration questions are used to calculate performance-based weights for the individual experts. Based on these weights, their actual judgments are combined accordingly to get consolidated results. These results are discussed, and a second elicitation round is undertaken to get refined results. This method combines the best from both worlds: mathematical performance-based combinations and behavioral group-think mechanisms.

In September 2019, we tested this method during a conference workshop to judge the risk of a specific cyber-security incident. This workshop was held at the EuroSPI 2019 conference in Edinburgh [51]. RISKEE, the tool used for this workshop, was also presented at this conference [36]). In total, 21 participants with mixed backgrounds and domains participated in this workshop (experts and laypeople from many domains, e.g., cyber-security, safety, automotive, medical, industry, and academia). Of those 21 participants, only 17 were eligible for being accepted, because four either did not do the calibration questions or did not finish the elicitation (due to either arriving after the calibration phase or leaving the room earlier).

4 Information and Calibration Score

In SEJ, two scores are used for judging the performance of an expert: information score and calibration score [5]. The information score assesses the uncertainty of a given expert (his “precision”), and the calibration score, how well the true value was hit (the “accuracy”). These are calculated based on e.g., three-point estimates stating the 90% confidence interval via the minimum and maximum plausible value, and the most likely value. These three estimates define a probability distribution with five areas for the true value to fall into: with 5% probability it should be lower than the minimum; with 45% probability, it should be between

the minimum and the most likely, and equally with 45% between the most likely and the maximum. The remaining 5% of cases it should be larger than the maximum value. Inside these areas, SEJ assumes that the probability is uniformly distributed. This would be the ideal “reference expert”. A log-likelihood χ^2 -test (also known as G-test) is used to compare this to the actual realizations of an expert. The result is the calibration value and signifies how well the expert fits the assumed reference expert. For our method, however, we had to make some changes, summarized as follows: Instead of uniform areas, we decided to take the PERT probability distribution for the three-point estimate. The distribution is used heavily in project management to predict uncertain time schedules [50]. Albeit the choice for PERT, our method is not limited to this and works with other distributions also. We use the distribution to get the probability with which the true value was predicted. Combining all realizations of the calibration questions, the resulting histogram H should correlate to a normal distribution \mathcal{N} . This is again tested via the log-likelihood χ^2 -test to calculate the calibration score, similar to how it was done in SEJ. Ultimately this results in calculating the Kullback-Leibler distance and applying this to the χ^2 distribution to get the model fit.

$$\text{Calibration} = 1 - \chi^2 \left(2 \sum H_i \log \frac{H_i}{\mathcal{N}_i} \right) \quad (1)$$

Moreover, the information score is calculated by calculating the entropy of given estimation compared to a uniform background distribution over a reasonably large range.

$$\text{Information} = \sum_{\text{Questions}} \frac{- \left(\sum \mathcal{P}_i \log \frac{\mathcal{P}_i}{\mathcal{U}_q} \right)}{\log (\mathcal{U}_{max} - \mathcal{U}_{min})} \quad (2)$$

Calibration and information scores are then multiplied to get the final combined score. The resulting weight is then the normalized value over the sum of combined scores for all experts.

$$\text{Combined Score} = \text{Calibration} \cdot \text{Information} \quad (3)$$

This combination now allows us to compare the expert to each other. Experts with higher calibration and information scores get higher weights, which is what we wanted.

4.1 Procedure

The whole workshop took approximately one hour and was split up in the following steps:

1. **Introduction** (10 min) We described the general process of expert judgment using distributions (“thinking in ranges” instead of single-point estimates) and explained some basic information about risk estimation. Every participant got a unique anonymous identifier to trace the different responses for each one of them.

2. **Calibration questions** (10 min) The calibration was done with a prepared questionnaire that should be filled out by the participants.
3. **Scenario explanation** (10 min) We explained the scenario of an imaginary car rental company, which could be the victim of a key-fob-hack [52]. The actual scenario was: Imagine being the CISO (chief information security officer) of a car rental company, having a fleet of 100 Tesla Model S cars in the beautiful city of Leuven (the city of the COSIC research group). What are the risks regarding the previously mentioned key-fob hack?
4. **First Round of expert elicitation** (5 min) Participants had to fill in the attack frequency, the attack success probability, and the resulting impact of a successful attack in the form of [minimum-mode-maximum]. This was done individually and without prior discussion.
5. **Discussion** (10 min) and **interpreting the results** (5 min) During the discussion, we clarified the scenario even more and let the participants exchange opinions and arguments. In the meantime, we analyzed the first round and showed the results to the participants.
6. **Second round of expert elicitation** (5 min) Participants were asked to judge the same scenario as in the first round, but now with their revised judgments.

4.2 Visualisation of Results: Guppy Plots

A *guppy plot*¹ is a specialized violin plot, which shows the responses of all experts in the form of probability distributions drawn horizontally along the x-axis. Figure 1 shows an example for this. The experts are listed along the y-axis, and the whole value range is displayed along the y-axis, which is scaled either linearly or logarithmically, depending on how different the magnitudes of the responses are. If the guppy plot is used to show calibration questions and answers, the true value is also indicated using a vertical line (or area) as an overlay. More examples for this in a simpler form (as vertical lines plots) can be found in [6, 20].

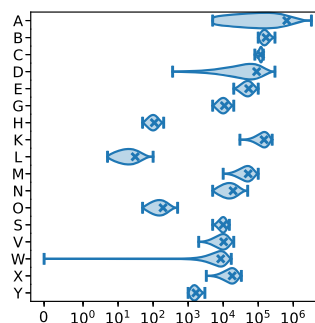


Fig. 1. An example guppy plot on a logarithmic scale.

4.3 Calibration Questions

The calibration questions have the purpose of examining the judgment quality of the experts. Based on the judgment quality, the assessor can calculate weights for the individual experts to combine the answers for the real questions in a

¹ The term “guppy plot” was coined by the wife of one of the authors, since they look like a group of guppy fish swimming in a fish tank. This is especially recognizable on logarithmic scales.

reasonable way. The calibration questions must fulfill three criteria to get a reasonable weighting of experts:

Firstly, they have to be on the same topic and ask for approximately the same facts as the real survey questions, so that the expert needs to apply the same know-how and skills to answer the real questions. This is important to assure that the performance of the calibration questions is transferable to the performance of the real survey questions.

Secondly, the answer must be unknown to the experts at the time of the elicitation. This is important because if the expert already knows the answers, the calibration does not assess the appropriate skills, and the weights are not applicable. If an expert already would know the answers, the elicitation would not be needed.

Thirdly, the calibrations questions should involve some uncertainty in order to assess their ability to judge it. Since the real survey questions inherently also involve uncertainty, it is essential to assess the skills of estimations and dealing with uncertain predictions.

5 Results of the Calibration

In this section, we visualize and discuss the results of the calibration questions and the actual survey results. Figure 2 shows the scores of calibration and information as well as the combined score for the participants. This combined score defines the weights to be applied for the actual survey results. The final weights are shown in Table 1.

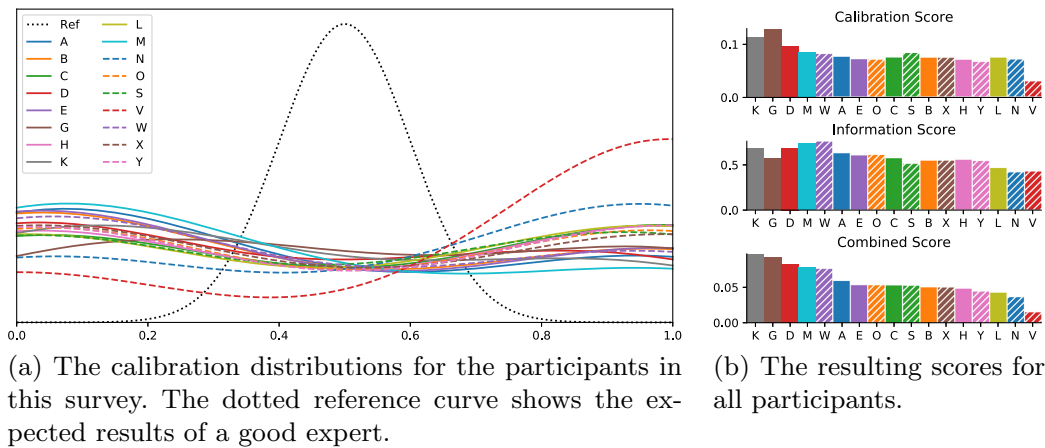


Fig. 2. The graph shows the performance of the participants during the calibration.

The results showed that the participants were uncertain and somewhat inadequate at estimating. This is of no surprise because Colson et al. [3,4] already showed that most experts are wrong at estimating. Furthermore, the survey participants had many different backgrounds and knowledge; therefore, it is quite

natural to have vastly different opinions and responses. The reference curve in Fig. 2a depicts how the responses by a good participant are expected to look like. A good expert is one who hits the most-likely value most of the time. It can be seen that the participants in our survey did not match this expectation. The calibration distribution also shows which participants tend to over- or underestimate the true values.

Table 1. The weights for the participants in decreasing order. These weights were calculated based on the performance on the calibration questions.

K	G	D	M	W	A	E	O	C	S	B	X	H	Y	L	N	V
0.10	0.09	0.08	0.08	0.08	0.06	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.04	0.04	0.02

Calibration Results for Attack Frequency. The results for the calibration on attack frequency are visualized in Fig. 4. They show that the surveyed participants greatly disagreed and are very uncertain about the actual values, which resulted in quite a disperse distribution in the end. Participants V, S, and O have a zero calibration score since they never hit the correct value. Therefore their combined score also results in zero. The other participants all had approximately the same calibration and information scores (as can be seen in Fig. 3). The questions were the following:

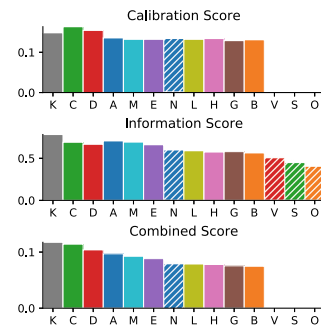


Fig. 3. The scores for the attack frequency.

- Q1 How many cars are stolen per year in the UK? (3.3 Million registered cars) → 114 660 (Source: Statista [11])
- Q2 How many people were affected by the Cambridge Analytica Facebook data leak in 2018? → 86 600 000 (Source: Facebook [40])
- Q3 How many bigger healthcare-specific data breaches happened in 2018 in the US? → 371 (Source: U.S. Department of Health and Human Services [49])

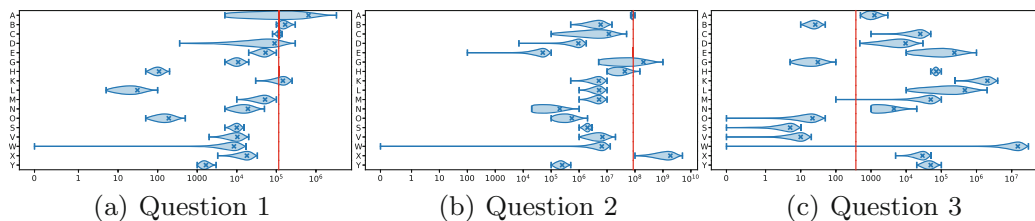


Fig. 4. The responses to the first three calibration questions, which assess the attack frequency showing rather small uncertainties but great disagreement amongst the participants.

Calibration Results for the Vulnerability. The results for the vulnerability calibration questions again show that participants are quite uncertain about the values. Furthermore, the resulting calibration scores in Fig. 5 show that no participant was significantly better than the other, as all calibration scores are about the same at the value range of 0.1. However, regarding the information score, they differed. Participant W and K had slightly better results than the others. The resulting combined score is still a quite linear slope from the best participants W and K to the worst participants N and H. Due to space limitations, we left out the visualization of the calibration questions. In summary, they showed huge uncertainty amongst the participants.

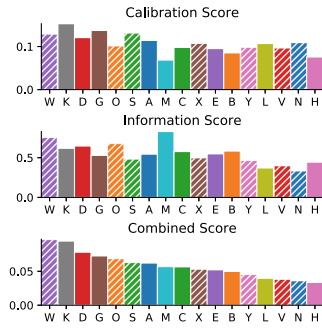


Fig. 5. The scores for the vulnerability questions.

Calibration Results for the Impact. Figure 7 shows some of the responses for the calibrations questions regarding the impact. It can again be seen that the participants are quite uncertain about the judgments (especially participant W). Three participants (W, V, S) did not have calibration scores since they never hit the actual value. For the others, the scores were quite evenly distributed (see Fig. 6). Interestingly, when looking at the combined score, the participants G, K, and M had slightly better scores than the others. The visualization for the impact was often heavily distorted by the difference in magnitudes of values (even when using logarithmic scales). Here are some questions which have been asked during the calibration:

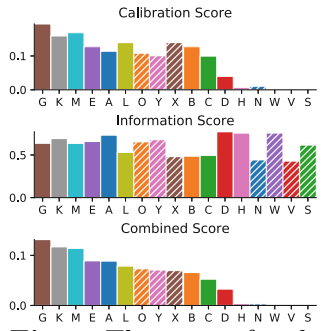


Fig. 6. The scores for the impact calibration.

- Q12 What is the price for a new Tesla Model S car? → 99000 (Source: Energysage [14])
- Q13 What is the cost of replacing a compromised electronic door lock in an office building? → € 270 (Source: Fixr [16])
- Q14 What is the estimated cost per record of a leaked user-sensitive data entry? → €135 (Source: IBM/Ponemon [27])

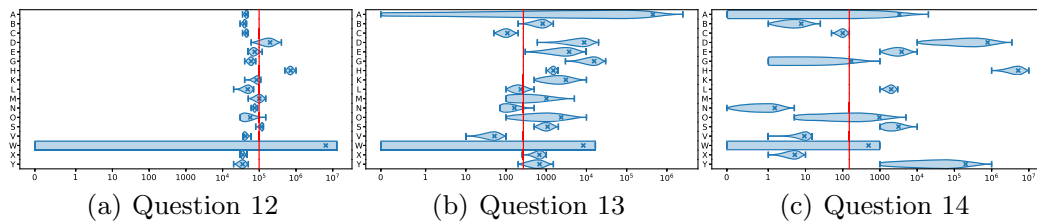


Fig. 7. The first three responses to the calibration questions about impact, showing a high variety of outcomes.

6 Results of the Risk Survey

Attack Frequency. The attack frequency defines how many attack attempts are made per timeframe (per year in our case). On comparing the results of the first elicitation round to the second, it can be concluded that the participants increased their minimum slightly and most-likely greatly but also significantly lowered the overall maximum of attacks per year after the discussions. This leads to a much higher concentration in the upper quantiles of the attack frequency distribution. Table 2 shows the actual values.

Table 2. The expected attack frequency for the entry node.

Round	Attack Frequency
1	2 – 34 – 415
2	5 – 100 – 332

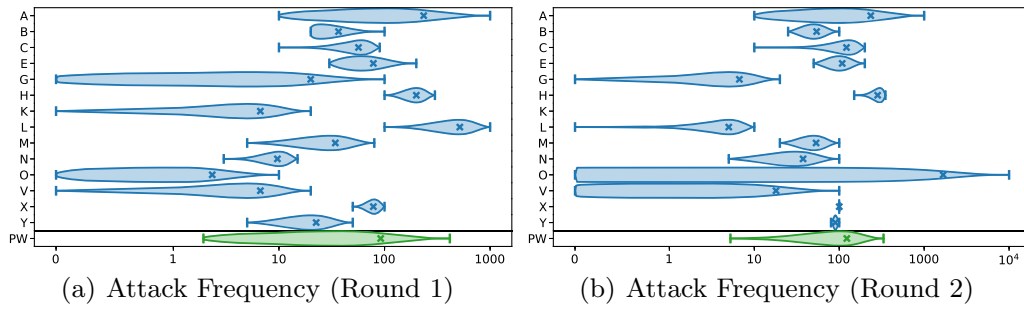


Fig. 8. The estimations of attack frequency for the two elicitation rounds. The PW row on the bottom is the performance-weighted combined score.

Vulnerability. The vulnerability was assessed for four nodes along the attack path. The resulting values are shown in Table 3. In the second round, the minimum and most-like values were increased, and the maximum estimation decreased slightly. This concentrated the density more in the higher quantiles of the distribution.

Table 3. The estimated vulnerabilities in percent for all four nodes along the attack path.

Node	First round	Second round
1	4.3 – 69.8 – 98.8	5.6 – 82.0 – 95.9
2	2.6 – 48.2 – 98.8	4.6 – 55.9 – 92.1
3	2.9 – 40.7 – 98.8	4.4 – 46.7 – 98.9
4	1.3 – 15.7 – 96.7	3.1 – 14.4 – 96.9

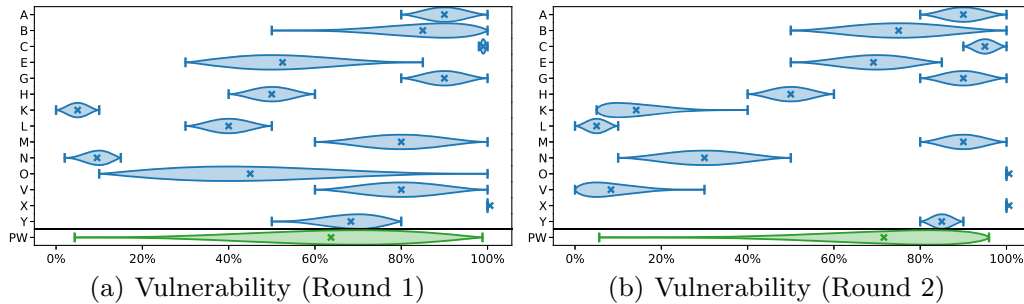


Fig. 9. The estimations of the vulnerability for the two elicitation rounds. The PW row on the bottom is the performance-weighted combined score. For the sake of conciseness and space only one of the four responses is shown.

Impact. The impact defines the financial loss in the case the whole attack is executed successfully, which in our case meant that a car gets stolen. In the second round, the participants increased their estimations, which led to an increase in the minimum value, most-like value (more than double), and maximum value (Table 4).

Table 4. The expected impact in Euro if the whole attack is successful. The values are rounded for simplification sake.

Round	Impact
1	€ 780 – 23 500 – 380 000
2	€ 1000 – 78 000 – 477 000

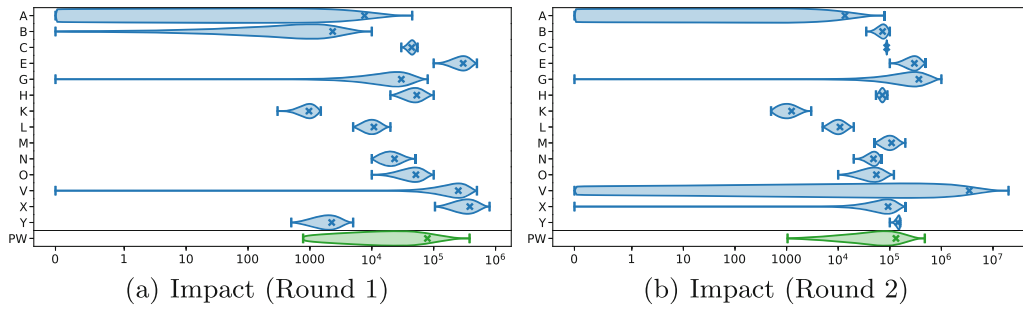


Fig. 10. The estimation of the impact for the two elicitation rounds. The PW row on the bottom is the performance-weighted combined score.

6.1 Final Risk Results

After weighing the participants by their performance during the calibration and using these values to calculate the risk along the risk tree with RISKEE, the final results depict the total risk of this cyber-security use-case. The results are shown in Fig. 11 and Fig. 12: The expected attack frequency will around 123 attacks per year (node 1), and the loss impact of a single attack being successful is around €131 800 (node 4).

Regarding the expected risk, there is a 95% chance that the loss will exceed approx. €24 000, the average expected loss will be around €868 800 (having a chance of 30%), and there is a 5% chance that the loss will even exceed around €3 Million. The value of having a 50% chance of exceeding is around €420 500, which is less than half of the average value. Furthermore, of the four attack steps, the first one is the most vulnerable (with 72% probability), while the last one is assumed to be the most difficult (with just 26% probability). This can be explained by the fact that in the first step, the attacker had only to be in the vicinity of the car, while in the last step the attacker has to unlock the car, enter it, start it, and drive away, which would be much more noticeable than to pass by.

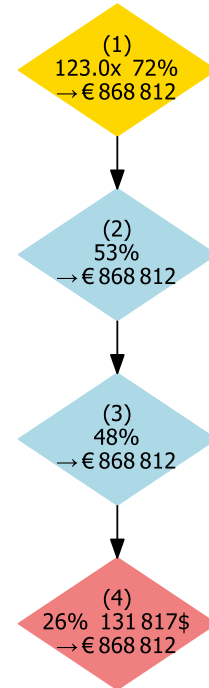


Fig. 11. The attack graph.

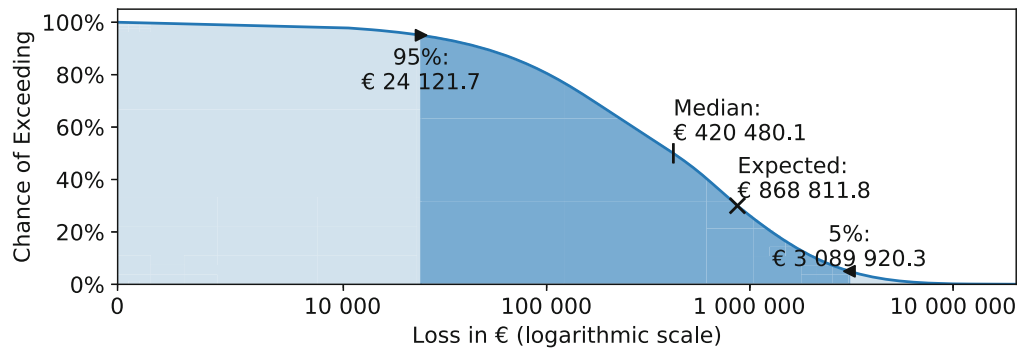
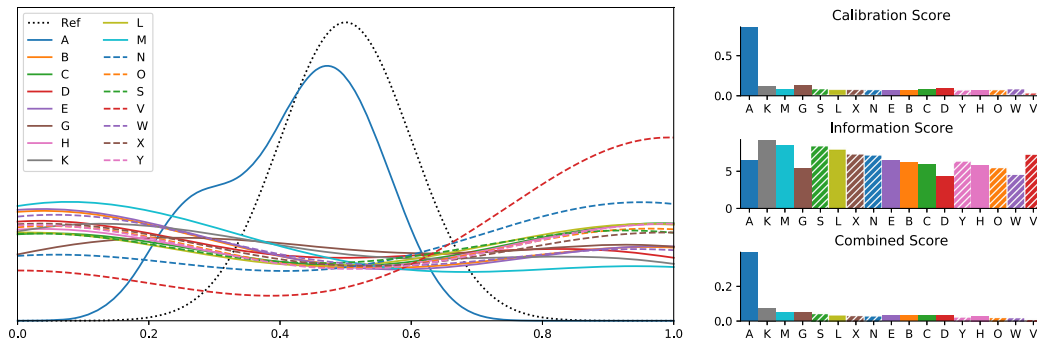


Fig. 12. The loss exceedance curve shows the probabilities of exceeding a certain amount of money along the whole range of possible outcomes. The values for the 95%, the median, the 5%, and the expected average are shown.

7 Discussion

In this survey, the scores are evenly distributed, which means that the participants performed equally on the calibration questions. The only problem here is that the calibration scores were very low. Figure 13 shows how this survey would have looked like if at least one participant had excellent responses. It can be seen that the weight for this participant would have been much higher than the other participants, and in such a way, this one would have overruled the other participants, which is desired behavior. A good expert should get more weight than a bad expert.



(a) The calibration distribution for the responses. A good participant would approximate the reference curve.

(b) The calibration score and combined score for the good participant dwarfs the scores of the other participants.

Fig. 13. The responses of participant A with artificial data representing the expected responses. It shows that a good expert would significantly overrule the other participants.

We assessed the participants' general estimation capability by combining all 14 calibration questions to get calibration and information scores. During analyzing the data, we realized that it would also have been interesting to assess the performance of the individual areas (attack frequency, vulnerability, and impact). For future surveys, we plan to assess the individual areas in more detail.

8 Conclusion

In this paper, we described a use-case of an expert elicitation for the assessment of cyber-security risks. We applied the method of structured expert judgment to get the input values, and used the tool RISKEE for uncertainty and risk propagation on risk trees to calculate the resulting loss expectancy. This served as a feasibility study to see if the method applies to real-world use-cases for estimation of risk in cyber-security. Our results support the argument of existing literature that people are bad in estimating uncertain values. Furthermore, they show that a risk assessment approach is feasible and can be done quite easily. The resulting loss exceedance curve proved to be an appropriate visualization of risk, because it showed a fully informative view over the whole risk space, while still being easy to comprehend.

In future work, we intend to improve the tools for the elicitation to streamline input and usage. Furthermore, we want to elaborate on different aspects of the risk analysis, e.g., modeling risk over time, or improving the results even more by de-biasing the experts based on their calibration which could ultimately pave the way for a self-calibrating expert judgment method that corrects the biases and errors of experts as good as possible to get even better prediction results.

References

1. Alberts, C., Dorofee, A., Stevens, J., Woody, C.: Introduction to the octave approach: Technical report, Defense Technical Information Center, Fort Belvoir, VA, August 2003. <https://doi.org/10.21236/ADA634134>. <http://www.dtic.mil/docs/citations/ADA634134>
2. Ammann, P., Wijesekera, D., Kaushik, S.: Scalable, graph-based network vulnerability analysis. In: Proceedings of the 9th ACM Conference on Computer and Communications Security - CCS 2002, p. 217. ACM Press, Washington (2002). <https://doi.org/10.1145/586110.586140>
3. Colson, A.R., Cooke, R.M.: Cross validation for the classical model of structured expert judgment. *Reliab. Eng. Syst. Saf.* **163**, 109–120 (2017). <https://doi.org/10.1016/j.ress.2017.02.003>
4. Colson, A.R., Cooke, R.M.: Expert elicitation: using the classical model to validate experts' judgments. *Rev. Environ. Econ. Policy* **12**(1), 113–132 (2018). <https://doi.org/10.1093/reep/rex022>. <https://academic.oup.com/reep/article/12/1/113/4835830>
5. Colson, A.R., et al.: Quantifying uncertainty about future antimicrobial resistance: Comparing structured expert judgment and statistical forecasting methods. *Plos One* **14**(7), e0219190 (2019). <https://doi.org/10.1371/journal.pone.0219190>

6. Cooke, R.M.: Quantifying uncertainty on thin ice: expert judgement assessment. *Nat. Clim. Change* **3**(4), 311–312 (2013). <https://doi.org/10.1038/nclimate1860>. <http://www.nature.com/articles/nclimate1860>
7. Cooke, R.M.: *Experts in Uncertainty: Opinion and Subjective Probability in Science*. Environmental Ethics and Science Policy Series. Oxford University Press, New York (1991)
8. Cooke, R.M., Goossens, L.L.: TU Delft expert judgment data base. *Reliab. Eng. Syst. Saf.* **93**(5), 657–674 (2008). <https://doi.org/10.1016/j.res.2007.03.005>
9. Cox, A.L.: What’s wrong with risk matrices? *Risk Anal.* **28**(2), 497–512 (2008). <https://doi.org/10.1111/j.1539-6924.2008.01030.x>
10. Cox, L.A.: *Risk Analysis of Complex and Uncertain Systems*. International Series in Operations Research & Management Science, vol. 129. Springer, New York (2009). <https://doi.org/10.1007/978-0-387-89014-2>
11. Clark, D.: Number of motor vehicle theft offences recorded in England and Wales from 2002/03 to 2018/19, July 2019. <https://www.statista.com/statistics/303551/motor-vehicle-theft-in-england-and-wales/>
12. Deb, A., Lerman, K., Ferrara, E.: Predicting cyber events by leveraging hacker sentiment. *Information* **9**(11), 280 [arXiv: 1804.05276](https://arxiv.org/abs/1804.05276) (2018). <https://doi.org/10.3390/info9110280>
13. Dobaj, J., Schmittner, C., Krisper, M., Macher, G.: Towards integrated quantitative security and safety risk assessment. In: Romanovsky, A., Troubitsyna, E., Gashi, I., Schoitsch, E., Bitsch, F. (eds.) *SAFECOMP 2019*. LNCS, vol. 11699, pp. 102–116. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-26250-1_8
14. EnergySage: What are typical Tesla car prices? Model S, Model X crossover and Model 3 costs explained. <https://news.energysage.com/how-much-does-a-tesla-cost/>
15. FIRST: Common Vulnerability Scoring System version 3.1 Revision 1 (2019). https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf
16. FixR: Smart Lock Installation Cost. <https://www.fixr.com/costs/smart-lock-installation>
17. Freund, J.: *Measuring and Managing Information Risk: A FAIR Approach*, p. 00000. Butterworth-Heinemann, Amsterdam (2015)
18. Gilovich, T., Griffin, D.W., Kahneman, D. (eds.): *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge University Press, Cambridge, New York (2002)
19. Hemming, V., Burgman, M.A., Hanea, A.M., McBride, M.F., Wintle, B.C.: A practical guide to structured expert elicitation using the IDEA protocol. *Methods Ecol. Evol.* **9**(1), 169–180 (2018). <https://doi.org/10.1111/2041-210X.12857>
20. Hemming, V., Walshe, T.V., Hanea, A.M., Fidler, F., Burgman, M.A.: Eliciting improved quantitative judgements using the IDEA protocol: a case study in natural resource management. *Plos One* **13**(6), e0198468 (2018). <https://doi.org/10.1371/journal.pone.0198468>
21. Houmb, S.H., Franqueira, V.N., Engum, E.A.: Quantifying security risk level from CVSS estimates of frequency and impact. *J. Syst. Softw.* **83**(9), 1622–1634 (2010). <https://doi.org/10.1016/j.jss.2009.08.023>
22. Howard, M., LeBlanc, D.: *Writing Secure Code*, 2nd edn. Microsoft Press, Redmond (2003)
23. Hubbard, D.W.: *The Failure of Risk Management: Why It’s Broken and How to Fix it*. Wiley, Hoboken (2009). oCLC: ocn268790760
24. Hubbard, D.W., Seiersen, R.: *How to Measure Anything in Cybersecurity Risk*, p. 00000. Wiley, Hoboken (2016)

25. Husak, M., Komarkova, J., Bou-Harb, E., Celeda, P.: Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Commun. Surv. Tutor.* **21**(1), 640–660 (2019). <https://doi.org/10.1109/COMST.2018.2871866>. <https://ieeexplore.ieee.org/document/8470942/>
26. Hutchins, E.M., Cloppert, M.J., Amin, R.M.: Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains, p. 14, January 2011
27. IBM: How much would a data breach cost your business? (2019). <https://www.ibm.com/security/data-breach>
28. ISACA: COBIT — Control Objectives for Information Technologies — ISACA (2019). <https://www.isaca.org/resources/cobit>
29. ISO: ISO 27000 - ISO 27001 and ISO 27002 Standards (2019). <http://www.27000.org/>
30. ISO/IEC: ISO 31000:2018 Risk management - Guidelines (2018). <https://www.iso.org/iso-31000-risk-management.html>, 00000
31. ISO/IEC: ISO/IEC 27000:2018 (2018)
32. Joint Task Force Transformation Initiative: Guide for conducting risk assessments. Technical report. NIST SP 800–30r1, National Institute of Standards and Technology, Gaithersburg, MD (2012). <https://doi.org/10.6028/NIST.SP.800-30r1>, <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf>
33. Kahneman, D., Frederick, S.: Representativeness revisited: attribute substitution in intuitive judgment. In: Gilovich, T., Griffin, D., Kahneman, D. (eds.) *Heuristics and Biases*, pp. 49–81. Cambridge University Press, 1 edn. (2002). <https://doi.org/10.1017/CBO9780511808098.004>
34. Kahneman, D., Tversky, A.: Subjective probability: a judgement of representativeness. *Cogn. Psychol.* **3**(3), 430–454 (1972)
35. Kohnfelder, L., Garg, P.: The threats to our products, p. 8 (1999)
36. Krisper, M., Dobaj, J., Macher, G., Schmittner, C.: RISKEE: a risk-tree based method for assessing risk in cyber security. In: Walker, A., O’Connor, R.V., Messnarz, R. (eds.) *EuroSPI 2019*. CCIS, vol. 1060, pp. 45–56. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-28005-5_4
37. Linstone, H.A., Turoff, M. (eds.): *The Delphi Method: Techniques and Applications*. Addison-Wesley, Reading [usw.] (1975). oCLC: 251991541
38. Macher, G., Sporer, H., Berlach, R., Armengaud, E., Kreiner, C.: SAHARA: a security-aware hazard and risk analysis method, pp. 621–624. In: *IEEE Conference Publications* (2015). <https://doi.org/10.7873/DATE.2015.0622>
39. Mell, P.M., Scarfone, K., Romanosky, S.: *A Complete Guide to the Common Vulnerability Scoring System Version 2* (2007)
40. Schroepfer, M.: An Update on Our Plans to Restrict Data Access on Facebook, April 2018. <https://about.fb.com/news/2018/04/restricting-data-access/>
41. National Institute of Standards and Technology: *Framework for Improving Critical Infrastructure Cybersecurity*, p. 41 (2014)
42. Oliver Wyman Forum: *Cybersecurity Why Is It So Hard And Getting Harder?* (2019). <https://www.oliverwymanforum.com/cyber-risk/2019/sep/why-is-cybersecurity-so-hard-and-getting-harder-what-can-be-done.html>
43. Poolsappasit, N., Dewri, R., Ray, I.: Dynamic security risk management using Bayesian attack graphs. *IEEE Trans. Dependable Secure Comput.* **9**(1), 61–74 (2012). <https://doi.org/10.1109/TDSC.2011.34>
44. Ruddle, A., et al.: *EVITA D2.3 v1.1* (2009)

45. Schmittner, C., Gruber, T., Puschner, P., Schoitsch, E.: Security application of failure mode and effect analysis (FMEA). In: Bondavalli, A., Di Giandomenico, F. (eds.) SAFECOMP 2014. LNCS, vol. 8666, pp. 310–325. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-10506-2_21
46. Schneier, B.: Attack trees - modeling security threats. *Dr. Dobb's J.* **24**(12), 21–29 (1999)
47. Yost, J.R.: The origin and early history of the computer security software products industry. *IEEE Ann. Hist. Comput.* **37**(2), 46–58 (2015). <https://doi.org/10.1109/MAHC.2015.21>
48. Tidwell, T., Larson, R., Fitch, K., Hale, J.: Modeling Internet attacks. In: Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, p. 7 (2001)
49. U.S. Department of Health and Human Services Office for Civil Rights: Breach Portal: Notice to the Secretary of HHS Breach of Unsecured Protected Health Information. https://ocrportal.hhs.gov/ocr/breach/breach_report.jsf
50. Vose, D.: Risk Analysis: A Quantitative Guide, 3rd edn. Wiley, Chichester, Hoboken (2008). oCLC: ocn174112755
51. Walker, A., O'Connor, R.V., Messnarz, R. (eds.): EuroSPI 2019. CCIS, vol. 1060. Springer, Cham (2019). <https://doi.org/10.1007/978-3-030-28005-5>
52. Wouters, L., Marin, E., Ashur, T., Gierlichs, B., Preneel, B.: Fast, furious and insecure: passive keyless entry and start systems in modern supercars. *IACR Trans. Crypt. Hardw. Embed. Syst.* **2019**(3), 66–85 (2019). <https://doi.org/10.13154/tches.v2019.i3.66-85>. <https://tches.iacr.org/index.php/TCHES/article/view/8289>

A.8 [P8] Expert Judgment for Cyber-Security-Risk

- Authors: **Michael Krisper**
- Year: 2020
- URL: https://2020.eurospi.net/images/eurospi/2020/Workshop-Report-Expert-Judgment-for-Cyber-Security-Risk_v2.pdf
- Bibliography-Reference: [168]
- Published at the EuroSPI 2020 conference website.

Summary This workshop report further describes the study of expert judgments for cybersecurity assessment which was done at the EuroSPI'19 conference and involved several experts in judging the risk of a specific cyber-security attack. The emphasis of this workshop report was the process and methodical aspects. It describes the applied IDEA protocol, which consists of multiple rounds of expert judgments and reconciliation in between. Furthermore, it describes the analysed cyber-security attack in more detail by stating each attack step.

The report then goes into detail regarding the calibration scores, and the results of the study split up for both rounds of judgment. Due to the bad calibration scores, it recommends using pre-selected experts from multiple fields to get the best results due to diversification and inclusion of many opinions. One of the lessons learned from this study is that experts should first receive some information and be allowed to prepare but should still give their first judgment individually and unaffected by others. Only in the following rounds should they reconcile.

The key findings of this user study were:

- Use multiple experts for assessments and diversify as well as possible.
- Combine expert judgments based on their judgment capability, which can be assessed using calibration questions.
- Let experts first build and state their own opinion, and then let them discuss and revise it.

My Contribution This paper was completely written and conceived by me. Although not mentioned as author, Georg Macher helped by assisting during the feasibility study at the conference.

Copyright © (CC-BY 4.0) Michael Krisper, 2020. The version included in this dissertation is a permitted reprint of the published version from the EuroSPI website, which is available here: https://2020.eurospi.net/images/eurospi/2020/Workshop-Report-Expert-Judgment-for-Cyber-Security-Risk_v2.pdf

Expert Judgment for Cyber-Security Risk

EuroSPI 2019 Workshop Report by Michael Krisper, Graz University of Technology

In the workshop “Expert Judgment for Cyber-Security Risk” at the EuroSPI conference in September 2019 in Edinburgh, Scotland [1], an expert elicitation was done to assess the risk of a cyber-security scenario involving car theft by hacking the passive keyless entry system. This elicitation was done with 21 volunteers from the conference and conducted by Michael Krisper and Georg Macher from Graz University of Technology. It followed the **IDEA protocol** by Victoria Hemming [2] and used a variation of **structured expert judgment** by Roger Cooke [3], [4].

Key Points:

1. Always use **multiple experts** for assessments. However, randomly chosen experts could lead to highly unprecise results, even when using 20 of them. We recommend a **diverse group** of domain experts (customers), technical experts (engineers) and quality supervisors (consultants).
2. **Combine** the expert response **based on their performance** during calibration – don’t simply trust them. Good experts should get much higher weight than bad experts. Don’t let the bad experts worsen the results.
3. Let the experts first build their **own unbiased opinion**, then let them **discuss and revise it**.

Procedure: The IDEA Protocol and Structured Expert Judgment

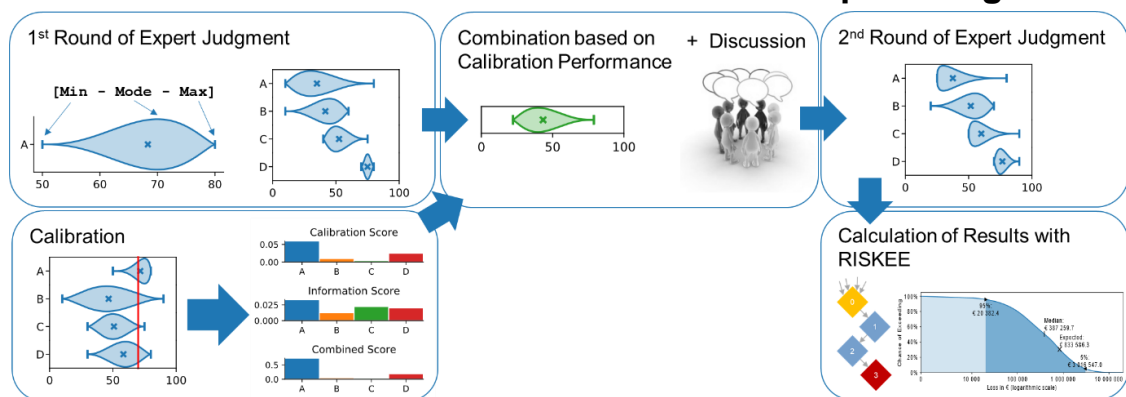


Figure 1: Our version of the IDEA protocol for EuroSPI 2019.

The IDEA protocol by Hemming et al. [2], [5] consists of two rounds of expert judgment with a discussion in between. During the discussion, the aggregated results of the first round are presented to the experts. The aggregation is done via a weighted combination of the responses based on the performance during the calibration questions. The performance is measured via the information and calibration score based on structured expert judgment by Roger Cooke [3]. In the EuroSPI workshop, we brought together 21 experts from different domains and with different background knowledge and did a study to assess the risk of a cyber-security scenario involving hacking and stealing cars. The whole workshop took approximately one hour and consisted of the following steps:

1. **Introduction** (10 Minutes) We described the general process to the workshop participants and explained some background information.
2. **Calibration** (10 Minutes) The calibration was done with a questionnaire that was filled out by the participants. The responses for the actual survey were weighted based on the calibration performance.
3. **Scenario explanation** (10 Minutes) We explained the scenario, which should be evaluated. We gave some background information and showed a possible attack which could be done.
4. **First Round of expert elicitation** (5 Minutes) Participants give their assessments for the scenario in the form of a three-point-estimate (min-mode-max). This was done individually and without prior discussion.
5. **Discussion and presentation of results** (15 Minutes) After the first round, a moderated discussion was done to clarify the scenario, let the experts exchange opinions and arguments and analyze the results of the first round.
6. **Second round of expert elicitation** (5 Minutes) Participants gave a revised second judgment.

Scenario: Fast Furious and Insecure – Hacking the passive-keyless entry system of Tesla Model S cars

The COSIC Group from KU Leuven published an attack, which allows stealing Tesla Model S cars, by exploiting flaws in the passive-keyless entry system [6]. The attack is using consumer hardware to communicate with the car and the key-fob, and a huge database of precomputed security keys. It needs vicinity to the car and the key-fob (not necessarily at the same time). The researchers demonstrated that the attack is feasible by acquiring the wireless communication hardware for a few hundred dollars and computing a rainbow table (which requires 5.4TB of disk space in total). These are acceptable efforts for being able to steal Tesla Model S cars. The attack consists of 4 steps, each taking only a few seconds to accomplish:

- *Phase 0:* Adversary acquires the car identifier wirelessly.
- *Phase 1:* Adversary does two constructed challenge-response requests with the key fob.
- *Phase 2:* The adversary recovers the key with the help of a precomputed rainbow table.
- *Phase 3:* With the recovered key, the adversary unlocks the car, starts it, and drives away.

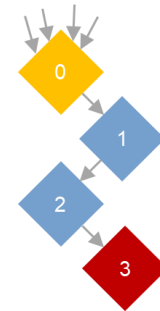
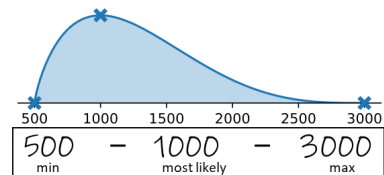


Figure 2: The Attack path.

The experts got a predefined sheet with each phase as a step on the attack path (see Figure 2). They had to fill in their estimated values in the form of [minimum-most-likely-maximum] (see Figure 3) for the attack frequency, the vulnerability, and the impact.

With this knowledge the experts should judge the risk of cyber-security for the following scenario: Imagine you are the CISO (chief information security officer) of a car rental company, having a fleet of 100 Tesla Model S cars in the beautiful city of Leuven (the town of the COSIC research group). **What are the risks concerning the previously mentioned attack?**



Example Question:
How many people died at the sinking of the Titanic?

Figure 3: An example three-point estimate [min-mode-max] visualized with the PERT-distribution.

Results

First, the calibration results are discussed, and then the actual survey results of the two assessment rounds.

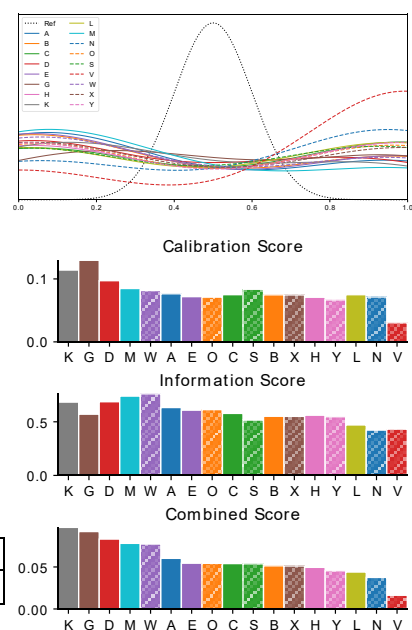
Calibration Results

The calibration results showed a disillusioning image of the experts' performance. The figure on the right shows the calibration distributions for the experts compared to a reference model of the expected outcome. Not a single expert came near the expected results. While this was a staggering finding of the survey, it was no surprise since Colson and Cooke [7] found out that this happens quite often. They looked at 33 independent studies in many domains that applied structured expert judgment, and they found out that in approximately one-third of all studies, less than two good experts were present. In 20% of the studies, not even a single good one took part, which is highly alarming and highlights the need for calibration.

The final weights of the experts had approximately the same magnitude, which means that they performed about equally during the calibration. A good expert would have got more weight than the others by multiple magnitudes.

K	G	D	M	W	A	E	O	C	S	B	X	H	Y	L	N	V
10%	9%	8%	8%	8%	6%	5%	5%	5%	5%	5%	5%	5%	5%	4%	4%	2%

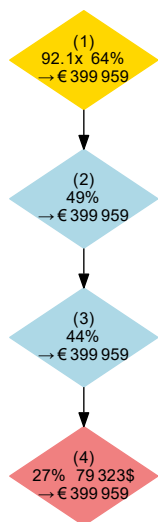
Table 1: The final weights for the experts, based on their calibration and information score.



Survey Results

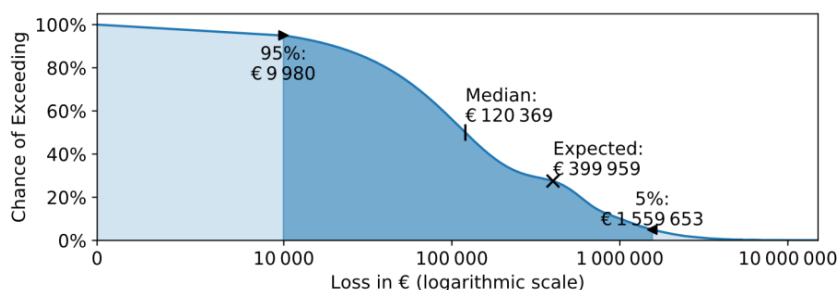
The results were evaluated using RISKEE [8], a tool for propagating and calculating uncertain risk values throughout a graph, which was also presented the first time at the EuroSPI 2019 conference [1].

Round 1



In the first round, the experts did not have the chance to discuss the scenario with each other. They judged the values independently and without external input, and the results are as follows. The average attack frequency was estimated to be about 92 times per year, and the average impact was about € 400 000. With a 95% probability, the loss will exceed about € 10 000, and with 5%, it exceeds € 1 560 000. The value with a 50% probability of exceeding is approximately € 120 000, which would correspond to one stolen car in our fleet of 100 cars.

In summary, we can expect the risk to be between € 10 000 and € 1 500 000, and we should prepare for one stolen car per year.

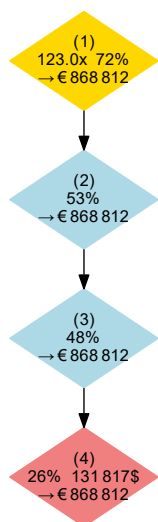


Discussion

During the discussion, many interesting questions came up. How difficult is the attack? How many people know the details of the attack? How likely is it that attackers can derive the details of the attack from the published materials by the COSIC researchers? What knowledge do the attackers need to calculate the rainbow table? Where are the cars parked? How often are the cars rented?

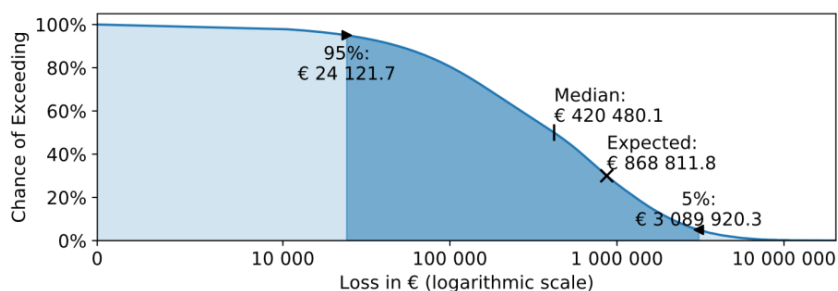
In hindsight, these were very informative because we could pin down many uncertainties and influence factors which we should investigate if we were doing a real risk assessment.

Round 2



In round two, the experts in general increased all their values, beginning with the attack frequencies as well as the vulnerabilities and the impact. It seems that discussion made them more suspicious and careful, which manifested in the nearly doubled risk values. The average attack frequency was now 123 times, while the average impact was around € 869 000. Also, the confidence interval doubled by being between € 24 000 and a whopping € 3 000 000.

As management, we would have to calculate with a median loss of € 420 000, which corresponds to about four stolen cars per year. This is quite an increase compared to the first round. Also, the 5% exceedance value of € 3 Million is quite worrying, because it would correspond to about 1/3 of the whole car fleet stolen per year. Fortunately, it has a very low probability.



Discussion and Conclusion

The study shows that it is beneficial to assess the quality of experts via calibration and combine their judgments based on that. Quality can be evaluated based on information and calibration score. Information tells us how precise or uncertain a given judgment is, compared to all other judgments. Calibration tells us how well the prediction captured the true value.

In this study, the overall calibration scores were rather low, which indicates that we, unfortunately, cannot put much trust in the results. Without calibration, we would have never known this. One reason for the low calibration scores could be that we just used random volunteers to participate in the study, and not preselected experts. We think that using preselected experts with distinct backgrounds from the following three fields would have led to better results: Firstly, experts with domain knowledge (the customers who know the problems in practice), experts with a technical background (technicians and engineers who have detailed technical knowledge), and generally knowledgeable supervisor (the consultants, who have experience, can ensure the quality, and translate between the others to avoid misunderstandings). Such diverse groups would have added different viewpoints and relevant information to the responses. To support this even further, future research about the ideal composition of expert councils must be done to get more data.

Another interesting finding of the study was that between the first and the second round of assessment, the overall risk more than doubled while the individual responses by the experts just increased slightly. It seems that the discussion ignited concerns in the participants, which in turn led to increasing the adjustment of their values. These adjustments multiplied up and resulted in doubling the total risk values.

Bibliography

- [1] A. Walker, R. V. O'Connor, and R. Messnarz, *Systems, Software and Services Process Improvement: 26th European Conference, EuroSPI 2019, Edinburgh, UK, September 18-20, 2019, Proceedings*. Springer International Publishing, 2019.
- [2] V. Hemming, T. V. Walshe, A. M. Hanea, F. Fidler, and M. A. Burgman, 'Eliciting improved quantitative judgements using the IDEA protocol: A case study in natural resource management', *PLoS ONE*, vol. 13, no. 6, p. e0198468, Jun. 2018, doi: 10.1371/journal.pone.0198468.
- [3] R. M. Cooke, *Experts in uncertainty: opinion and subjective probability in science*. New York: Oxford University Press, 1991.
- [4] A. R. Colson and R. M. Cooke, 'Expert Elicitation: Using the Classical Model to Validate Experts' Judgments', *Review of Environmental Economics and Policy*, vol. 12, no. 1, pp. 113–132, Feb. 2018, doi: 10.1093/reep/rex022.
- [5] V. Hemming, M. A. Burgman, A. M. Hanea, M. F. McBride, and B. C. Wintle, 'A practical guide to structured expert elicitation using the IDEA protocol', *Methods Ecol Evol*, vol. 9, no. 1, pp. 169–180, Jan. 2018, doi: 10.1111/2041-210X.12857.
- [6] L. Wouters, E. Marin, T. Ashur, B. Gierlichs, and B. Preneel, 'Fast, Furious and Insecure: Passive Keyless Entry and Start Systems in Modern Supercars', *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 3, pp. 66–85, May 2019, doi: 10.13154/tches.v2019.i3.66-85.
- [7] A. R. Colson and R. M. Cooke, 'Cross validation for the classical model of structured expert judgment', *Reliability Engineering & System Safety*, vol. 163, pp. 109–120, Jul. 2017, doi: 10.1016/j.ress.2017.02.003.
- [8] M. Krisper, J. Dobaj, G. Macher, and C. Schmittner, 'RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber Security', in *Systems, Software and Services Process Improvement*, Cham, 2019, vol. 1060, pp. 45–56, doi: 10.1007/978-3-030-28005-5_4.

A.9 [P9] Problems with Risk Matrices using Ordinal Scales

- Author: **Michael Krisper**
- Year: 2021
- arXiv Identifier: 2103.05440
- Category: Quantitative Finance - Risk Management
- URL: <http://arxiv.org/abs/2103.05440>
- Bibliography-Reference: [186]
- Published on arXiv in March 2021

Summary This paper discusses and summarizes several problems with risk assessment using ordinal scales and risk matrices. Such types of assessments are used in many established risk management methods, for example, FMEA, HARA, or MIL-STD-882C. Using ordinal scales and risk matrices leads to quantification errors, range distortions, risk inversion, undefined arithmetic, and many more problems which could make an assessment worthless. Thus, we make a plea for using quantifiable methods instead of qualitative assessment and refer to **RISKEE**, an example of a fully quantifiable risk assessment method. In this paper, the following problems during specific phases of risk assessment are shown:

Phase 1: Identification of influence factors

- | | |
|--------------------|-------------------------|
| (A) Incompleteness | (C) Irrelevance |
| (B) Correlations | (D) Non-Linear Behavior |

Phase 2: Rating the influence factors

- | | |
|-------------------------------------|----------------------------|
| (E) Ordinal Scales | (I) Neglecting Uncertainty |
| (F) Scale-Definition & Distribution | (J) Quantification Errors |
| (G) Range Compression | (K) Human Bias |
| (H) Ambiguity | (L) Human Inconsistency |

Phase 3: Combining the ratings

- | | |
|-------------------------------------|-----------------------------|
| (M) Undefined Arithmetic Operations | (O) Dominating Components |
| (N) Arbitrary Combination | (P) Neglecting Correlations |

Phase 4: Ranking and ordering the risks

- | | |
|--------------------------|-------------------------|
| (Q) Arbitrary Thresholds | (T) Ambiguous Order |
| (R) Inconsistency | (U) Risk Inversion |
| (S) Incoherence | (V) Unknown Uncertainty |

Phase 5: Making decisions

- | | |
|---|-----------------------|
| (W) Wrongly Perceived Impression of Benefit | (X) Deferred Feedback |
|---|-----------------------|

My Contribution This paper was written and conceived by me alone. No other authors were involved.

Copyright © (CC-BY 4.0) Michael Krisper, 2021. The version included in this dissertation is a permitted reprint of the published version from arXiv, which is available here: <http://arxiv.org/abs/2103.05440>

Problems with Risk Matrices Using Ordinal Scales

Michael Krisper

Institute of Technical Informatics

Graz University of Technology

Graz, Austria

michael.krisper@tugraz.at

Abstract—In this paper, we discuss various problems in the usage and definition of risk matrices. We give an overview of the general process of risk assessment with risk matrices and ordinal scales. Furthermore, we explain the fallacies in each phase of this process and give hints on which decisions may lead to more problems than others and how to avoid them. Among those 24 discussed problems are ordinal scales, semi-quantitative arithmetics, range compression, risk inversion, ambiguity, and neglectation of uncertainty. Finally, we make a case for avoiding risk matrices altogether and instead propose using fully quantitative risk assessment methods.

Index Terms—Risk Matrix, Risk Assessment, Risk Metric, Ordinal Scales, Range Compression, Consistency, Quantitative Methods, Qualitative Methods, Semi-Quantitative Methods, Human Bias

I. INTRODUCTION

Risk matrices are established tools to assess and rank risks in many domains and industries. They have become so common that everyone accepts and uses them without question. They have many seemingly benefits like the simplicity of usage, different coloring systems with traffic light semantics, intuitive understanding, and are seemingly proven in use over many decades. When there is little or no data available, they are praised as the weapon of choice for tackling risks and estimates in projects. Nevertheless, they have many flaws and problems that will be covered in this work.

More than a decade ago, Anthony Cox and his team started a riot against risk matrices which has not come to an end since [1]–[5]. He has shown and proven that risk matrices have severe problems that could diminish their usefulness to the point where they become even worse than random. More and more scientists, engineers, and managers have since supported the cause against risk matrices, and amongst them, Douglas Hubbard is one of the most prominent ones. In his excellent book series “How to measure anything” [6], [7], he also defends Cox et al. and demonstrates some ideas and techniques for a quantitative risk assessment method to overcome and avoid the problems of classical risk matrices.

In this work we build upon the findings of Cox [3], Hubbard [7], Artzner [8], Talbot [9], Kahnemann & Tversky [10], as well as many others over the last few decades. We show that risk matrices today still have some flaws, fallacies, and pitfalls and explain what those are. By showing them, we want to, once more, state a case for fully quantitative risk assessment using quantitative value ranges, ratio scales, and

probability distributions, which are considering the uncertainties throughout the risk analysis. Our focus in this paper lies in summarizing pitfalls and fallacies in risk assessment using ordinal scales and risk matrices with concise and understandable explanations and examples.

The paper is structured as follows: After this introduction, the motivation sections show some examples of what can go wrong using risk matrices and its consequence. Subsequently, we directly dive into the overview and descriptions of the problems, pitfalls, and fallacies of risk matrices found in the literature.

The authors are researchers at the Graz University of Technology and have a background in automotive safety, quality, and security. This work aims to show the problems of qualitative risk assessment methods to argue towards quantitative methods. In particular, in our research group, we are currently working on a method for integrated quantitative risk assessment [11], which combines safety and security. For that, we are developing a tool based on attack-trees using truly quantitative methods called RISKEE [12].

II. MOTIVATION

“*What is so bad about risk matrices?*”, one may ask, “*they are so widely accepted and established tools, they cannot be wrong.*”. Only because something is established does not mean it is without any flaws. In this section, we show some examples of pitfalls that may occur when using risk matrices. We will show some artificial examples and real-life use-cases, where the ranking of risks with risk matrices is illogical, unreasonable, or leads to problems.

A. Oil Leakage and the MIL-STD882C

This example was taken from “*What’s wrong with risk matrices*” by Cox et al. [3] and shows the problem of risk inversion (see the problem (U)). In this example, two physical hazards for environmental damage (fuel leakage in this case) are compared. The first event consists of 1 ounce of fuel spills five times per hour. The second event causes more damage but happens less frequently, with 10 pounds of fuel leaking once per week. According to the military standard, 882C [13], both would arguably get the highest frequency rating, but the one leaking 1 ounce would get a negligible hazard rating (resulting in a MEDIUM score). In contrast, the 10-pound event would get a marginal or even critical severity (resulting in a HIGH score). If we compute the risks quantitatively, we get another

HAZARD CATEGORY	CATASTROPHIC	CRITICAL	MARGINAL	NEGLIGIBLE
FREQUENCY				
FREQUENT	HIGH	HIGH	HIGH	MEDIUM
PROBABLE	HIGH	HIGH	MEDIUM	LOW
OCCASIONAL	HIGH	HIGH	MEDIUM	LOW
REMOTE	HIGH	MEDIUM	LOW	LOW
IMPROBABLE	MEDIUM	LOW	LOW	LOW

Fig. 1. An example of a risk matrix defined in the standard MIL-STD-882C [13].

result: The 1-ounce event produces 52.5 pounds of leakage per week (1oz*5*24*7), while the 10-pound event leaks 10 pounds. Thus, the first event should be rated way higher than the second event, which it is not. Figure 1 shows the risk matrix taken from MIL-STD-882C. This example shows cases where the qualitative risk score does not reflect the actual quantitative risk. Even worse, it results in an inverse order for the events' priorities, which is the consequence of risk inversion.

B. Failure Mode and Effects Analysis (FMEA)

Another example is the risk matrix in the Failure Mode and Effects Analysis (FMEA) [14]. Especially here, the severity scale is problematic because it combines four different effects scales into one ordinal scale and assigns them ranks from 1 to 10 (annoyances, failure of secondary functions, failure of primary functions, failure of safety). Furthermore, this ordinal scale is multiplied with other influence factors to get a resulting risk priority number (RPN), although multiplication is not defined on ordinal scales. Furthermore, a higher detectability factor could reduce the RPN tremendously but does not reduce the actual hazard. Is the risk of a hazard less severe just because we can detect it?

C. Hazard and Risk Analysis (HARA)

The Hazard and Risk Analysis (HARA) [15] is done at the concept and system level in the early stages of product development. The problem of this method is the ambiguity of the input scales, in particular, exposure. First, let us summarize the method itself: During the analysis, one assesses possible hazardous scenarios for their severity, exposure, and controllability. All three values are logarithmically distributed ordinal scales that assign a number for the rank. The ranks for the individual scales get added up, and the resulting number is translated into an ASIL (automotive safety integrity level) classification. Depending on the ASIL, there are exponentially more complex requirements to fulfill for developing a product in a safe way. These requirements become so high that it is challenging to implement them using only a single component. Thus, the ASIL can be decomposed into subsystems with lower ASIL but have to be independent, redundant, and diverse to avoid common cause failures. This decomposition increases the costs tremendously. A false ranking of the initial values has severe consequences to all subsequent development efforts

Class	E1	E2	E3	E4
Description	Very low probability	Low probability	Medium probability	High probability
Definition of frequency	Situations that occur less often than once a year for the great majority of drivers	Situations that occur a few times a year for the great majority of drivers	Situations that occur once a month or more often for an average driver	All situations that occur during almost every drive on average

Fig. 2. Informative examples for the exposure in the ISO 26262 [15].

C1	S1	S2	S3	C2	S1	S2	S3	C3	S1	S2	S3
E1	QM	QM	QM	E1	QM	QM	QM	E1	QM	QM	A
E2	QM	QM	QM	E2	QM	QM	A	E2	QM	A	B
E3	QM	QM	A	E3	QM	A	B	E3	A	B	C
E4	OM	A	B	E4	A	B	C	E4	B	C	D

Fig. 3. The risk matrices for the Hazard and Risk analysis in ISO 26262 [15], illustrated here by splitting it up into three parts with different controllability scores.

and costs of a product. Especially border cases are the problem here: Decreasing the score of a borderline case could decrease the resulting ASIL from D to C, which cuts the effort for product development to half. Let us examine this in the case of the exposure score. The exposure can be defined in two ways: either via the frequency of occurrence over time or as the proportion of duration in hazard situations compared to the total operating time of a product. These two aspects are reasonable because sometimes the frequency is needed (e.g., traffic situations), and sometimes the event's duration (e.g., radiation exposures). Here, ambiguity strikes the hardest: Changing the argumentation from one to the other could change the score entirely. Furthermore, even when staying in the same category, the scales are ambiguous. Figure 2 shows an excerpt for the exposure from the informative annex of the ISO 26262 [15]. Exposure rank 2 states *a few times per year for most drivers*, while E3 states *once a month or more often for an average driver*. There are two ambiguities here: Firstly, what exactly is the definition of the majority of drivers and the average driver? Does the *majority of drivers* mean more than 80%, 90%, 99%, or is it 51%? Secondly, where is the border between a few times per year and once a month or more often? Is it six times per year? Even if the boundaries would have been defined exactly, a quantification problem is still left, which we will discuss later on (problem (J)).

III. THE PROBLEMS OF RISK MATRICES

In the following sections, we go over the typical process of risk scoring methods based on risk matrices and explain why this is problematic and what the problems are. Furthermore, we compare this to quantitative risk assessment methods to show that they do not suffer from the described problems and should always be preferred over qualitative methods.

Qualitative (or semi-quantitative) risk assessment methods based on ordinal scales and risk matrices typically are done in five phases, which is illustrated by Figure 4. It shows an overview of the five phases and enlists the problems that may occur in each phase. Just to give a comparison, Figure 5 shows the corresponding quantitative approach, which also has five phases. The risk score is computed quantitatively by

estimating plausible ranges of input factors, simulating them using Monte-Carlo simulation, and comparing them to the risk appetite (also called risk affinity) using a loss exceedance curve [7], [12].

Nevertheless, in the following sections, we will discuss the five phases of qualitative risk assessment and their problems and compare them to the quantitative approach whenever reasonable:

- **Phase 1: Identifying the influence factors.** First, a set of influential factors has to be defined.
- **Phase 2: Rating of the factors.** In this phase, the input values are rated according to some scale.
- **Phase 3: Combining the ratings.** The scorings are combined to get a final risk score.
- **Phase 4: Ranking the combinations.** The risk score is ranked against other risks or filtered by some threshold.
- **Phase 5: Decision making based on the rankings.** In the end, decisions have to be made which risks to reduce. The goal is to bring the risks down to a tolerable level.

Phase 1: Identifying the influence factors

Before doing any risk assessment, one has to define the influence factors that affect the system's risk, which is assessed. Most standardized risk assessment methods defined the influence factors right away, e.g., impact, probability, severity, exposure, utility, loss. Some leave it open to be defined by the practitioners. Others are only defined within a single organization to be specialized for a specific situation, e.g., for actuary sciences, insurances, medicine, or the financial sector. For such industries, the identification of influence factors plays a massive role in risk assessment. The number of influence factors often is related to the used method. General risk assessment methods tend to use only two or three factors; specialized ones tend to use more. Furthermore, multiplicative methods also tend to use lesser factors, like two or three, and additive ones use more in general. Examples of the most frequently used factors for general risk assessment methods are the following:

- **Impact:** The impact corresponds to the actual outcome when the risk event occurs. A higher impact also means higher risk. This is also called severity, loss, magnitude, harm, effect, threat, consequence, or utility.
- **Probability:** The probability defines some notion of the likelihood that an event occurs. A higher probability also means higher risk. This is often also called exposure, likelihood, vulnerability, frequency
- **Control:** The control factor corresponds to a risk reduction possibility. Higher control over a situation results in lowering risk. Often this is also called controllability, mitigation, reduction, protection, detection, or reaction. Many methods omit this factor by reasoning that its behavior can also be modeled by reducing the probability or impact.

In contrast to that, many additive methods tend to use more specific factors like demographic features, medical attributes,

lifestyle attributes. A recently highly discussed example in Austria was the introduction of a rating scheme for the unemployment office in 2018 (Arbeitsmarktservice, AMS) [16]. This was a weighted additive scheme for rating the risk of future unemployment (or otherwise put: the chance for employability). It was based on several demographic and social factors, including, e.g., gender, age, disability, career, education, and many more. The weights for combining the factors were inferred via historical data and statistics and reflected society's bias explicitly. For example, in that scoring algorithm, women have lower job chances than men. This resulted in public discussions, similar to that for amazon's automatic firing algorithm [17]. More about that subject will be discussed in phase 3: *Combining the ratings*.

Now we discuss the problems which may occur in this first phase of selecting the influence factors:

(A) Incompleteness: To do a useful risk assessment, all essential factors have to be accounted for in the analysis. However, sometimes, factors are forgotten or overlooked. This could happen unknowingly or due to ignorance or inexperience. It could be complicated to determine the significant factors that influence the risk, especially for complex behavioral or technical systems. This is not only a problem for qualitative approaches but may happen in quantitative approaches also.

(B) Correlations: The selected factors could be correlated to each other, or in other words: they could influence each other. Even more dangerous is a negative correlation: If one factor grows, others may decline. This could result in worse than random results [3]. There is also a common pitfall of correlated influence factors regarding the view of information theory: When they are strongly correlated, they do not deliver more information than one of them alone. If two scales would always show the same value, it would suffice to use just one because the information gain would be the same. So the solution to this would be to try to avoid correlated factors. This is hard to detect for qualitative approaches, but for quantitative approaches, this could be detected via statistical methods (see problem P for more information).

(C) Irrelevance: Irrelevant factors make the risk assessment more difficult because they have to be judged, evaluated, and discussed but have no real impact on the result. There are two aspects of this: The first aspect is real irrelevance - a factor does not increase or decrease the resulting risk. Then it can be skipped in the analysis. The second aspect is, if a factor is the same for all risks, it has no relevance anymore. For example, if we would use a priority factor for risks, and every risk would be rated as a high priority, this does not help the final ranking because every risk would be equal.

(D) Nonlinear Behavior: An input factor could have non-linear behavior, making it difficult to model or rate in the next phase. Logarithmic or polynomial scales could cope with this. However, the more complex a factor is, the more difficult it is to model and judge, e.g., the driving speed as a risk factor is perceived as linear. However, the actual risk increases quadratically or even exponentially [18]–[20].

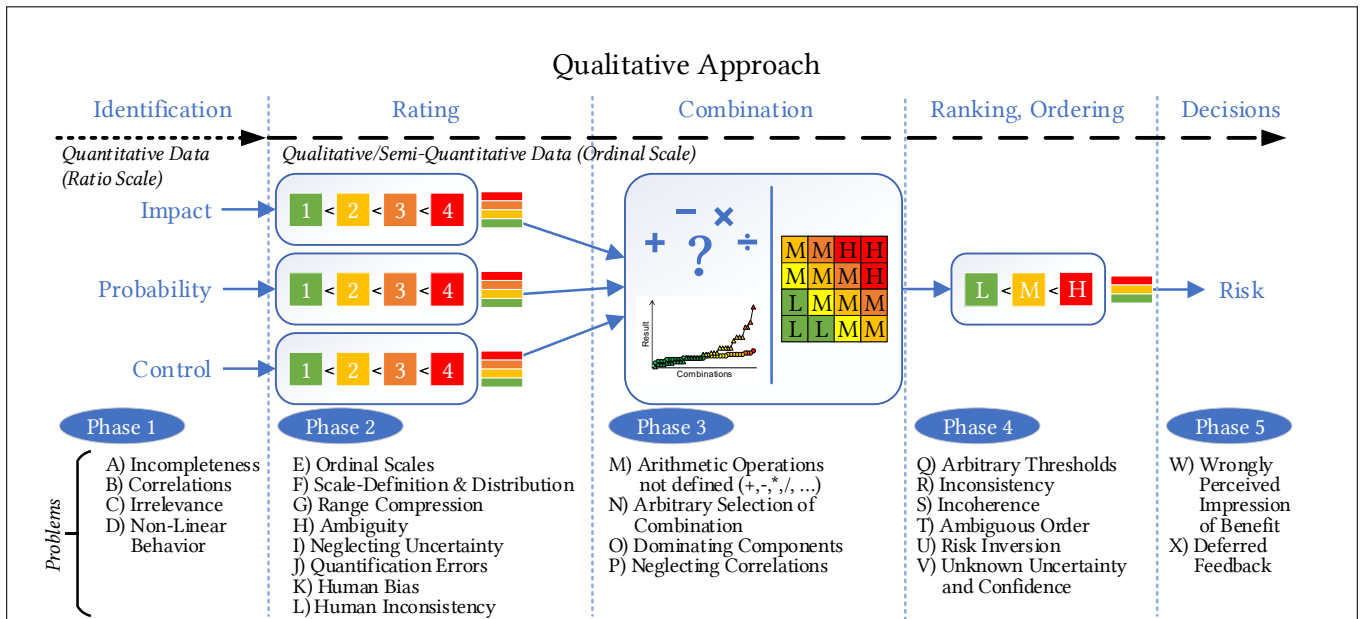


Fig. 4. The flow of information and the processing phases during a typical qualitative risk assessment approach based on ordinal scales and risk matrices.

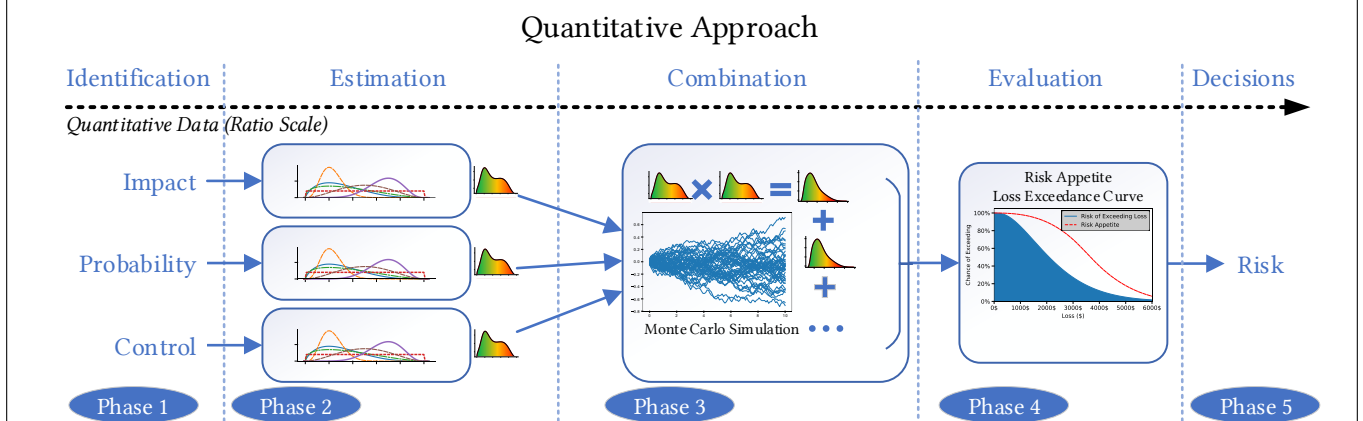


Fig. 5. The flow of information in a quantitative risk assessment approach using ratio scales, probability distributions, monte carlo simulation, and loss exceedance curves.

The mentioned problems apply to qualitative and quantitative risk assessment methods. Nevertheless, quantitative methods can at least tackle them by using mathematical tools. With statistical sensitivity analysis, correlations and irrelevances can be detected (ANOVA, Correlation Coefficients, Hypothesis Tests). Furthermore, non-linear behavior can be modeled in mathematical equations in quantitative models, which would be difficult in qualitative ordinal scales. Only incompleteness is a problem that is hard to solve for both methods. It is not trivial to detect that a factor is missing, which boils down to the often-cited management mantra by Tom DeMarco: “You cannot control what you cannot measure” [21]. Nevertheless, Hubbard et al. propose regular and immediate feedback as a tool to evaluate risk assessment methods [7]. In such a way, it could be detected that a model is not realistic and may have

left out some crucial influence factors.

Phase 2: Rating of the influence factors

After the influence factors are identified, they have to be estimated and rated. This is typically done using an ordinal scale defined by the used method or standard or has some internal company or domain-specific definition. In either way, it comes down to deciding for a class on an ordinal scale which the factor corresponds to in order to be able to rank the factors and use them for later risk comparison. In quantitative methods, this is done differently: here, an actual value, range, or even distribution on a ratio scale, which represents the reality, is chosen (no classification, just estimation of real values including the respective uncertainty). The assignment to classes in qualitative methods is one of the most discussed

problem areas in literature. We will go through the problems and show how quantitative methods can cope with most of them:

(E) Ordinal Scales: Qualitative Risk assessment methods mostly use ordinal scales. According to Stevens [22], ordinal scales only allow for ordering or ranking the items. Therefore arithmetic operations like addition or multiplication are undefined. Nevertheless, in risk assessment methods, this is done nearly all the time without question. Stevens defines the following scales, and their respective defined operations [22]:

- **Nominal scale:** Defines equality or inequality ($=$, \neq) of items. Examples: Different kinds of fruits like oranges, apples, or pears.
- **Ordinal scale:** Defines ordering relations ($<$, $>$) amongst items. Examples: School grades or the ranks in sports events.
- **Interval Scale:** Defines sums and differences ($+$, $-$) in addition to the ordering. Examples: Temperature; Time; often, values in sports events are stated as time differences compared to the first place.
- **Ratio scale:** Defines absolute ratios ($*$, $/$) between items in addition to the difference and ordering relations. Examples: Distance, Weight, Probabilities

A problem here is that by transforming quantitative values into a domain and scale, which only supports ordering relations, we lose the ability to do reasonable arithmetic, estimate uncertainty, or do any sophisticated mathematical analysis. Although the so-called “semi-quantitative” scales may give the illusion of doing calculations, the numbers are just placeholders for the class labels. They do not have mathematical foundations or actual connections to the real world. While one would refrain from multiplying “words” like *high risk* and *moderate impact* together, doing this with arbitrarily assigned numbers suddenly seems plausible. For example, if high risk=3 and moderate impact=3, then the risk is $3 \times 2 = 6$, but what is the meaning of 6?

(F) Semi-Quantitative Scale-Definition: The problems begin with the definition of a semi-quantitative distribution on the ordinal scale. There are many articles on how to design a numeric ordinal scale for use in a semi-quantitative assessment e.g. [23]–[25]. We give a short review of the different options here. What we want to achieve is a mapping from continuous quantitative data to a discrete ordinal scale. First, we have to decide how many ranks the ordinal scale should have and which ranges of values are assigned to which rank. Furthermore, if the ranks should be used for semi-quantitative arithmetics, the ranks must be assigned to numbers.

Decision 1: Number of Ranks: Does the scale have three levels (e.g., high, medium, low), 4, 5, or even 10 or 100 levels? A high number leads to a seemingly continuous scale, while a lower number is more comfortable to judge due to its coarseness [23]. An even number of levels has no neutral state, and therefore the assessment always points into a direction (either lower or higher). Uneven numbers of levels allow for a neutral position in the middle. In addition to that, Hubbard et al., as well as others, found out that people tend to avoid

extreme positions [26], [27]. Therefore, it could sometimes be reasonable to add an even more extreme level to an existing scale to outwit the bias of avoiding the most extreme. Using increasing or decreasing numbers, and even how the scale is presented can affect the outcome [27]. A further aspect of this is the next decision is if every factor should have the same number of levels for simplicity’s sake, or if they should have a different number of levels to fit the individual factor better. Scientists, like Rensis Likert, have researched the psychological effects of such scales for nearly a century now (he coined the term “Likert-scale”). However, for conciseness, we leave out further psychological debate about psychometric scales and refer to [28], [29] for further information.

Decision 2: Assignment of Quantitative Ranges to Ranks (Distribution): It is important to decide which ranges of values belong to which rank on the ordinal scale. Is this distribution scaled linear or logarithmic? Table II and Figure 6 show different kinds of distribution numerically and graphically, and here we enlist and describe some of the most common ways to define the distribution of ranks:

- **Linear:** Linear-based scales split a value range into equally distributed ranges and assign labels to them. E.g., low, medium, high. Linear-based ordinal scales relate approximately to ratio scales but still have the problem of assigning arbitrary numbers to the value ranges, which dismisses all arithmetic semantics. Sometimes they are inappropriate because, in reality, processes often behave quadratic or even exponential, and we still want to be able to cover small differences for the lower values. A linear scale would have to be very big and unhandy to cope with such behavior (imagine a scale from 0 to 100 in 0.1-interval steps. It would have 1000 different levels, while a logarithmic scale would only have 4).
- **Logarithmic:** a logarithmic scale considers processes covering large ranges while still being able to classify small ranges also, e.g., yearly, weekly, daily; small amounts of money vs. large amounts; 1, 10, 100, 1000; Injury is also very often scaled logarithmic (e.g., AIS scale).
- **Normally Distributed (Gaussian):** Scales that are arranged like a bell curve to distinguish between the tiny and huge exceptions, while average cases are all put into the same category. A variant of this is to arrange the values inversely, to distinguish the average cases better, but clumping up the extreme cases.
- **Arbitrary (Fitted):** Another possibility is a scale that is fitted arbitrarily. This can be a domain-specific definition from experts or a mathematical best fit with respect to some specific metric. With an arbitrary fit, it is possible to set the boundaries between distinct areas based on some criterion other than a mathematical distribution. The problem here is that such a fit could be highly subjective and only valid for a specific situation and point in time. One example is the energy labeling legislation in the EU [30]: While in the past the distinctions from A (best) to G (worst) were sufficient, newer technology-enabled lower

TABLE I
DIFFERENT LABELS AND SEMI-QUANTITATIVE NUMBER ASSIGNMENTS FOR ORDINAL RANKS.

Rating	Probability	Frequency	Increasing	Start at 0	Decreasing	Centered	3 Levels	4 Levels	Spaced out	Exponential
Very Low	Remotely	Never	1	0	5	-2		1	2	1
Low	Unlikely	Seldom	2	1	4	-1	1	2	4	2
Medium	As Likely as not	Sometimes	3	2	3	0	2		6	4
High	Likely	Often	4	3	2	1	3	3	8	8
Very High	Certain	Always	5	4	1	2		4	10	16

TABLE II
VARIANTS OF CLASSIFICATION SCHEMES FOR THE RANGE FROM 0 TO 100.

Level	Linear	Logarithmic	Gaussian	Inv. Gaussian
1	0...20	0...0.01	0...10	0...30
2	> 20...40	> 0.01...0.1	> 10...30	> 30...45
3	> 40...60	> 0.1...1	> 30...70	> 45...55
4	> 60...80	> 1...10	> 70...90	> 55...70
5	> 80...100	> 10...100	> 90...100	> 70...100

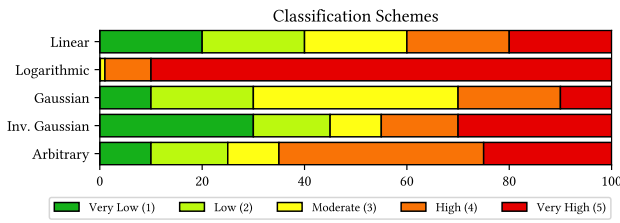


Fig. 6. Illustration of different ranges for classification into ordinal scales.

power consumption, therefore more categories have been introduced over time: A+, A++, A+++, and beginning with 2020 the scale will be completely rearranged to use the labels again A to G [30]. Energy labels based on a quantitative ratio scale would not suffer this problem (e.g., labels with power consumption in Watt and efficiency in percent, or net and gross power labels). Further examples can be found in literature like Ho et al. [24]. They show and compare the arbitrary definitions of probability scales for words of estimative probability [31], defined in several different standards.

Decision 3: Assignment of Semi-Quantitative Numbers to Ranks: Are the semi-quantitative assigned numbers centered around 0, increasing, or decreasing? Is the 0 included or not? Table II shows some examples of different scales, which are also visualized in Figure 6.

Aside from the distribution of values, the direction and location of the centers are significant [32]. Humans are susceptible and biased towards different orders, and labels in scales [27]. Table I shows different variants of number assignments to ordinal scale levels.

- **Increasing:** Numbering the scale in increasing order. This relates to the notion of “higher numbers result in higher risk”.
- **Decreasing:** Scaling the levels with decreasing ranges inverts the meaning. Here, less of something corresponds to higher risk, e.g., lower defense means a higher risk of

successful attacks.

- **Centered around 0:** Sometimes positive and negative aspects are modeled in the scale. e.g., losses or gains. In such scales, the neutral element is 0, while the more extreme cases fall to either side of the number range (positive or negative).
- **Including or omitting 0:** If the scale includes 0 and the combination includes multiplication, this 0-level could completely wipe out all other properties, regardless of how high they are. This is unwanted behavior since it conflicts with the monotonicity and relevance criteria for coherent risk metrics [8].

All these decisions are somewhat arbitrary and made mostly for convenience to have a more straightforward combination and ranking strategy later on.

(G) Range Compression: Through pressing the real values into a scheme of ordinal scales, the original uncertainty ranges get lost, and the whole value range of an ordinal class is applied to the values. Overlapping ranges get clipped, smaller ranges get widened.

(H) Ambiguity: The scales are often not defined precisely, and therefore can be argued and judged differently based on the experts’ opinion.

What is still light injury, what is already severe injury? Where is the border between once per week and once per month? How do “very low chance” and “remote chance” differ from each other? [24], [33]

(I) Neglecting Uncertainty: By classifying, the original uncertainty in the judgment gets lost. The class imposes a new default range for the uncertainty. This relates strongly to range compression and quantification errors. If the uncertainty was huge and would span multiple classes, this cannot be encoded. If the uncertainty is smaller and would span only a small fraction of a class, this cannot be encoded either and gets lost in the process.

(J) Quantification Errors: Especially on the border, quantification errors can happen. If a value changes slightly, it could step up to the next level in the ordinal scale or fall to the lower level. This could change the result tremendously (imagine going from 2 to a 1 with a multiplied combination, resulting in half of the resulting risk, but in reality, the value just changed a little bit).

(K) Human Bias: Humans are very biased [10], [34], [35]. They are scared of bad outcomes and tend to underestimate the probabilities, or they are biased towards the other way and tend to overestimate bad outcomes (risk affinity bias). Also, humans tend to judge events based on their own

experience, which is, by all means, also very flawed. Also, the cultural background and the daily condition play a huge role (see human inconsistency). This flaw can partly be covered by training for consistency and training for neutrality but is still there. Even if the probabilities are exactly defined, humans tend to misinterpret them [33]. Also, centering bias happens in this phase: Humans like to avoid the extreme values of a scale [26].

(L) Human Inconsistency: It is proven that humans are biased due to anchoring and framing. They change their judgments for the same questions based on the daily condition, the immediate situation before the judgment, or the scaling they have to do.

Phase 3: Combining the ratings

After all the influence factors were rated according to the ordinal scales, they get combined. Most methods do this either multiplicative or additive, some have a weighting scheme for addition, and some also use a deduction factor for reducing the final score. All these methods have no mathematical foundation since ordinal scales do not define arithmetic operations (only ordering relation). Still doing it introduces many problems which are discussed here.

(M) Undefined Semi-Quantitative Arithmetics: As already mentioned, ordinal scales do not support operations like addition, subtraction, multiplication, or division. They only define an ordering relation for ranking them. Any semi-quantitative calculation with such ordinal scale levels is just an arbitrary approach, without any foundations or support from mathematics. Connecting to the example already discussed in problem (E): Ordinal scales - what is the meaning of multiplying two different classes of ratings? Can the words “high risk” and “low severity” be multiplied? No. However, we tend to believe that semi-quantitative numbers can. If a high risk corresponds to 3, and a low severity corresponds to 1, the result would have been $3 \times 1 = 3$. However, here we stepped over a fallacy because if we tried this with the corresponding textual labels, it is clear that this is an invalid operation (high risk \times low severity = ?).

(N) Arbitrary Combinations: The way ratings are combined invalid and undefined, but the actual operations are also chosen arbitrarily. Should we multiply or add up all ratings? Should an optional reduction be subtracted, or should the scale be inverted to have a more comfortable mathematical formula? Should we weigh the ratings before we add them together? How are the weights defined? This strongly depends upon the scale definition (see the problem (F)). On ordinal scales, there is no correct way to do this. It is just a convention or definition by some standard. Mostly the kind of combination is chosen to result in a nice number to judge the final risk. Figure 7 shows all additive combination possibilities of the HARA risk scores in the ISO 26262 [15], compared to the same scores when multiplied. By adding them, there are equal groups of levels used for further processing - but when multiplying, they do not group up that nicely, especially the border between categories QM, A, and B are not intuitively

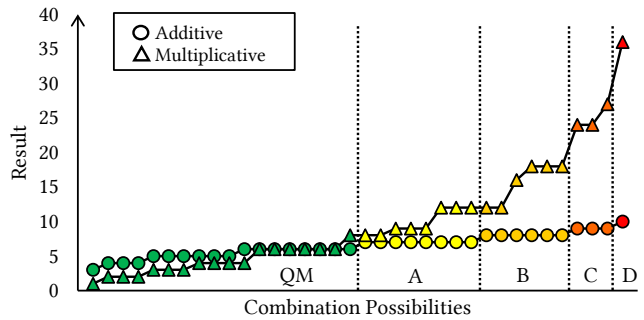


Fig. 7. Comparison of additive and multiplicative combinations of values. The example is based on the Hazard and Risk Analysis, and the areas are the respective ASIL classifications defined in the ISO 26262 [15]. While ASIL-C and ASIL-D are clearly distinguishable in both approaches, the boundaries of QM, A, and B are not that intuitive.

		Severity					Severity		
		1	2	3			1	2	3
Probability	+	1	2	3	4	*	1	2	3
	2	3	4	5	2		4	6	
	3	4	5	6	3		6	9	
	4	5	6	7	4		8	12	

Fig. 8. The tables show the results adding factors compared to multiplying them.

recognizable anymore. In addition to that, the table in Figure 8 shows numerical examples for the combination via addition and multiplication. We can see that multiplying produces more variance and also more risk classes than the addition.

Suppose the ratings include 0 as a number. In that case, multiplication could completely override all other ratings for a risk event, no matter how extreme they are (essentially making a risk irrelevant).

The definition of the weights and combination of influence factors for calculating the risks has developed into an industry because this is needed for actuaries, insurances, finance companies, and clinical and pharmaceutical industries. They try to define their weights and combinations according to some sophisticated model because, for them, it is the basis of million-dollar decisions. One example of a sophisticated combination is the algorithm for calculating the risk of unemployment in Austria’s unemployment office [16]. It is a weighted additive scheme based on several demographic and social factors. The weights for combining the factors were inferred via historical data and statistics and reflected society’s bias, which was highly disputed in the media. For example, female candidates had a higher risk of staying unemployed than male candidates, or that education was only a minor factor in getting a job.

(O) Dominating Components: If one property on the ordinal scale is low or high, it could dominate the others. For actual ratio scales, this is normal and reasonable. However, since ordinal scales lost their original real-world semantics and the numbers for the scale levels are just arbitrary definitions, the combination is not reasonable anymore. This could be a problem, especially if the scales have different distributions (one is linear, while the other is logarithmic), e.g., due to the

ordinal scales, a level from a linear distribution gets the same influence as a level from a logarithmic distribution. Increasing or decreasing the linear level would result in linear response of the actual risk while increasing/decreasing the logarithmic level would change the risk by a factor of magnitude. However, the calculated ordinal risk metric would still change only by a linear term, regardless of the actual quantitative risk change. This contradicts the positive homogeneity, and the translation invariance property [8].

(P) Neglection of Correlations: One of the most overlooked problems of risk matrices is the neglection of correlations. Cox et al. [3] stated that uncorrected negative correlations between risk matrices' influence factors could lead to worse than random results. While problem (B) already describes this, here, the actual effects are manifested. If we would know the correlations between the input factors, we could correct them in this phase by applying a correlation matrix or some other conversion factor to make up for this. This, of course, would only be possible when we had used quantitative risk assessments with ratio scales. On ordinal scales, it is difficult to model the effect one factor has on another. In the quantitative world, one can detect that a factor changes whenever another factor changes and how they are related to each other (positively or negatively correlated). Since ordinal scale levels are so coarse, this cannot be detected or corrected. The consequences of this may be severe: Correlations could add up and result in a high-risk value, or they annihilate themselves, and the actual risk would not change even when the input factors change. All in all, the calculated risk metric does not reflect the changes in the real quantitative risk, which is a severe problem and contradicts the positive homogeneity property [8].

Phase 4: Ranking and Ordering the risks according to the resulting risk metric

In this phase, the combined risk scores get ranked again and ordered for their importance. As already was the case in phase 2, this ranking is again an ordinal scale and suffers from the same problems. Here, it is even worse because the source data is not a ratio scale but a combined ordinal score which drags along all the problems described until now.

In this phase, the combined risk score is again ranked on an ordinal scale, e.g., all values above a specific threshold get a high rank, all under a specific threshold a low rank, and all in between get medium. E.g., scores from 1 to 5 get a low ranking, scores from 6 to 10 get medium, and 11 to 15 get a high ranking. In addition to all discussed problems of ordinal scales, this phase has even more problems, partly since this final ranking is the basis for decision making.

(Q) Arbitrary Thresholds: The thresholds for the ranges of the final risk levels are often chosen completely arbitrarily, with a high emphasis on simplicity. In the hazard and risk analysis, for example, all scores until 6 are grouped to the lowest risk level (QM), and above 6, every whole integer represents an own risk level (7=ASIL A, 8=ASIL B, 9=ASIL C, 10=ASIL D) [15]. This convention is convenient due to the

3x3	0-33	33-66	66-100
0-33	L	L	L
33-66	L	M	M
66-100	L	M	H

4x4	0-25	25-50	50-75	75-100
0-25	L	L	L	L
25-50	L	M	M	M
50-75	L	M	H	H
75-100	L	M	H	H

5x5	0-20	20-40	40-60	60-80	80-100
0-20	L	L	L	L	L
20-40	L	L	L	L	L
40-60	L	L	L	M	M
60-80	L	L	M	M	H
80-100	L	L	M	H	H

Fig. 9. The only possible consistent assignment of risk matrices with linear input scales and a 3-rank output score (L=low, M=medium, H=high), for 3x3, 4x4, and one of the two possible colorings for 5x5, according to Cox [3].

combination method of addition. This makes it easy to estimate the final risk score already during the individual scores in phase 2.

In comparison, quantitative methods also define an arbitrary threshold in this phase, but this would consist of a distribution called the “risk appetite”, which defines how much risk in terms of probabilities and real quantitative values is tolerable. For example, how much money loss is still tolerable with a 10% probability, 50% probability, or 90% probability that the loss is realized.

(R) Inconsistency: In their work, Cox et al. [3] describe what consistency for a risk matrix means and why this is important to achieve that property. At the same time, they prove that full consistency cannot be achieved when ordinal scales are used. Consistency means that the resulting risk score should relate directly to the real quantitative risk. For example, it should not be possible to switch from the lowest risk category to the highest by doing just a small change during the evaluation. It should not be possible that actual higher risks get scored below actual lower risks. Furthermore, all events in the same final risk class should represent the same level of actual risk, no matter how they are ranked, combined, and judged during the risk assessment. Cox defines three properties to ensure this: weak consistency, betweenness, and consistent coloring. Figure 9 shows examples for consistent risk matrices using linear scales as input and having 3 ranked ordinal scales as output (high, medium, low).

- Weak Consistency: This property defines that all events which are in the lowest-ranked risk class should have lower actual risks than all the events ranked in the highest risk class. If these two classes are disjunctive, a risk matrix has at least weak consistency.
- Betweenness: It should not be possible to jump directly

from the lowest risk class to the highest by just doing small changes in the input factors. Therefore, at least another class needed to create a border between the lowest and highest classes. This “middle” level may overlap with the lowest or highest class, but it should contain events higher than the lowest class and mostly lower than the highest risk class.

- **Consistent Coloring:** This property ensures that events on the same risk level should represent approximately the same actual risk in reality. It should not happen that two events are grouped into the same risk class but have opposite actual risks.

(S) Incoherence: Coherence is the notion of general properties for well-behaved risk metrics and was proposed by Artzner et al. [8] in 1999. They argue that a risk metric has to satisfy several properties (or axioms, as they called it) to be useful. Together these properties ensure that a risk metric is reasonable and well behaved. Since the final risk metric is obtained via a risk matrix, it should also satisfy these coherence properties:

- **Relevance:** $X > 0 \implies p(X) > 0$
When an event has an actual quantitative risk, the risk metric should also assign some positive value (the risk metric must not be zero). For ordinal scales which exclude the 0, this property holds.
- **Monotonicity:** $X \geq Y \implies p(X) \geq p(Y)$
If a real event has a higher risk than another, the risk metric should also come up with a higher or at least the same values. This is already sometimes violated due to the classification into ordinal scales. Using ordinal scales, an event with lower risk might get a higher score than an event with actual higher risk. Just recall the example of oil leakage from the motivation section.
- **Translation Invariance:** $p(X + \alpha r) = p(X) - \alpha$ This means that, by making some additional effort to reduce the risk, the respective risk metric should decrease by a corresponding amount. It should not happen that increasing or decreasing a risk produces an incoherent change of the risk metric. This also implies that if some action reduces multiple risks by the same amount, their relative order to each other must not change.
- **Subadditivity:** $p(X + Y) \leq p(X) + p(Y)$ When combining two risk events, the risk metric should be at most the addition of the single risk metric values. If the events overlap or are somehow correlated, it is less than the sum of the individual values.
- **Positive homogeneity:** $p(\lambda X) = \lambda p(X)$ This property ensures that the risk metric reflects affine changes in risk. If the risk doubles, also the metric should double.

Real quantitative methods would support these properties already with the most basic risk equation: $Risk = Impact \times Probability$. This equation fulfills all the mentioned properties of coherence and consistency when used with ratio scales or probability distributions.

(T) Ambiguous Order: If the risk matrix is at least weakly consistent, the highest and lowest-ranked risks can be ordered, but what about ordering inside the classes? If different risks result in having the same score, they cannot be prioritized anymore. Furthermore, the middle classes may partially have the same quantitative risk as the lowest or highest classes, making the ordering not very intuitive. An event with a middle-classed score may have a higher actual risk than the higher class score. Also, due to range compression, the highest risks get all clumped up together in the highest class, but the differences could be orders of magnitudes apart.

(U) Risk Inversion: The problem of risk inversion is a very severe one. Lower risks might get a higher score than actually higher risk or vice versa. We will repeat the thought experiment from the motivation section again to make this clear:

Think of an assessment of environmental contamination. Two means of transportation are compared: a car which leaks half a liter of oil every week, and a plane which happens to leak 100 liters every half a year. Furthermore, imagine the following scale for oil leakage:

- 0 ... 0.1 liter = Low impact (1)
- 0.1 ... 1 liter = Medium impact (2)
- 1 ... 10 liters = High impact (3)
- > 10 liters = Very high impact (4)

and the following time scales for the frequency:

- (4) Daily = Very high frequency (4)
- (3) Weekly = High frequency (3)
- (2) Monthly = Medium frequency (2)
- (1) Yearly = Low frequency (1)

The car would get a medium impact (2) and high frequency (3), which would result in a risk score of 6, but leaks about 100 liters per year. The plane would get very high impact (4), but only a low frequency (1), which results in a risk score of 4, and the quantitative leakage is 200 liter per year, which is double the leakage of the car. This is a typical example of risk inversion, where an actual higher risk event occurs to be scored with a lower risk score. Even if the plane’s frequency rating would have been medium (2), the score would still only be a little higher than the one of the car, while it should be double as high if it really would represent the actual risk.

(V) Unknown Uncertainty and Confidence: We already established that neglecting the uncertainty in phase 1 was not a good idea. Here we have to pay the price for this. Range compression and quantification errors have additionally contributed to completely wipe out any notion of uncertainty or confidence in our data. We cannot determine the uncertainty in our data anymore and are left only with the given ranges coming from the scale levels through the risk scores alone. Maybe our estimations have been very uncertain and may, therefore, be wrong? On the other hand, if we were very confident in our estimations, the scales’ ranges and the arbitrary calculations increased the error and uncertainty. How can we ever know if we neglected them along the way?

Quantitative methods could accomplish this by propagating the uncertainty throughout all calculations, or even better, by

using the values' distributions to consider even more details of the underlying quantitative data.

Phase 5: Making decisions based on the Risk Assessment

Based on the scoring and ranking of the risks, we want to decide which ones we want to mitigate and which ones we can tolerate. This last step strongly relates to decision theory [36]. Here, human bias plays a huge role again: the simple traffic-light system of a risk matrix is very appealing for management people. Also, just the task of talking about risks already gives an impression of achievement and benefit. Nevertheless, Hubbard et al. [7] debate that this impression may be deceitful and is just a perceived impression, not a real one. Simply discussing risks may already induce some satisfaction and the notion of accomplishment, but, as Hubbard argues, to make sure that the methods are beneficial, we have to measure their performance. Unfortunately there are little to no evidence that qualitative risk matrices work [9], and even that is just a pure argumentative one, but there are many pieces of evidence that they have quite some problems [3], [4], [6], [7], [32]. One big problem regarding the measurement of performance is that there could be years until some risk eventually occurs - hence there is no immediate feedback, which could be measured easily.

(W) Wrong Impression of Benefits: Because the risk assessment based on semi-quantitative methods seems so "easy" and "natural", there is the notion that it is correct and trustworthy. Risk matrices are established tools, which companies have used for many decades now. They may even appear to be "authoritative, and intellectually rigorous" [32] due to their seemingly correct semi-quantitative approach. However, as we established in this work, this is not the case. The benefit could be just an illusion, again bred by the human bias of uncertainty aversion and authority bias [10].

(X) Deferred Feedback: Hubbard et al. [6], [7], and others [37], stated the actual fact, that immediate feedback is an absolute must for being able to improve. The longer the feedback loop endures, the weaker the learning effect is. For risk assessment methods, the time frame between the assessment and the actual risk event may be years apart. Therefore, the initial evaluation is seldom reviewed for correctness, and methods themselves are even more rarely approved for their validity or performance. Often, the people who did the assessment already left the company long before, making it even more difficult to reevaluate and improve on the estimations. Unfortunately, this is also a problem that applies even to quantitative methods. It is important to check the validity of methods by measuring their prediction strength and comparing this with other methods to find the most suitable method for a purpose.

IV. CONCLUSION

In this work, we discussed many aspects and problems of risk matrices. We showed that in every phase of the risk assessment process, risk matrices have flaws and may introduce errors that could lead to wrong decisions in the end.

By showing this, we made another case against qualitative or semi-quantitative risk assessment methods and proposed quantitative approaches. In our research group, we are currently developing such a method based on quantitative risk assessment for cyber-security, called RISKEE [12]. The mathematics behind it can be used for any risk assessment, and we will make it available as soon as we have enough evidence supporting the correctness. In the future, we plan to investigate problems that exist even when using quantitative methods, e.g., detecting incompleteness or irrelevance of input factors and tackling the problem of deferred feedback to evaluate the appropriateness of the method. Also, combining several different expert judgments to get a realistic judgment is an area we want to tackle in future papers.

Our plea is to the safety and risk experts out there to reflect on the possible pitfalls of risk matrices and review their methods and estimations, whether they may have fallen into some of the possible traps lurking inside risk matrices. Furthermore, we encourage using quantitative risk assessment methods wherever possible.

REFERENCES

- [1] A. L. Cox, D. Babayev, and W. Huber, "Some Limitations of Qualitative Risk Rating Systems," *Risk Analysis*, vol. 25, no. 3, pp. 651–662, 2005.
- [2] A. L. Cox and D. A. Popken, "Some Limitations of Aggregate Exposure Metrics," *Risk Analysis*, vol. 27, no. 2, pp. 439–445, 2007.
- [3] A. L. Cox, "What's Wrong with Risk Matrices?" *Risk Analysis*, vol. 28, no. 2, pp. 497–512, Apr. 2008. [Online]. Available: <http://doi.wiley.com/10.1111/j.1539-6924.2008.01030.x>
- [4] L. A. Cox, *Risk analysis of complex and uncertain systems*, ser. International series in operations research & management science. New York: Springer, 2009, no. 129, oCLC: ocn276223899.
- [5] L. A. T. Cox, "Confronting Deep Uncertainties in Risk Analysis," *Risk Analysis*, vol. 32, no. 10, pp. 1607–1629, Oct. 2012. [Online]. Available: <http://doi.wiley.com/10.1111/j.1539-6924.2012.01792.x>
- [6] D. W. Hubbard, *How to measure anything: finding the value of intangibles in business*, 3rd ed. Hoboken, New Jersey: John Wiley & Sons, Inc, 2014, 00000.
- [7] D. W. Hubbard and R. Seiersen, *How to measure anything in cybersecurity risk*. Hoboken: Wiley, 2016, 00000.
- [8] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath, "Coherent Measures of Risk," *Mathematical Finance*, vol. 9, no. 3, pp. 203–228, Jul. 1999. [Online]. Available: <http://doi.wiley.com/10.1111/1467-9965.00068>
- [9] Julian Talbot, "What's right with risk matrices? | Julian Talbot on Risk, Success and Leadership," 2018. [Online]. Available: <https://www.juliantalbot.com/single-post/2018/07/31/Whats-right-with-risk-matrices>
- [10] D. Kahneman and A. Tversky, "Subjective probability: A judgement of representativeness," *Cognitive psychology*, vol. 3, no. 3, pp. 430–454, 1972. [Online]. Available: <http://datacolada.org/wp-content/uploads/2014/08/Kahneman-Tversky-1972.pdf>
- [11] J. Dobaj, C. Schmittner, M. Krisper, and G. Macher, "Towards Unified Quantitative Integrated Security and Safety Risk Assessment," *Under review*, p. 14, 2019.
- [12] M. Krisper, J. Dobaj, and G. Macher, "RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber Security," *Under Review*, p. 12, 2019.
- [13] USA Department of Defense, "Military Standard MIL-STD-882c - System Safety Program Requirements," 1979.
- [14] IEC, "IEC 60812: Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA)," 2006.
- [15] ISO, "ISO 26262," 2018.
- [16] J. Holl, G. Kernbeiß, and M. Wagner-Pinter, "Das AMS-Arbeitsmarktchancen-Modell," p. 16, 2018.
- [17] C. Lecher, "How Amazon automatically tracks and fires warehouse workers for 'productivity'," Apr. 2019. [Online]. Available: <https://www.theverge.com/2019/4/25/18516004/amazon-warehouse-fulfillment-centers-productivity-firing-terminations>

- [18] J.-L. Martin and D. Wu, "Pedestrian fatality and impact speed squared: Cloglog modeling from French national data," *Traffic Injury Prevention*, pp. –, Jan. 2017. [Online]. Available: <https://hal.archives-ouvertes.fr/hal-01557978>
- [19] C. Jurewicz, A. Sobhani, J. Woolley, J. Dutschke, and B. Corben, "Exploration of Vehicle Impact Speed – Injury Severity Relationships for Application in Safer Road Design," *Transportation Research Procedia*, vol. 14, pp. 4247–4256, 2016. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2352146516304021>
- [20] L. Aarts and I. van Schagen, "Driving speed and the risk of road crashes: A review," *Accident Analysis & Prevention*, vol. 38, no. 2, pp. 215–224, Mar. 2006. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0001457505001247>
- [21] T. DeMarco, *Controlling Software Projects: Management, Measurement, and Estimates: Management, Measurement and Estimation*. Englewood Cliffs, N.J: Pearson Education, Nov. 1982.
- [22] S. S. Stevens, "On the Theory of Scales of Measurement," *Science*, vol. 103, no. 2684, pp. 677–680, 1946.
- [23] T. R. Knapp, "Treating Ordinal Scales as Interval Scales: An Attempt To Resolve the Controversy," *Nursing research*, vol. 39, no. 2, pp. 121–123, 1990.
- [24] E. H. Ho, D. V. Budescu, M. K. Dhami, and D. R. Mandel, "Improving the communication of uncertainty in climate science and intelligence analysis," *Behavioral Science & Policy*, vol. 1, no. 2, pp. 43–55, 2015.
- [25] E. D. Smith, W. T. Siefert, and D. Drain, "Risk matrix input data biases," *Systems Engineering*, vol. 12, no. 4, pp. 344–360, Sep. 2009. [Online]. Available: <http://doi.wiley.com/10.1002/sys.20126>
- [26] D. W. Hubbard, *The failure of risk management: why it's broken and how to fix it*. Hoboken, N.J: Wiley, 2009, oCLC: ocn268790760.
- [27] G. Moors, N. D. Kieruj, and J. K. Vermunt, "The Effect of Labeling and Numbering of Response Scales on the Likelihood of Response Bias," *Sociological Methodology*, vol. 44, no. 1, pp. 369–399, Aug. 2014. [Online]. Available: <http://journals.sagepub.com/doi/10.1177/0081175013516114>
- [28] N. K. Malhotra, *Basic Marketing Research*, 4th ed. Boston: Pearson, Jul. 2011.
- [29] J. Dawes, "Do Data Characteristics Change According to the Number of Scale Points Used? An Experiment Using 5-Point, 7-Point and 10-Point Scales," *International Journal of Market Research*, vol. 50, no. 1, pp. 61–104, Jan. 2008. [Online]. Available: <https://doi.org/10.1177/1470785308050000106>
- [30] European Commission, "REGULATION (EU) 2017/ 1369 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL - of 4 July 2017 - setting a framework for energy labelling and repealing Directive 2010/ 30/ EU," p. 23, 2017.
- [31] S. Kent, "Words of Estimative Probability," 2012. [Online]. Available: <https://www.cia.gov/library/center-for-the-study-of-intelligence/csi-publications/books-and-monographs/sherman-kent-and-the-board-of-national-estimates-collected-essays/6words.html>
- [32] P. Thomas, R. B. Bratvold, and E. Bickel, "The Risk of Using Risk Matrices," in *SPE Annual Technical Conference and Exhibition*. New Orleans, Louisiana, USA: Society of Petroleum Engineers, 2013. [Online]. Available: <http://www.onepetro.org/doi/10.2118/166269-MS>
- [33] P. D. Windschitl and E. U. Weber, "The Interpretation of "Likely" Depends on the Context, but "70%" Is 70%—Right? The Influence of Associative Processes on Perceived Certainty," *Journal of Experimental Psychology: Learning, Memory, and Cognition*, vol. 20, no. 6, p. 20, 1999.
- [34] D. D. Ariely, *Predictably Irrational: The Hidden Forces That Shape Our Decisions*. New York, NY: Harper Perennial, Apr. 2010.
- [35] T. Gilovich, D. Griffin, and D. Kahneman, *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge, U.K. ; New York: Cambridge University Press, Jul. 2002.
- [36] R. A. Howard and A. E. Abbas, *Foundations of Decision Analysis*, 01st ed. Boston: Pearson, Jan. 2015.
- [37] C. Newport, *Deep work: rules for focused success in a distracted world*. London: Piatkus, 2016, oCLC: ocn951114416.

A.10 [P10] Towards an Automated Exploration of Secure IoT/CPS Design Variants

- Authors: Lukas Gressl, **Michael Krisper**, Christian Steger, Ulrich Neffe
- Year: 2020
- DOI: 10.1007/978-3-030-54549-9_25
- Bibliography-Reference: [178]
- Presented online at SAFECOMP 2020 virtual conference in September 2020.
- Published in SAFECOMP 2020 conference proceedings (LNCS volume 12234 and LNPSE volume 12234) in July 2020.

Summary In this paper, RISKEE is compared to a classical Bayesian attack graph method for design space exploration of a smart card chip design. While both methods found feasible and reasonable designs, RISKEE found fewer solutions than the Bayesian method because it also considers the impact attribute into the risk evaluation, which the Bayesian method did not. Therefore, the found solutions of RISKEE had higher quality results, at the cost of longer execution times due to using probability distributions, compared to the single-point estimates of the Bayesian method.

Specifically, RISKEE was integrated into the Security aware Design Space Exploration (SaDSE) framework. The SaDSE framework aims to find a secure system partitioning and task mapping, optimizing for either system performance or power dissipation. Within the SaDSE framework, RISKEE is used to model security attacks and their impacts on the system's cyber-security under design. Each attack scenario described by the system designers is modelled as a risk tree according to the method introduced by RISKEE. The system designers further define certain risk thresholds that must not be exceeded by the resulting embedded system design. The system partitioning and task mapping influence the resulting risks by assigning specific countermeasures to the attacks described in the risk trees, thus, reducing their vulnerabilities. We saw a significant benefit in using RISKEE during embedded system design space exploration with this integration. For integration, a conversion from Bayesian attack graphs to risk graphs had to be implemented because the paths had to be modelled differently, especially direct conditional probabilities opened up multiple paths in RISKEE, which would have been easily modelled in Bayesian graphs. Multiple paths in RISKEE resulted in slower performance since all paths have to be considered when calculating the total risk of a system.

My Contribution This paper was, for the most part, written by the first author, Lukas Gressl. My contribution was the provision and adaption of the RISKEE tool for conducting the design space exploration. I also supported the first author Lukas Gressl with integrating RISKEE into the existing design space exploration framework, and we discussed many of the paper's content and ideas.

Copyright ©2020 Springer Nature Switzerland AG. The version included in this dissertation is a permitted reprint of the published version from the SAFECOMP 2020 conference proceedings, which can be accessed via SpringerLink:

https://link.springer.com/chapter/10.1007/978-3-030-54549-9_25

Towards an Automated Exploration of Secure IoT/CPS Design-Variants

Lukas Gressl¹ (✉), Michael Krisper¹, Christian Steger¹, and Ulrich Neffe²

¹ Graz University of Technology (TU Graz), Graz, Austria
`{gressl,michael.krisper,stegeer}@tugraz.at`

² NXP Semiconductors Austria GmbH, Gratkorn, Austria
`ulrich.neffe@nxp.com`

Abstract. The advent of the Internet of Things (IoT) and Cyber-Physical Systems (CPS) enabled a new class of connected, smart, and interactive devices. With their continuous connectivity and their access to valuable information in both the digital and physical world, they are highly attractive targets for security attackers. Integrating them into the industry and our daily used devices adds new attack surfaces. These potential threats call for special care of security vulnerabilities during the design of IoT devices and CPS. Due to their resource-constrained nature, designing secure IoT devices and CPS poses a complex task, considering the selectable hardware components and task implementation alternatives. Researchers proposed a range of automatic design tools to support system designers in their task of finding the optimal hardware selection and task implementations. Said tools offer a limited way of modeling attack scenarios for a system under design. The framework proposed in this paper aims at closing this gap, offering system designers a way to consider security attacks and security risks during the early phase of system design. It offers designers the possibility to model security constraints from the view of potential attackers, assessing the probability of successful security attacks and the resulting security risk, alike. We demonstrate the framework's feasibility and performance by revisiting an industry partner's potential system design of a future IoT device.

Keywords: Cyber security · Embedded system design · Secure IoT systems · Secure CPS · Secure embedded consumer devices

1 Introduction

The increasing utilization of the Internet of Things (IoT) in the commercial market and cyber-physical systems (CPS) in the industry, opened a new attack surface. In the last decades, numerous cybersecurity exploits have been documented [1, 11]. The ongoing integration of such systems demands the consideration of cybersecurity exploits throughout the whole system design process. Introducing security measures causes additional performance delay and power consumption, contradicting the systems' requirements for fast response

times and high energy efficiency [19]. Considering the hardware and task implementation alternatives, finding the optimal solution satisfying performance and security poses a multi-objective optimization problem. Designers rely on automatic design space exploration (DSE) tools are used. There exist both classical DSE tools focusing on performance and power consumption [8, 13], and DSE frameworks offering the consideration of security constraints in a limited way [6, 7, 10, 16, 18, 20].

The framework presented in this paper introduces a new approach to introducing security constraints in early IoT/CPS design, based on both attack graphs and risk trees. Among a set of possible hardware components and task implementation alternatives, the framework finds the optimal selection of hardware components and task placements considering the system’s power consumption, performance, security attack mitigation capability, and security risk exposure. In this paper, we make the following contributions: (i) To the best of our knowledge, the framework presented here is the first to allow the consideration of security constraints modeled as Bayesian attack graphs (BAGs) and risk trees during early IoT/CPS design. (ii) We integrate both approaches and show their advantages and disadvantages. (iii) We show the framework’s feasibility based on a secure consumer device use case and the scalability of our approach.

The paper is structured as follows: Sect. 2 discusses related projects in DSE, security attack and risk modeling; Sect. 3 describes the security modeling approach, the framework’s design and implementation; Sect. 4 shows the impact of both security modeling approaches on the secure consumer device use-case; Sect. 5 gives a conclusion and discusses future work.

2 Related Work

Network administrators commonly use attack graphs when modeling attack scenarios on networks. They model attacks as consecutive steps, represented as nodes within the graph. Modeling them as BAGs adds information about the dependency of the distinct steps and the probability of their successful execution [3, 12]. Attack tree analysis (ATA) and fault tree analysis (FTA), generally used in safety analysis, use a similar modeling approach. Both scientists and engineers commonly use ATA and FTA [2]. RISKEE describes risk propagation within a system, and assesses said risk based on a tree representation [9].

A range of DSE tools considering functional safety or security constraints, in addition to the classical optimization goals, e.g., performance, power consumption, and others, have been presented in recent years [6, 7, 14, 16–18, 20]. A range of these tools focus on the abstract representation of security constraints in the design space, such as restricting the mapping of security vulnerable tasks to processor types with security extensions [16], integration of security functions into system design [20], or securing control loops [10]. In [5], security constraints and mitigation capabilities are introduced based on distinct security levels. Other works consider distinct security problems, e.g., integration of intrusion detection tasks [6], consideration of network security [7], or optimization of communication

protocols regarding message authentication [18]. These works cannot directly integrate the attacker’s perspective on the system into the DSE. Hence, they do not reflect the effect of security mechanism integration on distinct attack scenarios.

Contrasting, the framework presented in this paper allows the direct representation of security constraints in the form of BAGs and risk trees, allowing the representation of the overall system’s attack vulnerability and monetary security risk. Depending on the used modeling approach, the designer directly sees the effect of the system partitioning and task allocation on both security risk and security attack vulnerability. The framework allows the seamless interchange between the risk tree and the BAG representation for describing the security constraints posed on the IoT device/CPS under design. Considering the security performance and power overhead of the distinct solutions allows a detailed assessment of the costs and benefits of particular system designs, including their security attack mitigation capabilities.

3 Proposed Methodology

The framework allows the designer to model the system’s functionality, available architecture components, and security attack scenarios using four perspectives, as shown in Fig. 1. The work presented in [4] describes a preliminary approach to introducing security attack vulnerability into DSE. In this paper, we present a more elaborate approach, allowing the designers to describe the dependencies of the distinct security assets using rule sets. Furthermore, this paper introduces the usage of risk trees in addition to the BAG based approach. This usage of risk trees allows the framework to perform more detailed modeling of the impacts caused by successfully performed security attacks, shown in Sect. 4. However, the usage of risk trees induces additional computation time, also described in Sect. 4. The following paragraphs shortly describe the models behind the distinct perspectives serving as an input to the framework.

System Architecture and Task Representation

A task graph describes the system’s functionality with its nodes representing the tasks and the edges modeling the task’s dependencies (logical channels). Each task performs operations (*OP*) on a set of data entities coming with a set of security requirements (*SR*). High-level hardware components represent the system architecture, including communication buses that connect these components. Each hardware component has security mechanisms (*SM*) and mitigation capabilities. Each *SM* comes with a distinct performance overhead and power consumption. For each possible implementation of a task on a hardware component, the designers estimate the implementation’s worst-case execution time (WCET).

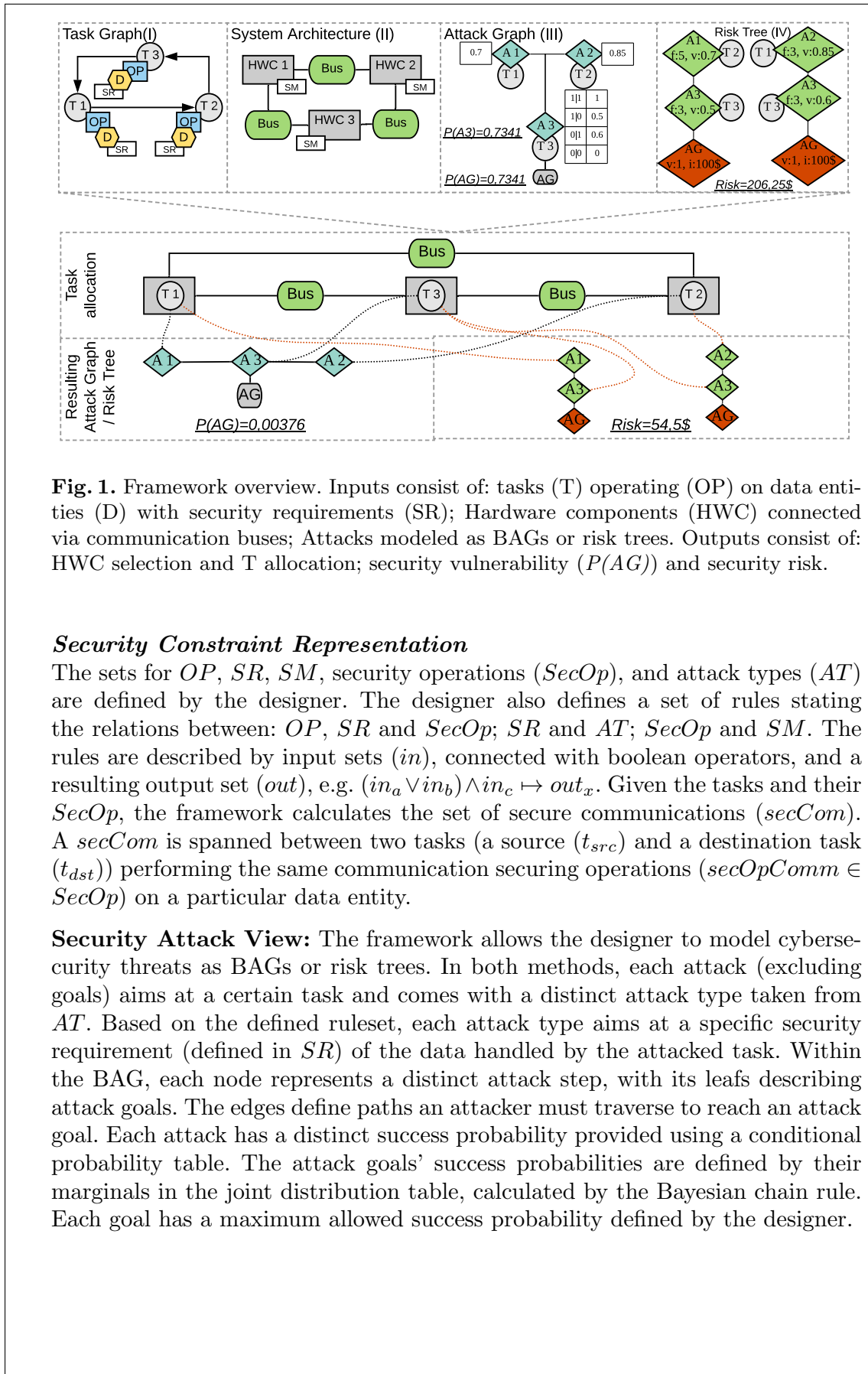


Fig. 1. Framework overview. Inputs consist of: tasks (T) operating (OP) on data entities (D) with security requirements (SR); Hardware components (HWC) connected via communication buses; Attacks modeled as BAGs or risk trees. Outputs consist of: HWC selection and T allocation; security vulnerability ($P(AG)$) and security risk.

Security Constraint Representation

The sets for OP , SR , SM , security operations ($SecOp$), and attack types (AT) are defined by the designer. The designer also defines a set of rules stating the relations between: OP , SR and $SecOp$; SR and AT ; $SecOp$ and SM . The rules are described by input sets (in), connected with boolean operators, and a resulting output set (out), e.g. $(in_a \vee in_b) \wedge in_c \mapsto out_x$. Given the tasks and their $SecOp$, the framework calculates the set of secure communications ($secCom$). A $secCom$ is spanned between two tasks (a source (t_{src}) and a destination task (t_{dst})) performing the same communication securing operations ($secOpComm \in SecOp$) on a particular data entity.

Security Attack View: The framework allows the designer to model cybersecurity threats as BAGs or risk trees. In both methods, each attack (excluding goals) aims at a certain task and comes with a distinct attack type taken from AT . Based on the defined ruleset, each attack type aims at a specific security requirement (defined in SR) of the data handled by the attacked task. Within the BAG, each node represents a distinct attack step, with its leafs describing attack goals. The edges define paths an attacker must traverse to reach an attack goal. Each attack has a distinct success probability provided using a conditional probability table. The attack goals' success probabilities are defined by their marginals in the joint distribution table, calculated by the Bayesian chain rule. Each goal has a maximum allowed success probability defined by the designer.

Risk based Attack Trees: The risk-based method uses RISKEE [9], which is a methodology for risk assessment based on attack trees with the enhancement of also modeling the consequences (impacts) of an attack, and accounting for multiple attacks over time (in the form of attack frequencies) instead of just simulating single events. The key feature of RISKEE is the usage of probability distributions for the estimation of uncertain values (which are inherent in risk assessment), providing a benefit compared to classical single-point estimates, which neglect uncertainties. The mean risk value, which is one of the results returned by RISKEE, is used as a metric for each defined attack goal. Attack goals come with a maximum allowed mean risks defined by the designer. By integrating RISKEE into the framework, we are the first to allow the consideration of risk-based security constraints during the automatic DSE for IoT/CPS.

Security Attack Mitigation: Additionally to the *SM*, each hardware component defines to what extent said mechanisms are capable of mitigating attacks. This attack mitigation ($m \in \mathbb{R} : m \in [0, 1]$) states the component's defensive capabilities. Assessing the attack mitigation is based on the judgment of the attacker's expertise and available time for breaking said defensive capabilities. Designers can deduce this mitigation capability from security assessments such as Common Criteria (CC)¹, from historical data recording known security incidents, or by expert judgments if no other information is available. The estimated mitigation factor reduces the attack probabilities (BAG) or vulnerabilities (RISKEE) $\lambda, \lambda_m \in \mathbb{R}$ of all attacks on tasks allocated on this particular hardware component, giving the mitigated probability λ_m ($\lambda_m = \lambda * (1 - m)$).

Secure Task Allocation and Partitioning: Based on the system's architecture, functionality, and the given attack scenarios, the framework finds a system partitioning and task allocations which meet the defined security constraints and optimizes either for performance or power consumption. Figure 1 depicts the BAG and RISKEE based approach and the influence of the partitioning and task allocation on the attack success probability and risk value. Hence, the task allocation must comply with a set of restrictions. (I) All tasks directly communicating with each other must be allocated on the same component or different components connected via a communication bus. (II) Each task must map to a hardware component capable of executing its *SecOp*, according to the rules defining the mapping of *SecOp* to *SM*. (III) Any task allocation and platform partitioning must fulfill the security attack constraints (in both the BAG- or RISKEE-based security attack modeling approach), meaning that for all attack goals, the defined thresholds on attack success probability or mean risk value must lie within the defined bounds.

Performance and Power Consumption Calculation: The execution times of the individual tasks depend on their component allocations, as each possible implementation of a task on a given component comes with a distinct WCET. Hence, the overall system performance depends on the selected components and the task allocations. The system power consumption consists of the component's

¹ <https://www.commoncriteriaportal.org/>.

static power dissipation and their dynamic power consumption, induced by the task implemented on them. Additionally, each component comes with a distinct security performance and power overhead for each *SM*. For each *secComm*, the framework adds the performance and power overhead of the *SM* used by the *secOpComm* of t_{src} and t_{dst} to the tasks' overall execution times and the component's power consumption, alike. For all tasks performing *SecOp* not included in any *secComm*, the framework considers the performance and power consumption overheads as well. The *secComm* must be considered separately, as a task can be both t_{src} and t_{dst} in different *secComm*. Without this consideration, the number of *SM* executions would not be integrated into the security overhead calculation correctly.

Optimization of Security Calculation: The implementation of the framework is based on the open-source *DeSyDe* framework². The framework spends its main computational effort calculating the attack probabilities (ap)/risks for each partitioning and task allocation, as for every new allocation or component selection, the BAG/RISKEE must be recalculated based on the altering attack mitigation. The framework orders the components in descending order according to their mitigation capabilities. In each calculation of the ap/risks, the framework checks if any of the said ap/risks do not fulfill the predefined limits. Upon reaching this break condition, the framework renders all further allocations on components with lesser mitigation capabilities to be insecure. Both the RISKEE and BAG based methods use the same graph structure. Hence, it is feasible to make a comparison between both methods. Opposed to BAGs, in which attack nodes can have multiple parents, the current design of RISKEE only considers single path attack scenarios. Hence, to guarantee a similar structure of the attack scenarios, the framework implements a graph-unwrapping method, turning a BAG into a set of RISKEE trees representing said BAG.

4 Experiments and Results

Using the framework, an use case based on a secure ranging system targeted for the consumer market was revisited. Table 1 describes the security rules defined by the designer to model the security aspects of the use case. The set of *OP* defines reading (r), writing (w) and storing (s) of data. The set of *SecOp* defines

Table 1. Security rules defined to model security aspects of the use case.

<i>SecOp</i> derived from <i>OP</i> and <i>SR</i>	<i>AT</i> attacking <i>SR</i>	<i>SecOp</i> using <i>SM</i>
$OP, SR \mapsto SecOp$	$AT \mapsto SR$	$SecOp \mapsto SM$
$(r \vee w) \wedge conf \mapsto so_{enc}$	$at_{inf} \mapsto conf$	$so_{enc} \vee so_{auth} \mapsto sm_{crypt}$
$(r \vee w) \wedge auth \mapsto so_{auth}$	$at_{spooft} \mapsto auth$	$(so_{enc} \vee so_{auth}) \wedge internal \mapsto sm_{te}$
$s \wedge (auth \vee conf \vee int) \mapsto so_{sst}$	$at_{tamp} \mapsto int$	$so_{sst} \mapsto sm_{tss}$

² <https://github.com/forsyde/DeSyDe>.

Table 2. Hardware components with security options. Mitigation factor (MF), performance (Perf) given in μs , and power consumption (PWC) in mW.

HWC	Security feature description	MF	Perf			PWC		
			sm_{crypt}	sm_{tss}	sm_{te}	sm_{crypt}	sm_{tss}	sm_{te}
AP	HW crypto; TEE	0.8	50	/	5	60	/	5
	SW crypto-lib sc sec., TEE	0.7	60	/	5	50	/	5
	SW crypto-lib sc sec.	0.5	40	/	/	50	/	/
	SW crypto functional	0.3	30	/	/	30	/	/
SE	HW crypto, sec store, (EAL 6+)	0.99	500	50	15	60	20	10
	HW crypto, sec store, (EAL 5+)	0.95	500	50	15	60	20	10
	HW crypto, sec store, (EAL 4+)	0.9	500	50	15	60	20	10
UR	HW crypto, TZ, HW firewall	0.8	80	/	15	50	/	10
	HW crypto, TZ	0.7	80	/	5	45	/	5
	HW crypto, 2 separate MCUs	0.85	80	/	20	50	/	10
	SW crypto-lib sc sec., TZ	0.5	160	/	5	90	/	5
	SW crypto functional	0.3	60	/	/	30	/	/

encryption (so_{enc}), authentication (so_{auth}) and secure storage (so_{sst}). The set of SR defines confidentiality ($conf$), authenticity ($auth$) and integrity (int). The set of security mechanisms (SM) defines cryptographic functionalities (sm_{crypt}), task encapsulation (sm_{te}) and tamper safe storage (sm_{tss}). The restriction of $internal$ holds if both t_{src} and t_{dst} of $secComm$ are placed on the same hardware component.

The system consists of a ranging node and a ranging anchor. The node authenticates to the anchor using a shared secret (master key) and setting up a secure session (session key). Within this session, node and anchor perform a two way ranging secured by a continually updated ranging key. The node determines its distance to the anchor in a secure way, without comprising its distance to potentially spying devices, or receiving faked ranging messages from attackers. The functionality consists of two phases, the authentication and the ranging phase, which is described by a task graph comprising 46 nodes. The authentication phase uses an external radio (e.g., Bluetooth Low Energy), the ranging phase uses ultra-wideband. Table 2 lists the security-relevant options for the hardware components available for both the anchor and the node device, giving their estimated performance (Perf) and power consumption (PWC) for their distinct SM . The devices consist of an application processor (AP), a secure element (SE), and a UWB Radio (UR). The security options comprise hardware supported cryptography (HW crypto), side-channel (sc) secured software cryptography library (SW crypto-lib sc sec.), software-based but not tested cryptography (SW crypto functional), Trusted Execution Environment (TEE) and Trust Zone (TZ), secure storage (sec. store), and hardware firewall (HW firewall). Only the SE offers secure storage.

Table 3. WCETs of security relevant tasks given in μ s.

Device	Task name	SR	AP	SE	UR
Key	Create challenge	c,a	100	150	–
Lock	Check challenge	c,a	100	170	120
Key & Lock	Derive session key	c,a,i	100	110	
Key & Lock	Derive ranging key	c,a	–	190	140
Lock	Start session	c,a	80	170	120
Key & Lock	Create secure nonce	c,a	–	120	200
Key & Lock	Create ranging message	c,a	120	–	–
Lock	Calculate distance	c,a	–	350	230

The attacks on the overall system comprise the disclosure of the key material, faking the secure authentication, which builds on a challenge request-response exchange, hijacking the ranging session, and compromising the exchanged ranging frames. Security analysts modeled these attacks using 56 nodes, both for the BAG and the RISK tree. Table 3 lists all security-relevant tasks as identified by modeling the attack scenarios, including their *SR* and WCETs on the hardware components on which system designers considered their implementations. Confidentiality (c), authenticity (a) and integrity (i) were considered as *SR*. The assessment of the attack success probabilities of the distinct attack steps for the BAG and the vulnerabilities for the RISKEE based approach were estimated using the Common Vulnerability Scoring System [15], using its *Base Metrics*.

We used the described use case as input to the framework and configured it to find the fastest, the most secure, the fastest secure, and most power-efficient and secure solution, both using the BAG and RISKEE based method. The overall system power consumption and performance was normalized. We assume that the described system performs distance-based access control. Hence, an attacker breaking the session key temporarily gains access to the secured location and might acquire the authorization to perform further criminal actions. Depending on the secured location, a successful attack might enable the disclosure of secret information, the theft of valuable items, or other critical actions. An attacker who can also disclose the keyless entry system’s master key could perform such an attack on multiple locations, depending on the key distribution policy.

Table 4. Most secure and fastest solution.

HWC	Options (most secure)	Options (fastest)
AP (node & anchor)	HW crypto; TEE	SW crypto functional
SE (node & anchor)	EAL 6+	EAL 4+
UR (node & anchor)	HW crypto; 2 separate MCUs	SW crypto functional
avg ap/avg rv	0.0005/114.4\$	0.016/4911\$
norm perf.	~2.57	1.0

Based on these considerations and a documented real-life incident³, risk experts set the impact of disclosing the system’s session key to 100.000\$, the impact of disclosing the master key to 10.000.000\$. This estimation bases on the assumption that with the session key, the attacker can only access one car temporarily. However, with the master key, the attacker might gain access to multiple cars. In this latter case, also the experts considered the reputational damage. They set the frequency for disclosing the session key to 10, and the frequency for the master key disclosure to 5 per year. We modeled these estimated impacts and frequencies in the RISKEE based approach. One must note that the attacks’ vulnerabilities and the attack success probabilities are equal for the RISKEE and BAG based approach. We set the maximum allowed risk value of 1.000\$ for all attack goals. For the BAG based method, we configured the framework to regard all solutions, in which at least one attack goal’s attack success probability exceeds the threshold of 0.002, as insecure. Table 4 describes the fastest, and the most secure system architecture found by the framework. The table shows that the framework can correctly identify optimal solutions based on distinct optimization criteria.

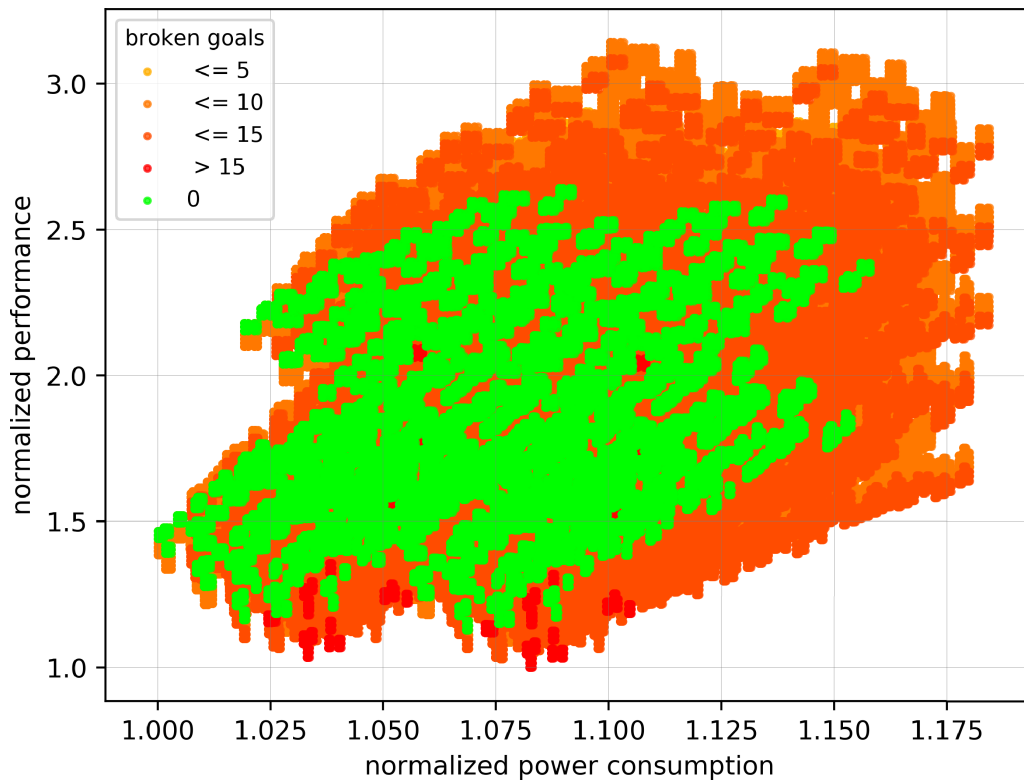


Fig. 2. Solution space identified by the framework using the BAG based method.

³ <https://www.wired.com/story/hackers-steal-tesla-model-s-seconds-key-fob/>.

Figures 2 and 3 show all solutions found by the framework based on their normalized system performance, power consumption and the number of exceeded security goals for BAG and RISKEE based security constraint calculation, respectively. Both the BAG and the RISKEE based method only consider a small number of solutions to meet their respective security constraints. Both approaches found the same solution space. Out of 5.898.240, the RISKEE based method only considered 320, the BAG 1.643 solutions to be secure. In comparison, the RISKEE method reduced the solution space of a secure solution by another 80.52%. Considering the solutions found using the BAG and the RISKEE based method, one must notice the difference in the selection of options for the distinct hardware components. This difference only comes from the frequency and the impact with which the risk experts considered the attacks on the key material in the RISKEE based approach. The BAG based method does not reflect these two attributes.

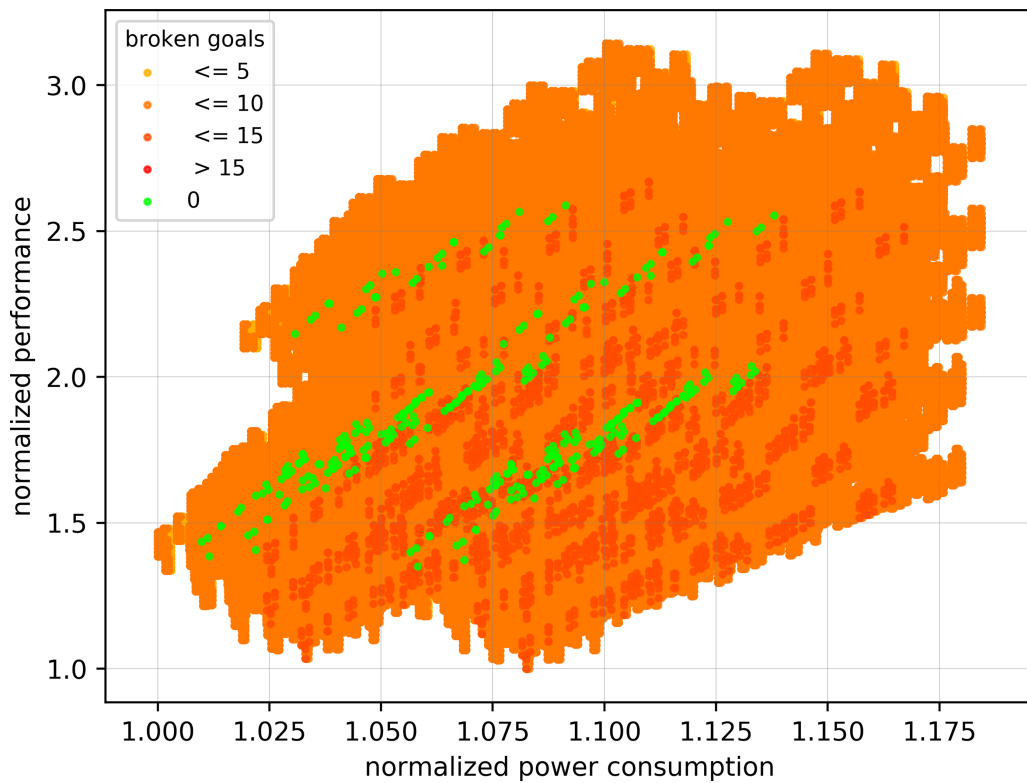
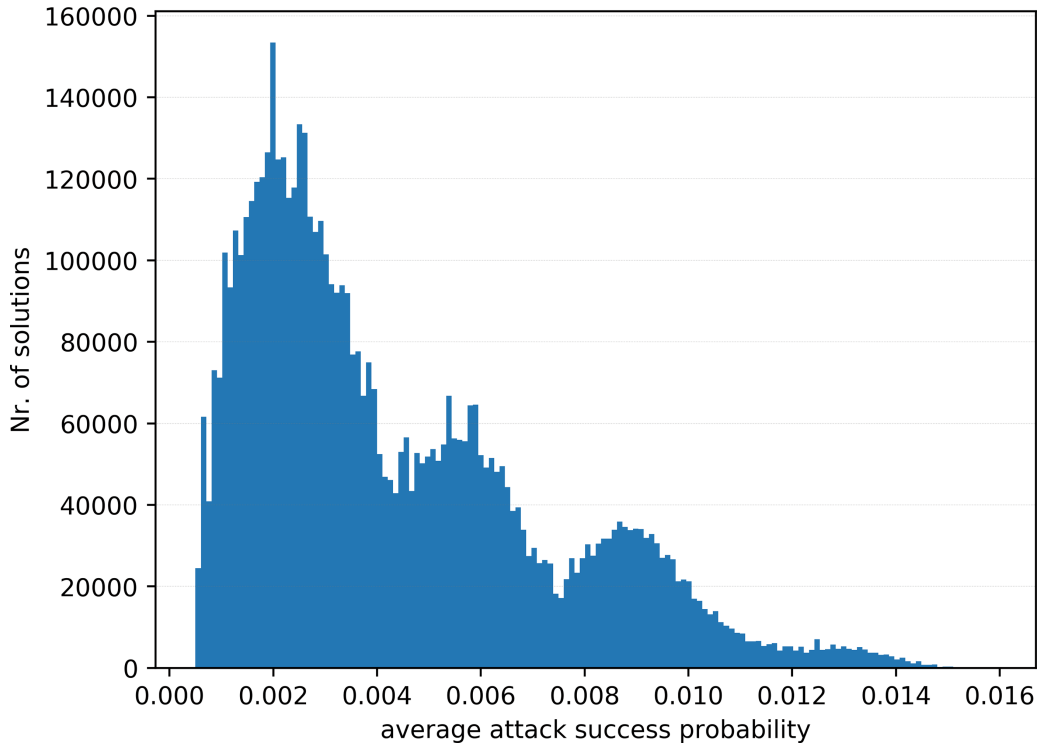


Fig. 3. Solution space identified by the framework using the RISKEE based method.

Figures 5 and 4 show the numbers of found solutions ordered by their average attack success probability and average mean risk, respectively. One can see that for the BAG based calculation, the majority of the found solutions (41.67%) has an average attack success probability of less than a fourth (~ 0.0005) of the solution with the highest attack success probability. Considering the RISKEE

Table 5. Fastest secure solutions found based on average attack probability (avg ap), average risk value (avg rv) and performance.

HWC	Fastest secure (BAG)	Fastest secure (RISKEE)
AP (node & anchor)	HW crypto; TEE	HW crypto; TEE
SE (node)	EAL 4+	EAL 6+
SE (anchor)	EAL 4+	EAL 5+
UR	HW crypto, TZ, HW firewall	HW crypto, TZ, HW firewall
avg ap/avg rv	0.00069	199.5\$
norm. perf.	~1.13	~1.35

**Fig. 4.** BAG based solutions found by the framework categorized according to their average attack success probability. Stepsize of 9.95×10^{-5} .

based calculation, the majority of solutions (64%) identified by the framework lies between 1406\$ and 2700\$ of the average mean risk value. For both calculation approaches, the framework found the least number of solutions (1.58% and 0.4% respectively for BAG and RISKEE based approach) in the most insecure fourth considering their average attack success probability/average mean risk. Table 5 describes the fastest secure solution found by the BAG and RISKEE method. Table 6 the most power-efficient secure solutions, given their average attack probability and average mean risk. One must notice that for both the secure solutions with optimal performance and power consumption, the RISKEE based solution

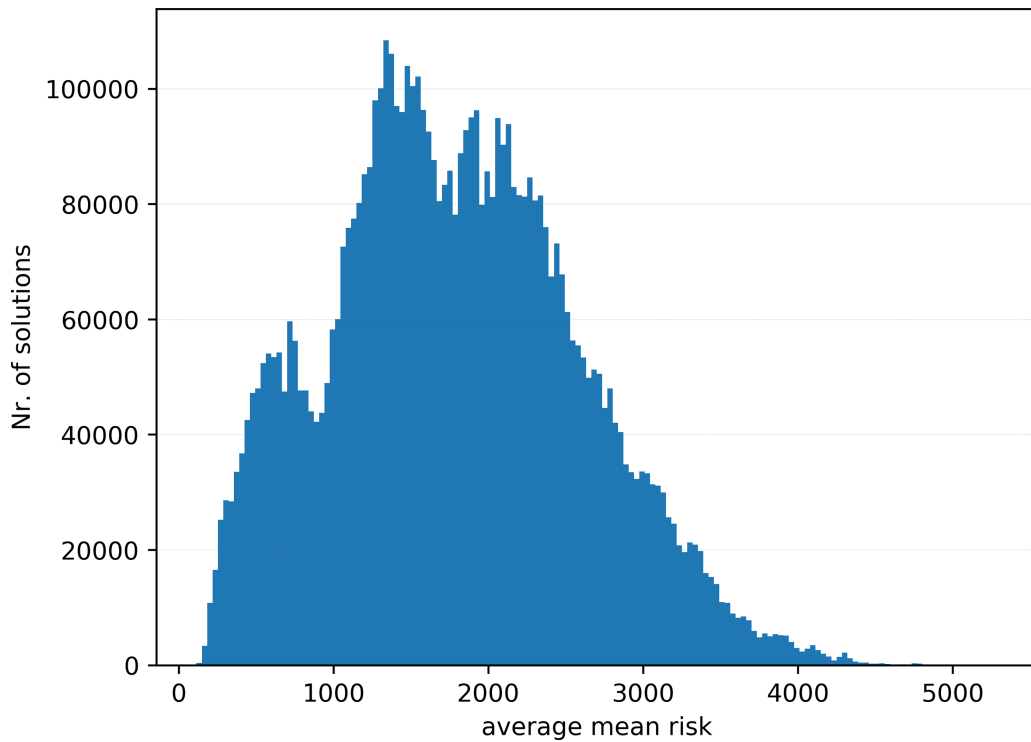


Fig. 5. RISKEE based solutions found by the framework categorized according to their average mean risk. Step size of 31.

chooses options with higher security attack mitigation capabilities than the BAG based approach, for both the SE and the AP of the anchor and node device. The increased level of security chosen for the SE is due to the high impact, with which the disclosure of the session key and the master key comes.

Said impact increases the influence of a successful key disclosure on the average mean risk of the overall system dramatically. A similar result can be seen when considering the most power-efficient and secure solutions, regarding their average attack success probability and mean risk value, respectively. Also, for this optimization criteria, the BAG based method chose less secure options for the SE, but also for the node’s AP, compared to the RISKEE based method.

Based on these results, we observed that a risk-based analysis, such as provided by RISKEE, improves the level of detail with which one can model attack scenarios. This higher granularity in the security constraints comes with additional computational overhead. The use case scenarios were executed on a system comprising 16 GB of RAM and a Intel® Core™ i7-4600U CPU with 2.10 GHz.

Table 7 shows the results of assessing the framework’s scalability and the computational overhead of calculating the security constraints using the BAG and RISKEE based methods. We executed both methods with attack graphs comprising 18, 37, and 56 attack nodes (AN), both with and without using the break criteria for the calculation of secure solutions, as described in Sect. 3. It includes the ratio between the execution times of the full security constraint

Table 6. Most power efficient and secure solutions found based on average attack probability (avg ap), average risk value (avg rv) and power consumption (power cons).

HWC	Most power eff. secure (BAG)	Most power eff. secure (RISKEE)
AP (node)	SW crypto-lib sc sec.; TEE	HW crypto; TEE
AP (anchor)	SW crypto-lib sc sec.; TEE	SW crypto-lib sc sec. TEE
SE (node)	EAL 4+	EAL 6+
SE (anchor)	EAL 4+	EAL 4+
UR (node)	HW crypto, TZ, HW firewall	HW crypto, TZ, HW firewall
UR (anchor)	HW crypto, TZ, HW firewall	HW crypto, TZ, HW firewall
avg ap/rv	0.00074	198.67\$
power cons	~1.014	~1.025

Table 7. Computational overhead for BAG and RISKEE based security constraint calculation for attack graphs with different number of attack nodes (AN).

# of AN	BAG (break)	BAG	RISKEE (break)	RISKEE
18	502s	551s/1.09	2021s	3509s/1.74
37	1943s	2052s/1.05	3315s	5597s/1.69
56	8556s	9337s/1.09	15826s	23670s/1.5

calculation and the optimized approach, both for the BAG and RISKEE based calculation. For the BAG based method, one must notice that the break criteria can speed up the calculation by $\sim 5\%$ to $\sim 9\%$. For the RISKEE based method, the calculation time is reduced by $\sim 50\%$ to $\sim 70\%$. In general, one can see that the RISKEE based method can capture more details for calculating security constraints. However, its calculation takes ~ 2.5 to ~ 6.3 times longer, when compared to the BAG based method. The higher reduction of the computational overhead for the RISKEE based method comes from the relatively higher risk calculation delay induced by this method. Hence, the more risk calculation the framework can skip, the higher the speedup of the overall calculation becomes. This speedup also shows that the attack probability calculation using the BAGs is much more efficient.

The consumer device based use case shows the difference in the BAG and RISKEE based calculation of secure system solutions. We show that the additional information regarding an attack's impact and frequency, used in the RISKEE based approach, can lead to vastly different results regarding the security constraints. This additional information leads to more time-consuming computation. Considering the maximal calculation time of the RISKEE based method (~ 6 h 30 min), a more efficient approach must be found. For future work, we will develop a combination of BAG and RISKEE based attack graphs.

5 Conclusion and Future Work

In this paper, we presented a DSE framework, which offers the designers to model cybersecurity threats as BAGs or risk trees. Thereby, the DSE framework automatically calculates a set of security constraints from these modeled security attack scenarios and finds an optimal and secure system partitioning and task allocation, with additional consideration of performance, power consumption, and other constraints. Based on a commercial consumer device use case, we showed the framework's feasibility and the distinct methods' scalabilities.

The approach's main limitation is the source from which to draw the information about the attack success probabilities and the attack frequencies for both BAG and RISKEE based calculation. At the moment, only security expert knowledge serves as input. One must also consider the same limitation for the assessment of the mitigation capabilities of hardware components. No method has yet been published on how to rate a system's ability to withstand security attacks. Hence our assumptions for the component's mitigation capabilities are based on CC certifications. In future work, we will focus on proposing such a method and on a combined calculation utilizing both the BAG and the RISKEE approach within the DSE framework.

Acknowledgment. Project partners are NXP Semiconductors Austria GmbH and the Technical University of Graz. This work was supported by the Austrian Research Promotion Agency (FFG) within the project UBSmart (project number: 859475).

References

1. Al-Mhiqani, M.N., et al.: Cyber-security incidents: a review cases in cyber-physical systems. *Int. J. Adv. Comput. Sci. Appl.* **9**(1), 499–508 (2018)
2. Ammann, P., et al.: Scalable, graph-based network vulnerability analysis. In: *Proceedings of the 9th ACM Conference on Computer and Communications Security* (2002)
3. Feng, N., et al.: A security risk analysis model for information systems: causal relationships of risk factors and vulnerability propagation analysis. *Inf. Sci.* **256**, 57–73 (2014)
4. Gressl, L., et al.: Consideration of security attacks in the design space exploration of embedded systems. In: *2019 22nd Euromicro Conference on Digital System Design (DSD)* (2019)
5. Gressl, L., et al.: A security aware design space exploration framework. In: *Proceedings of the 14th International Conference on Systems ICONS 2019*. ThinkMind (TM) Digital Library (2019)
6. Hasan, M., et al.: A design-space exploration for allocating security tasks in multi-core real-time systems. In: *Proceedings of the 2018 Design, Automation and Test in Europe Conference and Exhibition, DATE 2018* (2018)
7. Kang, E.: Design space exploration for security. In: *IEEE Cybersecurity Development Design* (2016)
8. Knerr, B.: Heuristic optimisation methods for system partitioning in HW/SW co-design. Ph.D. thesis, Vienna University of Technology (2008)

9. Krisper, M., Dobaj, J., Macher, G., Schmittner, C.: RISKEE: a risk-tree based method for assessing risk in cyber security. In: Walker, A., O'Connor, R.V., Messnarz, R. (eds.) EuroSPI 2019. CCIS, vol. 1060, pp. 45–56. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-28005-5_4
10. Li, L.W., et al.: Security-aware modeling and analysis for HW/SW partitioning. In: Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development (2017)
11. Nasser, M., et al.: Cyber-security incidents: a review cases in cyber-physical systems. *Int. J. Adv. Comput. Sci. Appl.* **9**(1), 499–508 (2018)
12. Poolsappasit, N., et al.: Dynamic security risk management using Bayesian attack graphs. *IEEE Trans. Depend. Secure Comput.* (2012)
13. Rosvall, K., et al.: Exploring power and throughput for dataflow applications on predictable NoC multiprocessors. In: Proceedings - 21st Euromicro Conference on Digital System Design, DSD 2018 (2018). <https://doi.org/10.1109/DSD.2018.00011>
14. Roudier, Y., Apvrille, L.: SysML-Sec - a model driven approach for designing safe and secure systems. In: Proceedings of the 3rd International Conference on Model-Driven Engineering and Software Development (2015)
15. Schiffman, M.: Common Vulnerability Scoring System (CVSS) (2019). <https://www.first.org/cvss/v3.1/specification-document>
16. Stierand, I., et al.: Integrating the security aspect into design space exploration of embedded systems. In: Proceedings of IEEE 25th International Symposium on Software Reliability Engineering Workshops, ISSREW 2014 (2014)
17. Tamas-Selicean, D., Pop, P.: Design optimization of mixed-criticality real-time embedded systems. *ACM Trans. Embed. Comput. Syst.* **14**(3), 1–29 (2015)
18. Xie, Y., et al.: Security/timing-aware design space exploration of CAN FD for automotive cyber-physical systems. *IEEE Trans. Industr. Inf.* **15**(2), 1094–1104 (2018)
19. Yu, R., et al.: Deploying robust security in Internet of Things. In: 2018 IEEE Conference on Communications and Network Security, CNS 2018 (2018)
20. Zheng, B., et al.: Cross-layer codesign for secure cyber-physical systems. *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **35**(5), 699–711 (2016)

A.11 [P11] Towards Security Attack and Risk Assessment during Early System Design

- Authors: Lukas Gressl, **Michael Krisper**, Christian Steger, Ulrich Neffe
- Year: 2020
- DOI: 10.1109/CyberSecurity49315.2020.9138896
- Reference: [180]
- Presented at Cyber Security 2020 (co-located Cyber Science 2020 event).
- Published in IEEEExplore Conference Proceedings for Cyber Security 2020.

Summary This paper advances the automated design space exploration method further by combining the speed of the Bayesian approach and the quality and additional information of RISKEE. In the previous publication, these two methods were compared, and it was shown that RISKEE is returning higher quality results, which are a sub-set of the Bayesian approach. However, it takes much longer to compute them due to using additional information and probability distributions.

To combine the best of both worlds, the idea in this paper is to use the Bayesian approach to prune the search space of valid solutions when searching for a system design that suffices specific security goals. In the paper, three different variants of search space pruning were analysed. This limited solution space is then applied to RISKEE to get an even more refined solution space by including additional attributes like the attack frequency and the impact an attack could impose. Since the search space was already significantly reduced before, the execution of RISKEE was faster (by 35%-40%). The combined approach with search-space pruning achieves higher performance while still maintaining high-quality outputs. This paper showed that it is possible to use different optimizations to speed up the execution times of RISKEE even in design space exploration.

My Contribution This paper was, for the most part, written by the first author, Lukas Gressl. My contribution was the provision and adaption of the RISKEE tool for conducting the design space exploration. I also supported the first author Lukas Gressl with integrating RISKEE into the existing design space exploration framework, and we discussed many of the paper's content and ideas.

Copyright ©2020 IEEE. Reprinted, with permission. From Gressl, Krisper, Neffe, Steger: Towards Security Attack and Risk Assessment during Early System Design, 2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security), June 2020. Available in IEEEExplore: <https://ieeexplore.ieee.org/document/9138896>

Towards Security Attack and Risk Assessment during Early System Design

Lukas Gressl*, Michael Krisper*, Christian Steger* and Ulrich Neffe†

*Graz University of Technology

Institute of Technical Informatics, Inffeldgasse 16, A-8010 Graz

Email: {gressl}{michael.krisper}{steger}@tugraz.at

†NXP Semiconductors Austria GmbH

Mikron-Weg 1, A-8101 Gratkorn

Email: ulrich.neffe@nxp.com

Abstract—The advent of the Internet of Things (IoT) and Cyber-Physical Systems (CPS) enabled a new class of smart and interactive devices. With their continuous connectivity and their access to valuable information in both the digital and physical world, they are attractive targets for security attackers. Hence, with their integration into both the industry and consumer devices, they added a new surface for cybersecurity attacks. These potential threats call for special care of security vulnerabilities during the design of IoT devices and CPS. The design of secure systems is a complex task, especially if they must adhere to other constraints, such as performance, power consumption, and others. A range of design space exploration tools have been proposed in academics, which aim to support system designers in their task of finding the optimal selection of hardware components and task mappings. Said tools offer a limited way of modeling attack scenarios as constraints for a system under design. The framework proposed in this paper aims at closing this gap, offering system designers a way to consider security attacks and security risks during the early design phase. It offers designers to model security constraints from the view of potential attackers, assessing the probability of successful security attacks and security risk. The framework's feasibility and performance is demonstrated by revisiting a potential system design of an industry partner.

Index Terms—Cyber Security; Embedded System Design; Secure IoT Systems; Design Space Exploration; Secure Embedded Consumer Devices

I. INTRODUCTION

The Internet of Things (IoT) is continuously revolutionizing both industry and commercial products. It offers the interaction of various devices, making them accessible to the Internet. However, the great benefits offered by the IoT make its devices susceptible to cyber-security attacks. Since its advent, reports have described numerous exploits caused by IoT devices [1], [2]. Designing secure IoT devices has, hence, become a necessity. IoT devices, embedded into larger systems, usually come with a multitude of non-functional requirements, such as timing constraints, limited power dissipation, etc. Security hardening of IoT devices generally conflicts with these requirements [3]. Finding an optimal design satisfying all requirements means solving a multi-objective optimization problem. Design space exploration (DSE) tools aid designers in this complex task during the early system design [4]–[6].

This paper describes a tool capable of finding secure system solutions considering constraints such as performance, power consumption, etc. It performs a selection of hardware components and task allocation, satisfying the non-functional requirements posed by the designer. Thereby, it allows the designer to model the security constraints as an attack tree, a risk tree, or a combined attack and risk tree. The main contributions are: (i) It describes the first DSE tool to offer the modeling of security constraints as Bayesian Attack Graphs (BAGs), risk trees (RISKEE), and a combination of both representations. (ii) It shows the costs and benefits of these approaches using the design of a secure access system.

The paper contains the following sections: Section II describes related work. Section III describes the tool's design and implementation. Section IV discusses the use case and shows the impact of the distinct security modeling approaches, considering the found solutions and the execution times. Section VI concludes the paper.

II. RELATED WORK

Classic DSE tools support system designers by performing task allocations and hardware component selections to satisfy constraints, such as system delay, power consumption, etc. [7], [8]. There exists a range of works in the context of DSE, putting their focus on cybersecurity requirements. Several works in this category focus on distinct problems. Xie et al. consider message authentication codes (MACs) transferred via a Controller Area Network. They optimize the packet sizes transferring the messages and their MACs to meet a global communication delay optimum [5]. Hasan et al. solve the problem of integrating security surveillance tasks into an existing task schedule without breaking existing timing constraints. Other works abstractly consider the integration of cybersecurity requirements. Roudier and Apvrille presented a framework that allows designers to integrate security and safety functions into early system design. The framework considers these functions during its DSE, optimizing its solutions to satisfy security and safety constraints in addition to timing- and power-consumption-constraints, etc. [9]. Zheng et al. integrate security constraints into the design of cyber-physical systems performing control-theoretic operations [10].

The frameworks discussed in this section either lack the holistic view on the system or lose too many details for considering security constraints during the early design phase. Furthermore, none of the frameworks considers the attacker’s view of the system under design. The tool presented here aims at closing this gap. It allows the designers to model security requirements as Bayesian attack graphs (BAGs), risk trees, or a combination of both representations. Latter two descriptions allow the further weighing of security risks and their mitigation costs. Furthermore, it supports designers to model security mechanisms used to mitigate the modeled attacks. These mechanisms induce additional timing- and power-consumption-overheads, etc. This additional overhead is considered by the tool when calculating the system performance, power consumption, etc., allowing the designer to perform an assessment of the costs and benefits of each solution.

A range of works has been studying BAGs, mostly in the domain of security analysis for networks. These BAGs split attack scenarios into distinct steps, represented as nodes in the graph. Each node contains the likelihood of successfully performing its represented attack step [11], [12]. The recently presented RISKEE approach models cyber-security in the form of a risk tree [13].

III. MODELING APPROACH AND CONSTRAINT CALCULATION

The framework spans a design space based on the functional-, architectural-, and attack-descriptions. System designers provide these descriptions. Based on these inputs, the framework derives the security constraints, performs different task mappings and hardware component selections, and calculates the distinct system design solutions. This section gives details about the input modeling and the solution calculations. The tool is based on [14] but allows for a more open representation of the relationship between tasks, hardware-components, and attacks. The main differences include the freely adaptable rule set defining the dependencies among the distinct input descriptions and its integration of a risk-based calculation of security constraints, introducing a combined approach using both BAGs and risk trees.

Functional description: The tool supports the description of the system’s functionality in the form of a task graph. In this graph, every node represents a distinct task, its edges the sequence of their execution. Each task in the graph might perform operations on a set of data entities. These data operations (O) comprise the transmission (tx), reception (rx), writing (w), reading (r) and storing (st) of the data entities. The data entities come with distinct sizes.

Architecture description: The architectural description comprises hardware components connected via communication buses. The system designers describe the hardware components and communication buses with classic characteristics, such as static and dynamic power consumption, etc. The communication also comprises the transmission speed.

Attack views: For the calculation of the security constraints, the framework takes as additional inputs the potential attack

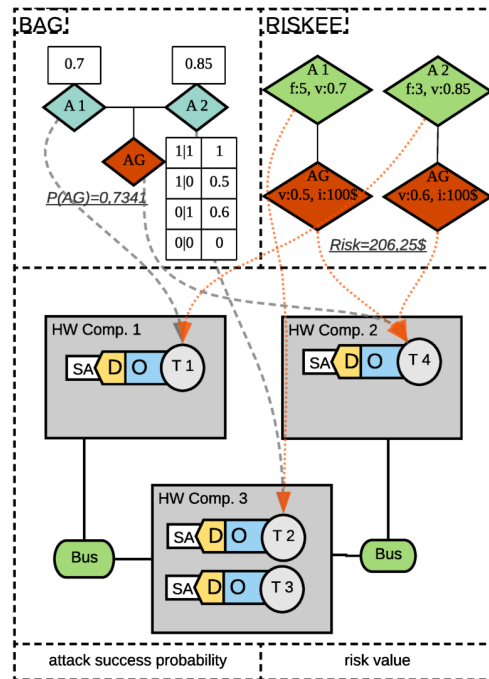


Fig. 1: Framework overview. Attacks (A) aim at tasks (T), allocated to hardware components (HW. Comp.), operating (O) on data entities (D) with security assets (SA).

scenarios described as BAGs or risk trees. Both representations come in a graph form and are, therefore, exchangeable and combinable. The BAG based method describes attack scenarios as distinct attack steps in a graph format. Each step has a distinct probability of being successfully performed, stored within the step’s conditional distribution table. The attacker’s goal is to reach the leaves of the BAG. The framework calculates the likelihood with which the attacker can reach a distinct goal using the Bayesian chain rule [15].

The second method uses RISKEE [13], which is a graph-based approach for risk assessment. RISKEE allows the modeling of the consequences (or impacts) of a successfully performed attack step. It enables the designer to add attack frequencies, thus accounting for multiple attacks over time. Furthermore, it uses probability distributions to add uncertainty when assessing an attack success probability. Hence, it provides a more detailed description compared to single-point estimates, which do not consider uncertainties. The framework uses the mean risk value (one metric returned by RISKEE) to express the risk induced by reaching an attack goal.

Each attack step in both methods aims at a distinct task of the functional description. Both the RISKEE and the BAG method are graph-based, making them combinable with each other. The only limitation is that for each attack goal, the system designers must model all nodes in the path to this goal consistently using the RISKEE or BAG approach (or both). The framework’s configuration defines with what method

it calculates the security constraints. The combined process comes with the advantage of the detailed modeling approach of the RISKEE method and faster execution time. As can be seen in Section IV, the combined approach calculates the attack success probability (asp) for each security goal using the BAG first. In a second step, it recalculates the goal using RISKEE, if the BAG based calculation did not exceed the goal's threshold, and its attack path allows it. The timing advantage of this approach is described in Section V.

Security characteristics: Additional characteristics are necessary for the functional and architectural description, as well as in the attack views, to calculate the security constraints. In the functional description, the designer adds a set of security assets (SA) to each data entity. These assets define what needs to be secured by the task, based on its interaction with the data entity (e.g., confidentiality). The assets are freely definable by the designer. The hardware components in the architectural description additionally contain a set of security mechanisms (SM) determining the capabilities with which they secure the SA of their allocated data entities. The SM further define, if they are usable for inter-task security within the same hardware component (internal (int)) or between tasks allocated on different components (external (ext)). Each component has a distinct attack mitigation factor ($amf \in \mathbb{R} : m \in [0, 1]$). This amf describes the degree to which attacks on the SA are mitigated by the component's SM . Hence, it describes the thoroughness with which the components implement their SM . In the attack view, each attack contains an attack-type (at). This attack-type defines what SA of the targeted data entity it aims at. The framework allows the unrestricted definition of SM and at by the designer.

Estimation of attack success probability and mitigation: To estimate the attack asp of an attack, the security experts use the Common Vulnerability Scoring System (CVSS) [16]. Our approach relies on the usage of the *Base Metrics* from the CVSS. These metrics cover an attack's complexity and its impacts, and, hence, can be used to describe the attacker's motivation on performing the attack. For the BAG based approach, the these metrics are combined in the asp . For the RISKEE based approach the impact is separately modeled. The estimation of the amf can be based on Common Criteria (CC) certifications, where available. The CC certifications describe the attacker's strength against whom the certified component should be able to withstand [17]. If no CC certification is available, the security experts must judge a component's attack vulnerability based on historic data or other documents.

Secure task allocation and partitioning: Based on the functional description, the architecture description, and the attack models, the framework calculates the security constraints posed on the system design. The security constraint calculation starts with determining the set of security actions ($SAct$) a task must perform to secure its data entities. These actions depend on the operations a task performs on its data entities and the data entities' SA . The framework then calculates the set of SM each component must offer to allow its mapped tasks to

perform their $SAct$. The framework defines the secure data exchanges (SDE), spanned between the distinct tasks, using the $SAct$. An SDE consist of a source and a destination task, where both source and destination task perform the same $SAct$ on a shared data entity. The framework determines for each SDE if the source and destination tasks map to the same or different hardware components. Thus, it checks if tasks' $SAct$ must be supported by internal or external SM . Each mapping between O , SA and $SAct$, $SAct$ and SM , and at to SA is definable by the designer through Boolean expressions. Table I describes the security rules applied in the example use case.

The framework finds a secure task allocation and system partitioning. First, it restricts the task to map only to components capable of performing their $SAct$. Second, it calculates the asp or risk (depending on the used method) for each component selection and task allocation. Thereby, the amf of a hardware component reduces the asp of all attacks aiming at tasks allocated on it ($asp_m = asp * (1 - amf)$). The framework performs this calculation of all solutions and marks them as either secure or insecure (if any goal exceeds its threshold). Figure 1 shows the task allocation, as well as the assignment of the attack nodes to distinct tasks.

Performance and power consumption: The performance and power consumption calculation builds on the estimation of the worst case execution time ($wcet$) of each task and the communication delay of each communication bus. This $wcet$ reflects the execution time of a task when implemented on a distinct hardware component. This task implementation is denoted with $impl(t_a)$ defining on what hardware component task t_a is implemented. Hence, $wcet(impl(t_a))$ denotes the $wcet$ of task t_a on the hardware component it is implemented on. The system performance also depends on the delays induced by the distinct SM used by each task implemented on a hardware component. This delay is denoted as $\delta(SMT(impl(t_a)))$, with $SMT(impl(t_a))$ being the used security measures of the hardware component allocating t_a . The communication overhead depends on the data entities (d) sent over the communication buses ($b_y(d)$) and the bus' transmission speed ($\lambda(b_y)$). The overall system performance ($sys_{perf} = \sum_{i=1}^n (wcet(impl(t_i)) + \delta(SMT(impl(t_i)))) + \sum_{j=1}^m (b_j(d) * \lambda(b_j))$) is the sum of the $wcet$ of all task mappings, the delays of the used SM , and the communication delays, where n denotes the number of all tasks, and m denotes all used communication buses used in the solution.

The power consumption of the system depends on the static, the dynamic power consumption, and the power consumption induced by the used SM , and the power consumed by the bus communication. The static power consumption $\rho_s = \sum_{j=1}^m pwr_s(c_j)$ is the sum of the static power consumption of all hardware components, with m being the number of all components used on the distinct solution. The dynamic power consumption $\rho_d = \sum_{i=1}^n (wcet(impl(t_i)) * pwr_d(impl(t_i)))$ is the sum of the tasks' $wcet$ multiplied with the dynamic power consumption of the hardware components they are mapped to. The power consumed by the used security mechanisms ($\phi(SMT(impl(t_a)))$) depends on the SMT used by the tasks

allocated on the hardware components. The overall security power consumption is $\rho_{sec} = \sum_{i=1}^n \phi(SMT(impl(t_i)))$. The power dissipated by the communication bus depends on the $b_y(d)$ and the bus' power consumption ($\epsilon(b_y)$). The system's communication power consumption is $\rho_{comm} = \sum_{j=1}^k (b_j(d) * \epsilon(b_j))$, with k being the number of all communication buses used by the solution. The system power consumption (sys_{pwr}) is the sum of static, dynamic, SM induced, and communication caused power consumption $sys_{pwr} = \rho_s + \rho_d + \rho_{sec} + \rho_{comm}$. This influence can be seen in Section IV.

Implementation: The framework's implementation is based on the *DeSyDe* framework¹. The *DeSyDe* framework is a DSE tool using a constraint programming approach. It is capable of finding exact solutions, adding break criteria such as maximum delay or power consumption. For the security constraint calculate, the framework performs a limited permutation approach to reduce the number of tasks to component mappings. This approach uses the attack goals' *asp*, or the risk exceedance, as break criteria. The approach optimizes the security constraint calculation by sorting the hardware components according to their *amf*. The second optimization comes with the combination of the BAG and the RISKEE method. The RISKEE method has a higher computation time for calculating the security constraints than the BAG approach. In the combined approach, the framework calculates each attack goal using the BAG method, and if the goal yields a *asp* below its threshold, it uses RISKEE to refine the result of the goal even further. This combined approach achieves highly informative results from RISKEE while still maintaining the performance of the BAG method. This combined approach's speedup is shown in Section V.

IV. EXPERIMENT

We used the framework for the early design of a keyless entry system. This system consists of a mobile node and a stationary system comprising several anchors. Both nodes and anchors use an application processor (AP), a secure element (SE), a Bluetooth Low Energy radio, and an ultra-wideband (UWB) radio (UR). The anchor system measures its distance to the node applying a double-sided two-way ranging algorithm using the UR. The devices use a set of SM offered by the hardware components. These SM secure the SA of the data entities and the communication and the overall ranging process between the node and the anchors. We derived these security characteristics conducting a STRIDE and CIA analysis [18], [19]. The rules listed in Table I describe the mappings between the SA , O , $SAct$, SM , and at .

Table II lists the hardware components that are usable for both anchor and the node device. This table lists the components' security features, their *amf*, and the performance and power consumption of their single security mechanisms (*sm*). The framework can choose between different options for the AP, the SE, and UR. They comprise the usage of hardware-based (HWC) and software-based cryptography

¹<https://github.com/forsyde/DeSyDe>

TABLE I: Security rules of the use case. The at comprise information disclosure (at_i), spoofing (at_s), and tampering (at_t). The $SAct$ comprise encryption (a_e), authentication (a_a), and secure storage (a_s). The SA comprise confidentiality (c), authenticity (a), and integrity (i). The SM comprise cryptography (sm_c), task encapsulation (sm_e), and tamper safe storage (sm_s)

$SAct$ from SA, O	AT to SA	$SAct$ to SM
$a_e = (r \vee w) \wedge c$	$at_i \mapsto c$	$sm_c = (a_e \vee a_a) \wedge ext$
$a_a = (r \vee w) \wedge a$	$at_s \mapsto a$	$sm_e = (a_e \vee a_a) \wedge int$
$a_s = st \wedge (a \vee c \vee i)$	$at_t \mapsto i$	$sm_s = a_s$

TABLE II: Hardware components with security options. Attack mitigation factor (*amf*), performance (Perf) given in μs , and power consumption (PWC) in mW

HWC	Sec. Feat.	<i>amf</i>	Perf/PWC		
			sm_c	sm_s	sm_t
AP	HWC, TEE	0.75	40/50	-/-	10/10
	SWC scp, TEE	0.6	70/60	-/-	10/10
	SWC scp	0.5	70/60	-/-	-/-
	SWC f	0.2	45/40	-/-	-/-
SE	EAL 6+	0.95	110/70	50/20	20/15
	EAL 5+	0.9	100/60	30/10	15/10
	EAL 4+	0.8	100/50	20/10	10/10
UR	HWC, FW	0.6	50/30	-/-	15/10
	HWC, TZ	0.5	50/30	-/-	10/10
	HWC, MS	0.7	60/40	-/-	20/20
	SWC, TZ	0.4	80/40	-/-	10/10
	SWC, f.	0.2	50/35	-/-	-/-

(SWC), trusted execution environment (TEE), trust zone (TZ), hardware firewall (FW), and microcontroller separation (MS). The SWC comes with side-channel protection (scp) or is purely functional (f). The HWC and SWC only characterize the security mechanisms offered by the hardware components and do not concern the implementation of the tasks mapped on the hardware component.

The functionality of the use-case described in here comprises a task graph with 57 nodes, resembling an authentication, session exchange, and ranging phase. The potential attacker who attacks the system is thought to be capable of sniffing and intercepting the communication between the devices, sniffing the communication within the single devices, tampering with the devices' memory, and intruding the software stack of the devices to a certain degree. The attacker has computational resources to brute-force weak cryptographic algorithms using small secret keys. The attacks comprise the secret key disclosure, faking secure authentication, hijacking the session, and compromising the ranging messages. The overall attack scenario consists of 64 attack nodes. Both the BAG and the risk tree have the same attack nodes. Table III lists the security relevant tasks with their SA and WCETs.

The purpose of the use case is to show the framework's feasibility and capability of finding solutions based on the given constraints and the optimization goal. Hence, we configured the framework to find the most secure solution, the solution with the best overall performance, the fastest yet secure solution, and the secure solution with the least power consumption. The attack goals' thresholds were uniformly set

TABLE III: WCETs of security relevant tasks given in μ s.

Device	Task Name	SA	AP	SE	UR
Node	create challenge	c,a	80	100	-
Anchor	check challenge	c,a	70	90	100
Node & Anchor	derive session key	c,a,i	-	140	80
Node & Anchor	derive rng key	c,a	-	120	60
Anchor	start rng session	c,a	-	150	100
Node & Anchor	create sec. nonce	c,a	-	150	80
Node & Anchor	create ranging message	c,a	-	90	50
Anchor	calculate distance	c,a	-	250	140
Node & Anchor	create chall. open	c,a	30	80	50
Node & Anchor	check chall. open	c,a	20	90	50

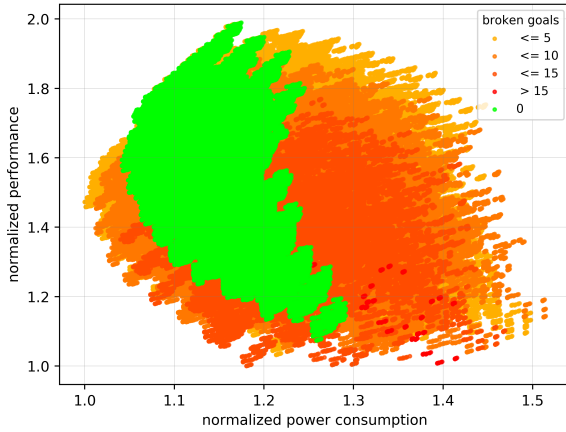


Fig. 2: Solution space identified by the framework using the BAG based method.

to 0.005 for the BAG based security constraint calculation, and to 2,000\$ for the RISKEE based one. The combined BAG and RISKEE method uses the same thresholds.

The system presented in the use case targets secure access to a vehicle, supporting a secure keyless entry system. It allows access to the car only to authorized persons who are nearby. The distance between the vehicle hosting the anchor system, and the authorized person holding the tracked node, is acquired by executing a ranging protocol based on the exchanging of UWB packets. The system generates a timely limited session key to secure the ranging process. This session key is derived from a master key stored at each device and exchanged after every ranging session. Disclosing the session key enables the attacker to gain access to the car for a short period, whereas revealing the master key means timely unlimited access to one or many cars, depending on the key distribution strategy.

The impact of disclosing the session key was set to 250,000\$ and for the master key to 30,000,000\$. We set these values to evaluate the risk induced by the value of the session and master keys. We assumed that the master key is less prone to disclosure, as, naturally, it should be more challenging to access. Hence, we set the frequency with which a potential attacker attempts to discover the master key to 10 times per year, for the session key to 50 times per year. The risk tree used for the use case resembles these impact and frequency values. The vulnerabilities of all attacks in the risk tree equal the asp

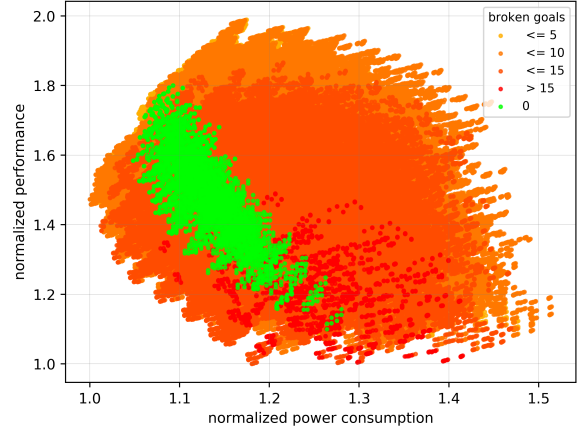


Fig. 3: Solution space identified by the framework using the RISKEE based method.

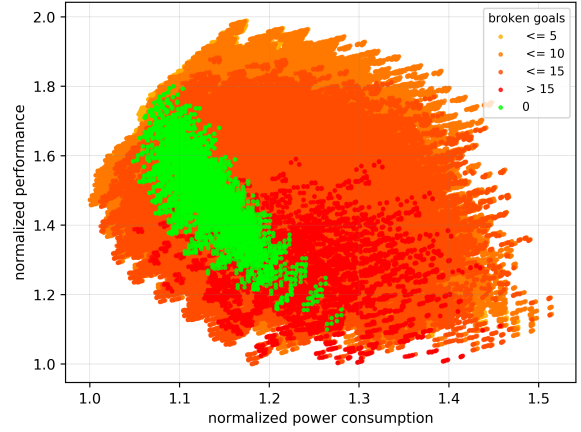


Fig. 4: Solution space identified by the framework using the combined method of RISKEE and BAG.

of the identical attack steps in the BAG. Hence, the differences between the RISKEE and BAG based methods stem from the characterization of the attack impact and frequency.

The framework calculates, additionally to the security constraints (how many attack goals are reached), the average attack probability ($AP_{avg} = \frac{\sum_{i=0}^G asp_i}{G}$) and the average mean risk value ($MRV_{avg} = \frac{\sum_{i=0}^G mrv_i}{G}$), where G is the number of all attack goals, and mrv is the goal's mean risk value. The mean risk value is one parameter returned by the RISKEE. The framework uses this parameter to assess the risk of the attacker reaching the attack goal. The framework calculates these values to characterize the security soundness of a solution. Table IV shows the solution with the lowest security vulnerability (considering the AP_{avg} and MRV_{avg}) and the system setup with the best performance. It shows that the framework can identify the optimal solutions.

Figures 2, 3, and 4 depict the solution space identified using the BAG based, the RISKEE based, and the com-

bined approach. All three approaches found overall 947.072 solutions that fulfill the basic security functionalities. Each solution represents a valid mapping of tasks to hardware components regarding the $SAct$ and SM . The BAG based computation resulted in 576.000 solutions found to meet all attack goal thresholds. The RISKEE based method found 384 solutions that do not exceed any attack goal threshold. The BAG- and RISKEE-based calculation approach largely differ in the number of secure solutions. The BAG-based approach reduces the number of solutions by 39.18%, the RISKEE-based approach by 99.96%. The additional attributes of the RISKEE method allow a narrowing of the solution space. The vulnerabilities in RISKEE and the ap in the BAG are identical. Hence, the reduction of secure solutions only stems from the frequency and the impact of the attacks. The master key's disclosure mainly influences the number of secure solutions.

The impact of a successful key disclosure also influences the fastest and least power consuming secure solutions, as described in Table V and VI. Due to the high impact of the secret key disclosure, the RISKEE based method selects hardware components with a higher amf than the BAG based approach. Especially the selection of the SE with the highest EAL (EAL6+) is of importance, as it stores the master key, and hosts the task deriving the session key. Thus, the framework must select SEs with high-security levels. Otherwise, the high impact of the key disclosure attacks would cause their attached security goals to exceed their thresholds.

The combined approach, which uses the BAG based calculation together with RISKEE, produced the same amount of secure solutions as the RISKEE based approach. The combined approach first calculates the asp of each goal using the BAG method. Only if this calculation produces an asp not exceeding the goal's threshold, the combined approach recalculates the goal's mean risk using the RISKEE method. Hence, by delivering the same amount of secure solutions as the RISKEE based approach, it shows that all solutions found by the RISKEE method are also contained in the secure solutions found by the BAG approach. The combined method is capable of producing the same set of secure solutions. However, for the insecure solutions, it can occur that the combined approach yields more broken goals than the BAG or RISKEE based method alone, as some goals may exceed their thresholds in the BAG but not in the RISKEE based calculation, and vice versa.

Table IV shows the fastest and most secure solutions found by the framework. The thresholds of the attack goals do not constrain the fastest solution. However, the solution still considers the feasibility of the $SAct$ when mapping the tasks to the hardware components. The most secure solution selects the task mapping and system partitioning with the lowest AP_{avg} and MRV_{avg} , respectively.

Table V describes the fastest secure solution found by the BAG and RISKEE method. Table VI the most power-efficient secure solutions, given their average attack probability and average mean risk. One can see that for both the secure solutions with optimal performance and power consumption, the

TABLE IV: Most secure and fastest solution found based on AP_{avg} , and MRV_{avg} , with the delay normalized to system with lowest delay.

HWC	Options (most secure)	Options (fastest)
AP (node & anchor)	HWC, TEE	SWC f.
SE (node & anchor)	EAL 6+	EAL 4+
UR (node & anchor)	HWC, MS	SWC f.
AP_{avg} / MRV_{avg}	0.0007 / 117.45\$	0.005 / 3905.37\$
norm delay	~1.73	1.0

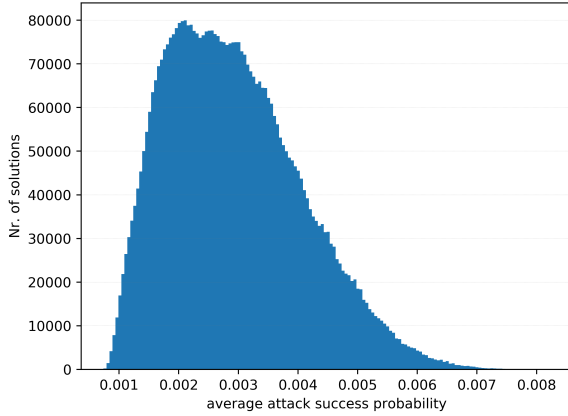
TABLE V: Fastest secure solutions found based on AP_{avg} , MRV_{avg} , and the delay normalized to system with lowest delay.

HWC	fastest secure (BAG)	fastest secure (RISKEE)
AP (node & anchor)	HWC, TEE	HWC, TEE
SE (node)	EAL 4+	EAL 6+
SE (anchor)	EAL 4+	EAL 6+
UR	HWC, TZ	HWC, FW
AP_{avg} / MRV_{avg}	0.0017	126.74\$
norm. delay	~1.074	~1.16

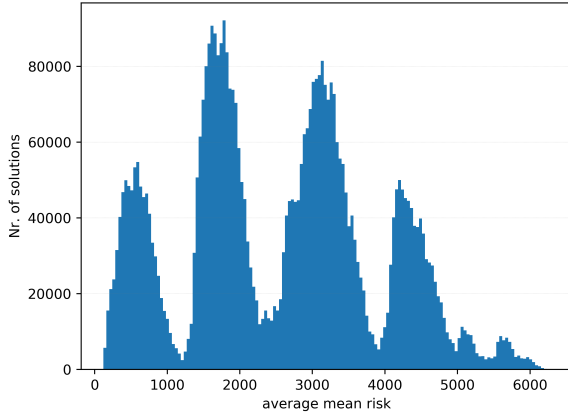
RISKEE based solution chooses options with higher security attack mitigation capabilities than the BAG based approach, for both the SE and the AP of the anchor and node device. The more secure SE is chosen because of the high impact a possible disclosure of the session key or the master key causes. Said impact increases the influence of a successful key exposure on the average mean risk of the overall system dramatically. A similar result can be seen when considering the most power-efficient and secure solutions, regarding their average AP_{avg} and MRV_{avg} , respectively. Also, for this optimization criteria, the BAG based method chose less secure options for the SE and the node's AP, compared to the RISKEE based method.

Figures 5b and 5a show the numbers of found solutions ordered by their AP_{avg} and MRV_{avg} , respectively. One should notice that the BAG based method produced more secure solutions, considering their number. The majority of the found solutions (43.08%) have an AP_{avg} between 0.0007 and 0.0026. Only 0.38% are found to have an AP_{avg} exceeding 0.006. Considering the RISKEE based approach, the histogram shows four peaks (placed at the MRV_{avg} of 656\$, 1845\$, 3444\$, and 4223\$). These peaks show how the RISKEE based approach delivers more pointedly results compared to the BAG based method. The RISKEE based method also delivers fewer secure solutions. Only 28% of the solutions induce an MRV_{avg} of less than 1658\$, whereas 39.48% come with an MRV_{avg} higher than 1658\$ and lower than 3196\$. Only 4.58% of the found solutions are regarded as highly risky, coming with an MRV_{avg} of more than 4735\$. The combined approach mixes both risk values in \$ and attack probabilities in %. Hence, a histogram over the solutions found by the combined approach gives no further insight.

We show how we use the framework for designing secure systems. We model the security constraints both with the BAG based and the RISKEE based approach. With the modeled use case, we show how the impact and frequency added by the RISKEE method influences the calculation of the security constraints, defining the attack consequences more precisely. This



(a) BAG based solutions found by the framework categorized according to their average attack success probability. Stepsize of 4.99×10^{-5}



(b) RISKEE based solutions found by the framework categorized according to their average mean risk. Stepsize of 41\$

Fig. 5: Solutions found for the secure ranging use case modeled using BAGs or RISKEE.

TABLE VI: Most power efficient and secure solutions found based on AP_{avg} , MRV_{avg} , and power consumption (power cons) normalized to solution with lowest sys_{pwr}

HWC	most power eff. secure (BAG)	most power eff. secure (RISKEE)
AP (node)	HWC, TEE	HWC TEE
AP (anchor)	HWC, TEE	HWC TEE
SE (node)	EAL 4+	EAL 6+
SE (anchor)	EAL 4+	EAL 6+
UR (node)	HWC, TZ	HWC, MS
UR (anchor)	HWC, TZ	HWC, MS
avg ap / rv	0.0013	126.17\$
power cons	~1.0415	~1.068

more accurate definition comes, however, with the drawback of higher computation time. Hence, the framework introduces the combined approach of BAG and RISKEE. Section V shows the performance of the different methods.

V. TIMING EVALUATION

The framework was executed on a system providing 16 GB of RAM and an Intel® Core™ i7-4600U CPU with 2.10 GHz.

TABLE VII: Computational overhead for BAG, RISKEE and combined approach for the different variants.

Variants	BAG	RISKEE	Combo
Var. I	910.253s	1656.605s	1218.024s
Var. II	27.241s	297.445s	262.56s
Var. III	15.357s	112.092s	80.649s

We executed the use case using the BAG, the RISKEE, and the combined approach of BAG and RISKEE (Combo). We compared the execution time with and without using the permutation limitation (pl), as explained in Section III. Table VII and VIII list the execution times (with and without pl) for the use case presented in here. We split the use case into three variants. The first variant (Var. I) is the full execution of the whole use case. The second variant (Var. II) consists of the authentication and session establishing phase, leaving out the ranging phase. The third variant (Var. III) consists only of the authentication phase. As the variants use different phases, they also come with different numbers of found solutions. These solutions also include insecure mappings, as those also influence the computation time of the framework. Var. I consists of 947072 solutions, Var. II of 19728 solutions, and Var. III of 580 solutions. The timings presented in the tables reflect the actual CPU times (spent in user mode and kernel).

TABLE VIII: Computational overhead for BAG, RISKEE and combined approach for the different variants. With permutation limitation (pl)

Variants	BAG (pl)	RISKEE (pl)	Combo (pl)
Var. I	453.326s	1072.002s	744.09s
Var. II	26.165s	214.594s	179.992s
Var. III	13.824s	51.529s	45.966s

The timings listed in Table VII and VIII show how the different approaches scale. The framework's execution time contains a dynamic part which only depends on the computation time of finding the solutions. It further also contains a static performance overhead consisting of a ramp-up and clean up phase. Hence, the execution time spent per solution decreases with the increase in calculated solutions. The framework comes with an worst-case execution time of 27min37s considering the complete use case solutions.

As can be seen in Table VII, the chosen approach for calculating the security constraints has a great impact on the overall execution time needed for finding the task mappings and system partitioning. The greatest difference between the execution time lies between the BAG and RISKEE based approach. The additional execution time needed for the RISKEE method compared to the BAG approach comes with ~82% for Var. I. For this variant, the combined BAG and RISKEE approach comes with a speedup of 26.47%, compared to the RISKEE based approach. Figure 6 depicts the calculation times. The usage of the break criteria also increases the execution speed of the security constraint calculation. The permutation limitation speeds up the calculation using the

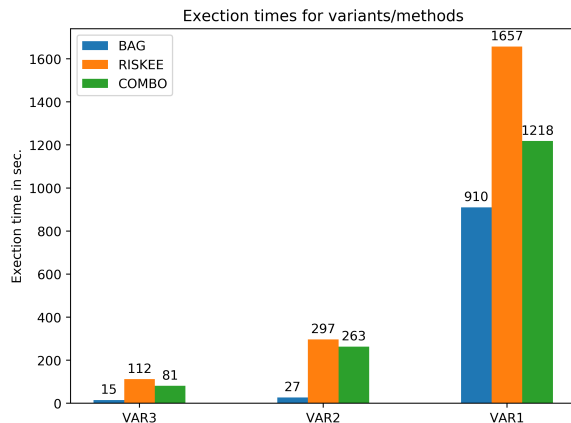


Fig. 6: Execution times for the different methods calculating the use case variants.

BAG method by 50.2%, the RISKEE method by 35.29%, and the combined method by 38.91%, when calculating the security constraints for Var. I. The execution time greatly depends on the attack scenarios modeled for the use case.

VI. LIMITATIONS AND CONCLUSION

The main limitation of the approach presented here is its dependency on accurate estimations of potential attacks and the effectiveness of their countermeasures. To estimate attacks accurately, one should follow state-of-the-art estimation techniques, such as [16], [18], [19]. These methods are usable when capturing the attack success probabilities of known systems. Especially the CVSS presents a good base as it covers attack complexity and impact metrics from which one can draw an estimation of the attacker's motivation. Attacks on known systems can be used to estimate the potential of new attacks, as many of them share the same base. However, for completely new attacks the approach presented here is not applicable. Furthermore, considering the estimation for the countermeasures, no method has yet been published on how to rate a system's ability to withstand security attacks. Our assumptions made for the use case stem from CC certifications. Here, we linked the attack's complexity from the CVSS to the mitigation capability of the CC regarding an attacker's strength. Methods on how to estimate a system's vulnerability to completely new attacks is out of this paper's scope.

The main purpose of our approach is to provide a design framework for integrating a system's attack susceptibility with its traditional requirements on performance, power consumption, etc. The framework offers the system designers to model security constraints as BAGS or risk trees, in an early design phase. It automatically calculates the task mappings and system partitioning to arrive at secure system solutions, given its inputs. It is capable of providing the designer with secure solutions adhering to various other system constraints, such as performance, power consumption, etc. Using a secure access system use case, we showed the feasibility and the

scalability of our approach. By introducing a combined calculation method, which first calculates attack goals using the BAG approach and then verifies them using the risk trees, we were able to speed up the overall calculation time.

ACKNOWLEDGMENT

Project partners are NXP Semiconductors Austria GmbH and the Technical University of Graz. This work was supported by the Austrian Research Promotion Agency (FFG) within the project UBSmart (project number: 859475).

REFERENCES

- [1] Seokung Yoon et al. Security issues on smarthome in iot environment. In *Computer Science and its Applications*. Springer Berlin Heidelberg, 2015.
- [2] Mohammed Nasser et al. Cyber-Security Incidents: A Review Cases in Cyber-Physical Systems. *Int. Journal of Advanced Computer Science and Applications*, 9, 2018.
- [3] Ruozhou Yu et al. Deploying robust security in internet of things. *2018 IEEE Conf. on Communications and Network Security, CNS 2018*, 2018.
- [4] Kathrin Rosvall et al. Exploring power and throughput for dataflow applications on predictable NoC multiprocessors. *Proc. - 21st Euromicro Conf. on Digital System Design, DSD 2018*, 2018.
- [5] Yong Xie et al. Security/Timing-aware Design Space Exploration of CAN FD for Automotive Cyber-Physical Systems. *IEEE Transactions on Industrial Informatics*, 2018.
- [6] Ingo Stierand et al. Integrating the security aspect into design space exploration of embedded systems. *Proc. of IEEE 25th Int. Symp. on Software Reliability Engineering Workshops, ISSREW 2014*, 2014.
- [7] Bastian Knerr. *Heuristic Optimisation Methods for System Partitioning in HW / SW Co-Design*. PhD thesis, Vienna University of Technology, 2008.
- [8] Kathrin Rosvall et al. Throughput Propagation in Constraint-Based Design Space Exploration for Mixed-Criticality Systems. *Proc. of the 9th Workshop on Rapid Simulation and Performance Evaluation: Methods and Tools - RAPIDO '17*, 2017.
- [9] Yves Roudier and Ludovic Apvrille. SysML-Sec - A Model Driven Approach for Designing Safe and Secure Systems. *Proc. of the 3rd Int. Conf. on Model-Driven Engineering and Software Development*, 2015.
- [10] Bowen Zheng et al. Cross-Layer Codesign for Secure Cyber-Physical Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2016.
- [11] Nan Feng et al. A security risk analysis model for information systems: Causal relationships of risk factors and vulnerability propagation analysis. *Information Sciences*, 2014.
- [12] Nayot Poolsappasit et al. Dynamic Security Risk Management Using Bayesian Attack Graphs. *IEEE Transactions on Dependable and Secure Computing*, 2012.
- [13] Michael Krisper et al. RISKEE : A Risk-Tree Based Method for Assessing Risk in Cyber Security. In *Proc. of EuroSPI 2019: European System, Software & Service Process Improvement & Innovation*, 2019.
- [14] L. Gressl et al. Consideration of Security Attacks in the Design Space Exploration of Embedded Systems. In *2019 22nd Euromicro Conf. on Digital System Design (DSD)*, 2019.
- [15] David Heckerman and John S. Breese. Causal independence for probability assessment and inference using Bayesian networks. *IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans*, 1996.
- [16] M. Schiffman. Common Vulnerability Scoring System (CVSS). URL <https://www.first.org/cvss/v3.1/specification-document>, 2019.
- [17] ISO/IEC. Common Criteria for Information Technology Security Evaluation Part 2. *Security*, (September), 2012.
- [18] Kim Fenrich. Securing your control system: the "cia triad" is a widely used benchmark for evaluating information system security effectiveness. *Power Engineering*, 112(2):44-49, 2008.
- [19] Shawn Hernan, Scott Lambert, Tomasz Ostwald, and Adam Shostack. Uncover security design flaws using the stride approach (2006). URL <http://msdn.microsoft.com/en-gb/magazine/cc163519.aspx>, 15.

Bibliography

- [1] SAE International, “J3016 Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles (J3016_201806),” 2018. [Online]. Available: https://saemobilus.sae.org/content/J3016_201806/ → Cited on page 1.
- [2] J. Shuttleworth, “SAE Standards News: J3016 automated-driving graphic update,” 2019. [Online]. Available: <https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic> → Cited on page 1.
- [3] D. C. Miller and C. Valasek, “Remote Exploitation of an Unaltered Passenger Vehicle,” 2015. → Cited on page 1.
- [4] L. Wouters, E. Marin, T. Ashur, B. Gierlichs, and B. Preneel, “Fast, Furious and Insecure: Passive Keyless Entry and Start Systems in Modern Supercars,” *IACR Transactions on Cryptographic Hardware and Embedded Systems*, vol. 2019, no. 3, pp. 66–85, May 2019. [Online]. Available: <https://tches.iacr.org/index.php/TCHES/article/view/8289> → Cited on pages 1 and 82.
- [5] proofpoint, “2020 State of the Phish: An in-depth look at user awareness, vulnerability and resilience,” proofpoint, Annual Report, 2020. → Cited on page 1.
- [6] Ponemon Institute, “Cost of Insider Threats 2020,” IBM Security and Ponemon Institute, Tech. Rep., 2020. [Online]. Available: <https://www.ibm.com/downloads/cas/LQZ4RONE> → Cited on pages 1 and 50.
- [7] HackerOne, “The 2021 Hacker Report,” HackerOne, Annual Report, 2021. [Online]. Available: <https://www.hackerone.com/resources/reporting/the-2021-hacker-report> → Cited on page 1.
- [8] CrowdStrike, “2021 Global Thread Report,” CrowdStrike, Annual Report, 2021. [Online]. Available: <https://go.crowdstrike.com/rs/281-OBQ-266/images/Report2021GTR.pdf> → Cited on page 1.
- [9] Juniper Research, “Juniper Research: The Future of Cybercrime and Security,” *Computer Fraud and Security*, vol. 2018, no. 9, p. 4, Sep. 2018. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1361372318300824> → Cited on page 2.
- [10] K. Bissell, R. M. Lasalle, and P. D. Cin, “State of Cybersecurity Report 2020,” Accenture, Tech. Rep., 2020. [Online]. Available: <https://www.accenture.com/us-en/insights/security/invest-cyber-resilience> → Cited on page 2.
- [11] R. M. Cooke, D. Marti, and T. Mazzuchi, “Expert forecasting with and without uncertainty quantification and weighting: What do the data say?” *International Journal of Forecasting*, vol. 37, no. 1, pp. 378–387, Jan. 2021. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0169207020300959> → Cited on pages 5, 25, 61, and 77.

- [12] A. L. Cox, “What’s Wrong with Risk Matrices?” *Risk Analysis*, vol. 28, no. 2, pp. 497–512, Apr. 2008. [Online]. Available: <http://doi.wiley.com/10.1111/j.1539-6924.2008.01030.x> → Cited on page 5.
- [13] D. W. Hubbard and R. Seiersen, *How to measure anything in cybersecurity risk*. Hoboken: Wiley, 2016. → Cited on pages 5, 25, 34, and 62.
- [14] R. M. Cooke, *Experts in uncertainty: opinion and subjective probability in science*, ser. Environmental ethics and science policy series. New York: Oxford University Press, 1991. → Cited on pages 5, 23, 25, 60, 61, 62, 63, and 77.
- [15] J. Navarro, A. Deruyver, and P. Parrend, “A systematic survey on multi-step attack detection,” *Computers & Security*, vol. 76, pp. 214–249, Jul. 2018, 00000. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0167404818302141> → Cited on page 6.
- [16] M. Albanese and S. Jajodia, “A Graphical Model to Assess the Impact of Multi-Step Attacks,” *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, vol. 15, no. 1, pp. 79–93, Jan. 2018. [Online]. Available: <http://journals.sagepub.com/doi/10.1177/1548512917706043> → Cited on pages 7 and 23.
- [17] P. A. Watters, S. McCombie, R. Layton, and J. Pieprzyk, “Characterising and predicting cyber attacks using the Cyber Attacker Model Profile (CAMP),” *Journal of Money Laundering Control*, vol. 15, no. 4, pp. 430–441, Oct. 2012. [Online]. Available: <https://www.emerald.com/insight/content/doi/10.1108/13685201211266015/full/html> → Cited on page 7.
- [18] S. Li, R. Rickert, and A. Sliva, “Risk-Based Models of Attacker Behavior in Cybersecurity,” in *Social Computing, Behavioral-Cultural Modeling and Prediction*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, vol. 7812, pp. 523–532, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-642-37210-0_57 → Cited on page 7.
- [19] Merriam-Webster Online Dictionary, “Risk,” 2020. [Online]. Available: <https://www.merriam-webster.com/dictionary/risk> → Cited on page 9.
- [20] Cambridge Online Dictionary, “Risk,” 2020. [Online]. Available: <https://dictionary.cambridge.org/dictionary/english/risk> → Cited on page 9.
- [21] Oxford English Dictionary, “Risk,” 2020. → Cited on page 9.
- [22] ISO, “ISO/Guide 73:2009(en), Risk management – Vocabulary,” ISO, Tech. Rep., 2009. [Online]. Available: <https://www.iso.org/standard/44651.html> → Cited on pages 9 and 28.
- [23] —, “ISO 31000:2018 Risk management – Guidelines,” ISO, Tech. Rep., 2018, 00000. [Online]. Available: <https://www.iso.org/iso-31000-risk-management.html> → Cited on pages 9, 10, and 28.
- [24] P. Slovic and E. U. Weber, “Perception of Risk Posed by Extreme Events,” in *Risk Management strategies in an Uncertain World*, Apr. 2002, p. 21. → Cited on page 10.
- [25] B. B. Mandelbrot and R. L. Hudson, *The (mis)behaviour of markets: a fractal view of risk, ruin, and reward*. London: Profile, 2008. → Cited on page 10.
- [26] N. N. Taleb, *The black swan: the impact of the highly improbable*, 2nd ed. New York: Random House Trade Paperbacks, 2010, oCLC: ocn213400968. → Cited on page 10.

-
- [27] M. Wucker, *The gray rhino: how to recognize and act on the obvious dangers we ignore*, 1st ed. New York: St. Martin's Press, 2016. → Cited on page 10.
- [28] D. A. Shore, "Today's Leadership Lesson: Mind the Wildlife and Prepare for Tomorrow's Disruption," *Journal of Health Communication*, vol. 25, no. 4, pp. 301–302, Apr. 2020. [Online]. Available: <https://www.tandfonline.com/doi/full/10.1080/10810730.2020.1749193> → Cited on page 10.
- [29] K. Munro, "China cabinet: black swans, grey rhinos, an elephant in the room," Jan. 2019. [Online]. Available: <https://www.lowyinstitute.org/the-interpreter/china-cabinet-black-swans-grey-rhinos-elephant-room> → Cited on page 10.
- [30] UNESCO, "UNESCO moving forward the 2030 Agenda for Sustainable Development," UNESCO, Tech. Rep., 2017. [Online]. Available: <https://en.unesco.org/creativity/sites/creativity/files/247785en.pdf> → Cited on page 10.
- [31] L. A. T. Cox, Jr., "What's Wrong with Hazard-Ranking Systems? An Expository Note," *Risk Analysis*, vol. 29, no. 7, pp. 940–948, Jul. 2009. [Online]. Available: <http://doi.wiley.com/10.1111/j.1539-6924.2009.01209.x> → Cited on page 11.
- [32] L. A. Cox, "Some Limitations of "Risk = Threat × Vulnerability × Consequence" for Risk Analysis of Terrorist Attacks," *Risk Analysis*, vol. 28, no. 6, pp. 1749–1761, Dec. 2008. [Online]. Available: <http://doi.wiley.com/10.1111/j.1539-6924.2008.01142.x> → Cited on page 11.
- [33] M. Krisper, J. Dobaj, and G. Macher, "Pitfalls, Fallacies, and Other Problems in Risk Matrices Using Ordinal Scales," *Submitted, Currently Under Review*, p. 11, 2019. → Cited on page 11.
- [34] L. A. Gordon and M. P. Loeb, "The Economics of Information Security Investment," *ACM Transactions on Information and System Security*, vol. 5, no. 4, p. 20, 2002. → Cited on page 11.
- [35] S. S. Stevens, "On the Theory of Scales of Measurement," *Science*, vol. 103, no. 2684, pp. 677–680, 1946. → Cited on page 11.
- [36] D. G. Malcolm, J. H. Roseboom, C. E. Clark, and W. Fazar, "Application of a technique for research and development program evaluation," *Operations Research*, vol. 7, no. 5, pp. 646–669, 1959. [Online]. Available: <https://doi.org/10.1287/opre.7.5.646> → Cited on pages 15 and 35.
- [37] J. Freund, *Measuring and managing information risk: a FAIR approach*. Amsterdam: Butterworth-Heinemann, 2015. → Cited on pages 15, 23, 25, 35, 36, and 41.
- [38] The Open Group, "FAIR – ISO/IEC 27005 cookbook (c103)," The Open Group, Tech. Rep., 2010. → Cited on pages 15, 35, and 36.
- [39] D. Vose, *Risk analysis: a quantitative guide*, 3rd ed. Chichester, England ; Hoboken, NJ: Wiley, 2008, oCLC: ocn174112755. → Cited on pages 16, 35, and 63.
- [40] B. A. Turlach, "Bandwidth Selection in Kernel Density Estimation: A Review," *Handbook of Systemic Autoimmune Diseases*, p. 34, 1999. → Cited on page 18.
- [41] N.-B. Heidenreich, A. Schindler, and S. Sperlich, "Bandwidth selection for kernel density estimation: a review of fully automatic selectors," *AStA Advances in Statistical Analysis*, vol. 97, no. 4, pp. 403–433, Oct. 2013. [Online]. Available: <http://link.springer.com/10.1007/s10182-013-0216-y> → Cited on page 18.
-

- [42] N. Silver, “2016 Election Forecast,” Jun. 2016. [Online]. Available: <https://projects.fivethirtyeight.com/2016-election-forecast/> → Cited on page 18.
- [43] D. W. Scott, “Multivariate Density Estimation and Visualization,” in *Handbook of Computational Statistics*, J. E. Gentle, W. K. Härdle, and Y. Mori, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 549–569. [Online]. Available: http://link.springer.com/10.1007/978-3-642-21551-3_19 → Cited on page 18.
- [44] C. Chesneau, F. Comte, and F. Navarro, “Fast nonparametric estimation for convolutions of densities,” *Canadian Journal of Statistics*, vol. 41, no. 4, pp. 617–636, Dec. 2013. [Online]. Available: <http://doi.wiley.com/10.1002/cjs.11191> → Cited on page 18.
- [45] H. Ku, “Notes on the use of propagation of error formulas,” *Journal of Research of the National Bureau of Standards, Section C: Engineering and Instrumentation*, vol. 70C, no. 4, p. 263, Oct. 1966. [Online]. Available: https://nvlpubs.nist.gov/nistpubs/jres/70C/jresv70Cn4p263_A1b.pdf → Cited on page 20.
- [46] JCGM, “JCGM 100:2008 Evaluation of measurement data – Guide to the expression of uncertainty in measurement,” BIPM, Tech. Rep., Sep. 2008. [Online]. Available: https://www.bipm.org/utls/common/documents/jcgm/JCGM_100_2008_E.pdf → Cited on page 20.
- [47] B. D. Hall, “THE GUM TREE DESIGN PATTERN FOR UNCERTAINTY SOFTWARE,” in *Advanced Mathematical and Computational Tools in Metrology VI*. World Scientific Publishing Co. Pte. Ltd., 2004, pp. 199–208. [Online]. Available: http://eproceedings.worldscinet.com/9789812702647/9789812702647_0017.html → Cited on page 20.
- [48] M. H. DeGroot and J. Mortera, “Optimal Linear Opinion Pools,” *Management Science*, vol. 37, no. 5, pp. 546–558, May 1991. [Online]. Available: <http://pubsonline.informs.org/doi/abs/10.1287/mnsc.37.5.546> → Cited on page 20.
- [49] L. A. T. Cox, “Some Limitations of “Risk = Threat × Vulnerability × Consequence” for Risk Analysis of Terrorist Attacks,” *Risk Analysis*, vol. 28, no. 6, pp. 1749–1761, Dec. 2008. → Cited on page 21.
- [50] A. L. Cox and D. A. Popken, “Some Limitations of Aggregate Exposure Metrics,” *Risk Analysis*, vol. 27, no. 2, pp. 439–445, 2007. → Cited on page 21.
- [51] A. L. Cox, D. Babayev, and W. Huber, “Some Limitations of Qualitative Risk Rating Systems,” *Risk Analysis*, vol. 25, no. 3, pp. 651–662, 2005. → Cited on page 21.
- [52] A. N. Kolmogorov, V. M. Tikhomirov, and A. N. Kolmogorov, “On the Notion of Mean,” in *Mathematics and mechanics*, ser. Selected works of A.N. Kolmogorov. Dordrecht ; Boston: Kluwer Academic Publishers, 1991, no. v. 1. → Cited on page 21.
- [53] Kenneth B. Stolarsky, “Generalizations of the Logarithmic Mean,” *Mathematics Magazine*, vol. 48, no. 2, pp. 87–92, Mar. 1975. → Cited on page 21.
- [54] M. de Carvalho, “Mean, What do You Mean?” *The American Statistician*, vol. 70, no. 3, pp. 270–274, Jul. 2016. [Online]. Available: <https://www.tandfonline.com/doi/full/10.1080/00031305.2016.1148632> → Cited on page 21.

-
- [55] C. O. Imoru, "The power mean and the logarithmic mean," *International Journal of Mathematics and Mathematical Sciences*, vol. 5, no. 2, pp. 337–343, 1982. [Online]. Available: <http://www.hindawi.com/journals/ijmms/1982/718951/abs/> → Cited on page 21.
- [56] P. S. Bullen, *Handbook of means and their inequalities*, ser. Mathematics and its applications. Dordrecht ; Boston: Kluwer Academic Publishers, 2003, no. v. 560. → Cited on page 21.
- [57] S. Caltagirone, A. Pendergast, and C. Betz, "The Diamond Model of Intrusion Analysis," Center for Cyber Intelligence Analysis and Threat Research Hanover MD, Tech. Rep., 2013, 00000. → Cited on pages 23, 24, 28, 33, 34, 41, and 53.
- [58] V. Hemming, T. V. Walshe, A. M. Hanea, F. Fidler, and M. A. Burgman, "Eliciting improved quantitative judgements using the IDEA protocol: A case study in natural resource management," *PLOS ONE*, vol. 13, no. 6, Jun. 2018. → Cited on pages 23, 25, 60, and 77.
- [59] NASA, "Fault Tree Analysis - A Bibliography," NASA Langley Technical Report Server, Tech. Rep. NASA/SP-2000-6111, Jul. 2000. → Cited on pages 23 and 31.
- [60] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, *Fault Tree Handbook (NUREG-0492)*. U.S. Nuclear Regulatory Commission, 1981. → Cited on pages 23 and 33.
- [61] B. Schneier, C. Salter, O. S. Saydjari, and J. Wallner, "Toward A Secure System Engineering Methodology," NSA, Charlottesville, VA, USA, Tech. Rep., 1998, 00002. [Online]. Available: <https://www.schneier.com/academic/paperfiles/paper-secure-methodology.pdf> → Cited on page 23.
- [62] B. Schneier, "Attack Trees - Modeling security threats," *Dr. Dobb's Journal*, vol. 24, no. 12, pp. 21–29, 1999, 00000. → Cited on pages 23, 28, 33, and 41.
- [63] N. Poolsappasit, R. Dewri, and I. Ray, "Dynamic Security Risk Management Using Bayesian Attack Graphs," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 1, pp. 61–74, Jan. 2012, 00000. → Cited on pages 23 and 88.
- [64] R. Wolthuis and F. Phillipson, "Quantifying Cyber security Risks," in *Innovating in Cyber Security*. Netherlands: TNO, 2019, pp. 20–26. → Cited on page 24.
- [65] W. Wu, R. Kang, and Z. Li, "Risk assessment method for cyber security of cyber physical systems," in *2015 First International Conference on Reliability Systems Engineering (ICRSE)*. Beijing, China: IEEE, Oct. 2015, pp. 1–5. [Online]. Available: <http://ieeexplore.ieee.org/document/7366430/> → Cited on page 24.
- [66] The Open Group, "Risk Analysis (O-RA)," The Open Group, Tech. Rep., Oct. 2013. → Cited on pages 25, 28, 34, and 35.
- [67] —, "Risk Taxonomy (O-RT) 2.0," The Open Group, Tech. Rep., Oct. 2013. → Cited on pages 25, 35, 45, 48, and 49.
- [68] RiskAMP, "The beta-PERT Distribution | RiskAMP," 2010. [Online]. Available: <https://www.riskamp.com/beta-pert> → Cited on pages 25 and 35.
- [69] N. Silver, *The signal and the noise: why so many predictions fail—but some don't*. Penguin Books, 2015, OCLC: 934814149. [Online]. Available: <http://www.myilibrary.com?id=712071> → Cited on page 25.

- [70] R. M. Cooke, “Quantifying uncertainty on thin ice: Expert judgement assessment,” *Nature Climate Change*, vol. 3, no. 4, pp. 311–312, Apr. 2013. [Online]. Available: <http://www.nature.com/articles/nclimate1860> → Cited on pages 25 and 63.
- [71] V. Hemming, M. A. Burgman, A. M. Hanea, M. F. McBride, and B. C. Wintle, “A practical guide to structured expert elicitation using the IDEA protocol,” *Methods in Ecology and Evolution*, vol. 9, no. 1, pp. 169–180, Jan. 2018. → Cited on page 25.
- [72] H. A. Linstone and M. Turoff, Eds., *The Delphi method: Techniques and applications*. Reading, Mass. [usw.]: Addison-Wesley, 1975, oCLC: 251991541. → Cited on page 25.
- [73] T. Gilovich, D. Griffin, and D. Kahneman, *Heuristics and Biases: The Psychology of Intuitive Judgment*. Cambridge, U.K. ; New York: Cambridge University Press, Jul. 2002. → Cited on page 25.
- [74] D. Kahneman and A. Tversky, “Subjective probability: A judgement of representativeness,” *Cognitive psychology*, vol. 3, no. 3, pp. 430–454, 1972. [Online]. Available: <http://datacolada.org/wp-content/uploads/2014/08/Kahneman-Tversky-1972.pdf> → Cited on pages 25, 61, and 62.
- [75] P. E. Tetlock and D. Gardner, *Superforecasting: the art and science of prediction*, 1st ed. New York: Crown Publishers, 2015. → Cited on page 25.
- [76] National Institute of Standards and Technology, “Framework for Improving Critical Infrastructure Cybersecurity, Version 1.1,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. NIST CSWP 04162018, Apr. 2018. [Online]. Available: <http://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.04162018.pdf> → Cited on pages 26 and 28.
- [77] Joint Task Force Transformation Initiative, “NIST SP 800-30r1 Guide for conducting risk assessments,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. NIST SP 800-30r1, 2012. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf> → Cited on pages 26, 28, and 29.
- [78] VDA QMC, “White Paper FMEA,” VDA QMC, Tech. Rep., 2019. → Cited on pages 26 and 30.
- [79] ISO, “ISO 26262-1:2018,” ISO, Tech. Rep., 2018. [Online]. Available: <https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/83/68383.html> → Cited on pages 26, 28, 29, 30, and 31.
- [80] FIRST, “Common Vulnerability Scoring System version 3.1 Revision 1,” FIRST, Tech. Rep., 2019. [Online]. Available: https://www.first.org/cvss/v3-1/cvss-v31-specification_r1.pdf → Cited on pages 26, 28, and 35.
- [81] A. S. Tanenbaum and M. v. Steen, *Distributed systems: principles and paradigms*, 2nd ed. Harlow, Essex: Pearson Education, 2014. → Cited on page 26.
- [82] A. Puder, K. Römer, and F. Pilhofer, *Distributed systems architecture: a middleware approach*. Amsterdam; Boston: Elsevier, 2006. → Cited on page 26.
- [83] T. Amorim, D. Ratasich, G. Macher, A. Ruiz, D. Schneider, M. Driussi, and R. Grosu, “Runtime Safety Assurance for Adaptive Cyber- Physical Systems: ConSerts M and Ontology-Based Runtime Reconfiguration Applied to an Automotive Case Study,” in *Solutions for Cyber-Physical Systems Ubiquity*. IGI Global, 2018, pp. 137–168. → Cited on page 26.

-
- [84] A. Bondavalli and L. Simoncini, "Failure classification with respect to detection," in *Proceedings of the Second IEEE Workshop on Future Trends of Distributed Computing Systems*, Sep. 1990, pp. 47–53. → Cited on page 26.
- [85] P. D. Ezhilchelvan, S. K. Shrivastava, and M. J. Elphick, *A characterisation of faults in systems*. University of Newcastle upon Tyne, Computing Laboratory, 1985. → Cited on page 26.
- [86] P. Fenelon and B. Hebborn, "Applying HAZOP to Software Engineering Models," in *Risk Management And Critical Protective Systems: Proceedings of SARSS 1994*. Society, 1994, pp. 1–1. → Cited on page 26.
- [87] A. Beugnard, J.-M. Jézéquel, and N. Plouzeau, "Contract Aware Components, 10 years after," *Electronic Proceedings in Theoretical Computer Science*, vol. 37, pp. 1–11, 10 2010. → Cited on page 26.
- [88] T. Amorim, A. Ruiz, C. Dropmann, and D. Schneider, "Multidirectional Modular Conditional Safety Certificates," in *Computer Safety, Reliability, and Security*, ser. Lecture Notes in Computer Science. Springer, Cham, Sep. 2014, pp. 357–368. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-24249-1_31 → Cited on page 26.
- [89] J. Iber, T. Rauter, and C. Kreiner, "A Self-Adaptive Software System for Increasing the Reliability and Security of Cyber-Physical Systems," *Solutions for Cyber-Physical Systems Ubiquity*, pp. 223–246, 2018. → Cited on page 26.
- [90] M. Gamal Aboelimged, "Six Sigma quality: a structured review and implications for future research," *International Journal of Quality & Reliability Management*, vol. 27, no. 3, pp. 268–317, Mar. 2010. [Online]. Available: <https://www.emerald.com/insight/content/doi/10.1108/02656711011023294/full/html> → Cited on pages 28, 58, and 74.
- [91] ISO, "ISO 26262-9:2011(en), Overview of ISO 26262," ISO, Tech. Rep., 2011. [Online]. Available: <https://www.iso.org/obp/ui/iso:std:iso:26262:-9:ed-1:v1:en> → Cited on pages 28 and 29.
- [92] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, "SAHARA: A Security-Aware Hazard and Risk Analysis Method," in *Proceedings of 2015 Design, Automation and Test in Europe Conference and Exhibition (DATE)*. IEEE Conference Publications, 2015, pp. 621–624. [Online]. Available: <http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=7092463> → Cited on pages 28, 31, and 32.
- [93] IEC, "IEC 60812: Analysis techniques for system reliability - Procedure for failure mode and effects analysis (FMEA)," IEC, Tech. Rep., 2006. → Cited on pages 28 and 30.
- [94] C. Schmittner, Z. Ma, and P. Smith, "FMVEA for Safety and Security Analysis of Intelligent and Cooperative Vehicles," in *Computer Safety, Reliability, and Security*, A. Bondavalli, A. Ceccarelli, and F. Ortmeier, Eds. Cham: Springer International Publishing, 2014, vol. 8696, pp. 282–288, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-10557-4_31 → Cited on page 28.
- [95] B. Sapiro, *Binary Risk Analysis*, 1st ed. [Online]. Available: https://binary.protect.io/BRA_draft1.1.pdf → Cited on page 28.

- [96] A. Lautenbach and M. Islam, “HEAVENS - HEALing Vulnerabilities to ENhance Software Security and Safety,” HEAVENS Consortium, Tech. Rep. Dnr 2012-04625, 2016. [Online]. Available: https://autosec.se/wp-content/uploads/2018/03/HEAVENS_D2_v2.0.pdf → Cited on pages 28 and 32.
- [97] C. J. Alberts, S. G. Behrens, R. D. Pethia, and W. R. Wilson, “Operationally Critical Threat, Asset, and Vulnerability Evaluation (OCTAVE) Framework, Version 1.0:,” Defense Technical Information Center, Fort Belvoir, VA, Tech. Rep., Jun. 1999. [Online]. Available: <http://www.dtic.mil/docs/citations/ADA367718> → Cited on page 28.
- [98] M. Krisper, J. Dobaj, G. Macher, and C. Schmittner, “RISKEE: A Risk-Tree Based Method for Assessing Risk in Cyber Security,” in *Systems, Software and Services Process Improvement*, A. Walker, R. V. O’Connor, and R. Messnarz, Eds., vol. 1060. Cham: Springer International Publishing, 2019, pp. 45–56, series Title: Communications in Computer and Information Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-28005-5_4 → Cited on pages 28 and 156.
- [99] ISO, “ISO/TR 31004:2013 risk management – guidance for the implementation of iso 31000,” ISO, Tech. Rep., 2013. [Online]. Available: <https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/05/66/56610.html> → Cited on page 28.
- [100] IEC, “IEC 31010:2019 risk management – risk assessment techniques,” IEC, Tech. Rep., 2019. [Online]. Available: <https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/21/72140.html> → Cited on page 28.
- [101] Joint Task Force Transformation Initiative, “NIST Special Publication 800-39 Managing Information Security Risk: Organization, Mission, and Information System View,” National Institute of Standards and Technology, Gaithersburg, MD, Tech. Rep. NIST SP 800-39, 2011, edition: 0. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-39.pdf> → Cited on page 29.
- [102] M. A. Fikri, F. A. Putra, Y. Suryanto, and K. Ramli, “Risk Assessment Using NIST SP 800-30 Revision 1 and ISO 27005 Combination Technique in Profit-Based Organization: Case Study of ZZZ Information System Application in ABC Agency,” *Procedia Computer Science*, vol. 161, pp. 1206–1215, 2019. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S1877050919319453> → Cited on page 29.
- [103] S. Gray and J. Pfeufer, “Alignment of VDA and AIAG FMEA handbooks,” VDA GMC, Tech. Rep., 2018. → Cited on page 31.
- [104] C. A. Ericson, “Fault Tree Analysis-A History,” *Proceedings of the 17th International System Safety Conference, 1999*, pp. 87–96, 1999. [Online]. Available: <https://ci.nii.ac.jp/naid/10025437030/en/> → Cited on page 31.
- [105] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, *NUREG-0492 Fault Tree Handbook*. Washington, D.C., USA: Systems and Reliability Research Office of Nuclear Regulatory Research U.S. Nuclear Regulatory Commission, Jan. 1981. → Cited on page 31.
- [106] M. Stamatelatos, W. Vesely, J. Dugan, J. Fragola, J. Minarick, and J. Railsback, “Fault Tree Handbook with Aerospace Applications,” NASA, Tech. Rep., 2002. → Cited on page 31.

-
- [107] IEC, *IEC 61025 Fault tree analysis (FTA)*. IEC, 2006, 00000 OCLC: 1074793189. → Cited on page 31.
- [108] G. Macher, E. Armengaud, E. Brenner, and C. Kreiner, “A Review of Threat Analysis and Risk Assessment Methods in the Automotive Context,” in *Computer Safety, Reliability, and Security - 35th International Conference, SAFECOMP 2016*, 2016. → Cited on page 31.
- [109] E. Lisova, I. Sljivo, and A. Causevic, “Safety and security co-analyses: A systematic literature review,” *IEEE Systems Journal*, 2018. → Cited on page 31.
- [110] G. Macher, H. Sporer, R. Berlach, E. Armengaud, and C. Kreiner, “SAHARA: A Security-Aware Hazard and Risk Analysis Method,” in *Design, Automation & Test in Europe Conference & Exhibition (year), 2015*, 2015. → Cited on page 31.
- [111] Microsoft Corporation, “The STRIDE Threat Model,” Microsoft Corporation, Tech. Rep., 2005. [Online]. Available: [https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878\(v=cs.20\)](https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)) → Cited on pages 31 and 32.
- [112] C. Schmittner, T. Gruber, P. Puschner, and E. Schoitsch, “Security Application of Failure Mode and Effect Analysis (FMEA),” in *33rd International Conference, SAFECOMP 2014*. Springer, Cham, 2014, pp. 310–325. → Cited on page 32.
- [113] IEC, “Analysis techniques for system reliability – Procedure for failure mode and effects analysis (FMEA),” IEC, Tech. Rep., 2006. → Cited on page 32.
- [114] M. Muckin and S. C. Fitch, “A Threat-Driven Approach to Cyber Security,” Lockheed Martin, Tech. Rep., 2015. [Online]. Available: <https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Threat-Driven-Approach.pdf> → Cited on page 33.
- [115] S. C. Fitch, “Defendable Architectures,” Lockheed Martin, Tech. Rep., 2019. [Online]. Available: <https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Defendable-Architectures.pdf> → Cited on page 33.
- [116] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, “Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains,” Lockheed Martin, Tech. Rep., Jan. 2011. [Online]. Available: <https://www.lockheedmartin.com/content/dam/lockheed-martin/rms/documents/cyber/LM-White-Paper-Intel-Driven-Defense.pdf> → Cited on page 33.
- [117] P. Ammann, D. Wijesekera, and S. Kaushik, “Scalable, graph-based network vulnerability analysis,” in *Proceedings of the 9th ACM conference on Computer and communications security - CCS '02*. Washington, DC, USA: ACM Press, 2002, p. 217. → Cited on page 33.
- [118] T. Tidwell, R. Larson, K. Fitch, and J. Hale, “Modeling Internet Attacks,” in *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security*, 2001, p. 7. → Cited on page 33.
- [119] L. Xing and S. V. Amari, “Fault Tree Analysis,” in *Handbook of Performability Engineering*, K. B. Misra, Ed. London: Springer London, 2008, pp. 595–620. → Cited on page 33.

- [120] R. Messnarz and H. Sporer, “Functional Safety Case with FTA and FMEDA Consistency Approach,” in *Systems, Software and Services Process Improvement*. Springer International Publishing, 2018. → Cited on page 33.
- [121] F. Xie, T. Lu, X. Guo, J. Liu, Y. Peng, and Y. Gao, “Security Analysis on Cyber-physical System Using Attack Tree,” in *2013 Ninth International Conference on Intelligent Information Hiding and Multimedia Signal Processing*. Beijing, China: IEEE, Oct. 2013, pp. 429–432. [Online]. Available: <http://ieeexplore.ieee.org/document/6846669/> → Cited on page 33.
- [122] X. Lyu, Y. Ding, and S.-H. Yang, “Safety and security risk assessment in cyber-physical systems,” *IET Cyber-Physical Systems: Theory & Applications*, vol. 4, no. 3, pp. 221–232, Sep. 2019. [Online]. Available: <https://digital-library.theiet.org/content/journals/10.1049/iet-cps.2018.5068> → Cited on page 33.
- [123] E. M. Hutchins, M. J. Cloppert, and R. M. Amin, “Intelligence-Driven Computer Network Defense Informed by Analysis of Adversary Campaigns and Intrusion Kill Chains,” *Leading Issues in Information Warfare & Security Research*, vol. 1, no. 1, p. 14, 2011. → Cited on page 34.
- [124] D. W. Hubbard, *How to measure anything: finding the value of intangibles in business*. John Wiley & Sons, Inc, 2014. → Cited on pages 34 and 62.
- [125] ISO/IEC, “ISO/IEC 15408: Information Technology Security Evaluation,” ISO/IEC, Tech. Rep., 2005. → Cited on page 34.
- [126] FAIR Institute, “FAIR methodology for quantifying information risk,” FAIR Institute, Tech. Rep., 2015. [Online]. Available: <https://www.fairinstitute.org> → Cited on pages 35, 36, and 71.
- [127] N. Metropolis and S. Ulam, “The Monte Carlo Method,” *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341, Sep. 1949. → Cited on page 35.
- [128] ISO/IEC, “ISO/IEC 27005:2018 information technology - security techniques - information security risk management,” ISO/IEC, Tech. Rep., 2018. → Cited on page 36.
- [129] —, “ISO/IEC 27001:2013 information technology - security techniques - information security management systems - requirements,” ISO/IEC, Tech. Rep., 2013. → Cited on page 36.
- [130] V. McCoy, “FAIR On-A-Page: Same Great Model, Fresh New Look,” May 2017. [Online]. Available: <https://www.fairinstitute.org/blog/fair-model-on-a-page> → Cited on page 36.
- [131] J. Bornholt, T. Mytkowicz, and K. S. McKinley, “Uncertain<t>: A first-order type for uncertain data,” in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS ’14. ACM, 2014, pp. 51–66. [Online]. Available: <http://doi.acm.org/10.1145/2541940.2541958> → Cited on page 37.
- [132] T. Mytkowicz, A. Diwan, M. Hauswirth, and P. F. Sweeney, “Producing Wrong Data without Doing Anything Obviously Wrong!” in *Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS XIV. New York, NY, USA: Association for Computing Machinery, 2009, pp. 265–276. [Online]. Available: <https://doi.org/10.1145/1508244.1508275> → Cited on page 37.
- [133] J. Kahlert and H. Frank, *Fuzzy-Logik und Fuzzy-Control: eine anwendungsorientierte Einführung mit Begleitsoftware*. Braunschweig: Vieweg, 1993. → Cited on page 37.

-
- [134] C. Alvarez, J. Corbal, and M. Valero, “Fuzzy Memoization for Floating-Point Multimedia Applications,” *IEEE Transactions on Computers*, vol. 54, no. 7, pp. 922–927, Jul. 2005. [Online]. Available: <http://ieeexplore.ieee.org/document/1432675/> → Cited on page 37.
- [135] E. Darulova, “Programming with Numerical Uncertainties,” Phd Thesis, ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE, Lausanne, FR, 2014. [Online]. Available: <https://people.mpi-sws.org/~eva/papers/thesis.pdf> → Cited on page 37.
- [136] E. Darulova and V. Kuncak, “Sound Compilation of Reals,” in *Proceedings of the 41st ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, ser. POPL ’14. New York, NY, USA: Association for Computing Machinery, 2014, pp. 235–248, event-place: San Diego, California, USA. [Online]. Available: <https://doi.org/10.1145/2535838.2535874> → Cited on page 37.
- [137] A. D. Gordon, T. A. Henzinger, A. V. Nori, and S. K. Rajamani, “Probabilistic Programming,” in *Proceedings of the on Future of Software Engineering*, ser. FOSE 2014. New York, NY, USA: ACM, 2014, pp. 167–181. → Cited on page 37.
- [138] J. Bornholt, T. Mytkowicz, and K. S. McKinley, “Uncertain<T>: A First-order Type for Uncertain Data,” in *Proceedings of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems*, ser. ASPLOS ’14. New York, NY, USA: ACM, 2014, pp. 51–66. [Online]. Available: <http://doi.acm.org/10.1145/2541940.2541958> → Cited on page 37.
- [139] J. Bornholt, N. Meng, T. Mytkowicz, and K. S. McKinley, “Programming the Internet of Uncertain <T>hings,” in *Sensors to Cloud Architectures Workshop (SCAW)*, 2015, p. 7. → Cited on page 37.
- [140] E. Darulova, V. Kuncak, R. Majumdar, and I. Saha, “Synthesis of fixed-point programs,” in *2013 Proceedings of the International Conference on Embedded Software (EMSOFT)*. IEEE, Sep. 2013, pp. 1–10. [Online]. Available: <http://ieeexplore.ieee.org/document/6658600/> → Cited on page 37.
- [141] A. Izycheva and E. Darulova, “On sound relative error bounds for floating-point arithmetic,” in *2017 Formal Methods in Computer Aided Design (FMCAD)*. IEEE, Oct. 2017, pp. 15–22. [Online]. Available: <http://ieeexplore.ieee.org/document/8102236/> → Cited on page 37.
- [142] JCGM, “JCGM 100:2008 evaluation of measurement data guide to the expression of uncertainty in measurement,” BIPM, Tech. Rep., 2008-09. [Online]. Available: <https://www.bipm.org/en/publications/guides/gum.html> → Cited on page 38.
- [143] —, “JCGM 102:2011 evaluation of measurement data - supplement 2 to the "guide to the expression of the uncertainty in measurement" - extension to any number of output quantities,” BIPM, Tech. Rep., 2011. [Online]. Available: <https://www.bipm.org/en/publications/guides/gum.html> → Cited on page 38.
- [144] —, “JCGM 106:2012 Evaluation of Measurement Data - The Role of Measurement Uncertainty in Conformity Assessment,” BIPM, Tech. Rep., Jan. 2013. [Online]. Available: https://www.bipm.org/utis/common/documents/jcgm/JCGM_106_2012_E.pdf → Cited on page 38.

- [145] —, “JCGM 104:2009 evaluation of measurement data - an introduction to the "guide to the expression of uncertainty in measurement" and related documents,” BIPM, Tech. Rep., 2009. [Online]. Available: <https://www.bipm.org/en/publications/guides/gum.html> → Cited on page 38.
- [146] —, “JCGM-WG1-SC1-N10 guide to the expression of uncertainty in measurement (GUM) - supplement 1: Numerical methods for the propagation of distributions,” BIPM, Tech. Rep., 2004. [Online]. Available: <https://www.bipm.org/en/publications/guides/gum.html> → Cited on page 38.
- [147] A. R. Colson and R. M. Cooke, “Expert Elicitation: Using the Classical Model to Validate Experts’ Judgments,” *Review of Environmental Economics and Policy*, vol. 12, no. 1, pp. 113–132, Feb. 2018. [Online]. Available: <https://academic.oup.com/reep/article/12/1/113/4835830> → Cited on pages 40 and 77.
- [148] A. R. Colson, I. Megiddo, G. Alvarez-Uria, S. Gandra, T. Bedford, A. Morton, R. M. Cooke, and R. Laxminarayan, “Quantifying uncertainty about future antimicrobial resistance: Comparing structured expert judgment and statistical forecasting methods,” *PLOS ONE*, vol. 14, no. 7, p. e0219190, Jul. 2019. [Online]. Available: <http://dx.plos.org/10.1371/journal.pone.0219190> → Cited on pages 40 and 77.
- [149] L. H. J. Goossens and R. M. Cooke, “Expert judgement – Calibration and combination,” p. 15, 2005. → Cited on page 40.
- [150] R. M. Cooke and L. L. Goossens, “TU Delft expert judgment data base,” *Reliability Engineering & System Safety*, vol. 93, no. 5, pp. 657–674, May 2008. → Cited on page 40.
- [151] INTERPOL, “COVID-19 Cybercrime Analysis Report- August 2020.pdf,” INTERPOL, Tech. Rep., Aug. 2020. [Online]. Available: <https://www.interpol.int/content/download/15526/file/COVID-19%20Cybercrime%20Analysis%20Report-%20August%202020.pdf> → Cited on page 47.
- [152] L. Ellison, “Second Key Note for Oracle OpenWorld customer conference,” San Francisco, Oct. 2017. [Online]. Available: <https://www.forbes.com/sites/bobevans1/2017/10/04/larry-ellison-on-cyber-attacks-its-a-war-and-were-losing-this-cyberwar/?sh=257bd2253dc6> → Cited on page 48.
- [153] M. Sailio, O.-M. Latvala, and A. Szanto, “Cyber Threat Actors for the Factory of the Future,” *Applied Sciences*, vol. 10, no. 12, p. 4334, Jun. 2020. [Online]. Available: <https://www.mdpi.com/2076-3417/10/12/4334> → Cited on page 48.
- [154] European Union Agency for Network and Information Security., *ENISA threat landscape report 2018: 15 top cyber threats and trends*. LU: Publications Office, 2019. [Online]. Available: <https://data.europa.eu/doi/10.2824/622757> → Cited on page 48.
- [155] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, “Guide to Industrial Control Systems (ICS) Security,” National Institute of Standards and Technology, Tech. Rep. NIST SP 800-82r2, Jun. 2015. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-82r2.pdf> → Cited on page 48.

-
- [156] Center for Internet Security, “Cybersecurity Spotlight – Cyber Threat Actors,” CIS Center for Internet Security, Tech. Rep., 2020. [Online]. Available: <https://www.cisecurity.org/spotlight/cybersecurity-spotlight-cyber-threat-actors/> → Cited on page 48.
- [157] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, “Equation of State Calculations by Fast Computing Machines,” *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953. [Online]. Available: <http://aip.scitation.org/doi/10.1063/1.1699114> → Cited on page 56.
- [158] S. Geman and D. Geman, “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-6, no. 6, pp. 721–741, Nov. 1984. [Online]. Available: <http://ieeexplore.ieee.org/document/4767596/> → Cited on page 56.
- [159] A. R. Colson and R. M. Cooke, “Cross validation for the classical model of structured expert judgment,” *Reliability Engineering & System Safety*, vol. 163, pp. 109–120, Jul. 2017. → Cited on pages 61, 79, and 83.
- [160] D. W. Hubbard, *The failure of risk management: why it’s broken and how to fix it*. Hoboken, N.J: Wiley, 2009, oCLC: ocn268790760. → Cited on page 61.
- [161] D. Kahneman, *Thinking, fast and slow*. London: Penguin Books, 2012, oCLC: 798805166. → Cited on page 62.
- [162] M. Krisper, J. Dobaj, and G. Macher, “Assessing Risk Estimations for Cyber-Security Using Expert Judgment,” in *Systems, Software and Services Process Improvement*. Springer International Publishing, 2020, p. 15. → Cited on pages 63, 77, and 186.
- [163] S. L. Savage, *The Flaw of Averages: Why We Underestimate Risk in the Face of Uncertainty*. Hoboken, New Jersey: John Wiley & Sons, Inc, 2012. → Cited on page 71.
- [164] T. Rose, *The end of average: how we succeed in a world that values sameness*, 1st ed. New York: HarperOne, 2015. → Cited on page 71.
- [165] E. R. Tufte, *Visual Display of Quantitative Information*, 2nd ed. Cheshire, Conn: Bertrams, Jan. 2001. → Cited on pages 71 and 72.
- [166] E. Gamma, R. Helm, R. E. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*, ser. Addison-Wesley professional computing series. Reading, Mass: Addison-Wesley, 1995. → Cited on page 73.
- [167] M. Krisper, J. Iber, J. Dobaj, and C. Kreiner, “Patterns for Implementing Uncertainty Propagation,” in *Proceedings of the 23rd European Conference on Pattern Languages of Programs*. Irsee Germany: ACM, Jul. 2018, pp. 1–6. [Online]. Available: <https://dl.acm.org/doi/10.1145/3282308.3282323> → Cited on pages 74 and 132.
- [168] M. Krisper, “Expert Judgment for Cyber-Security Risk,” EuroSPI, Tech. Rep., 2020. [Online]. Available: https://2020.eurospi.net/images/eurospi/2020/Workshop-Report-Expert-Judgment-for-Cyber-Security-Risk_v2.pdf → Cited on pages 77 and 202.

- [169] R. M. Cooke, “Supplementary Online Material for Cross Validation of Classical Model for Structured Expert Judgment,” 2016. [Online]. Available: http://rogermcooke.net/rogermcooke_files/Cross%20Validation%20SEJ%20SOM%20RESS.pdf → Cited on page 79.
- [170] H. Akoglu, “User’s guide to correlation coefficients,” *Turkish Journal of Emergency Medicine*, vol. 18, no. 3, pp. 91–93, Sep. 2018. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S2452247318302164> → Cited on page 80.
- [171] D. S. Moore, W. Notz, and M. A. Fligner, *The basic practice of statistics*, 7th ed. New York: W.H. Freeman and Company, 2015, oCLC: ocn904727526. → Cited on page 80.
- [172] J. Cohen, *Statistical power analysis for the behavioral sciences*, 2nd ed. Hillsdale, N.J: L. Erlbaum Associates, 1988. → Cited on page 80.
- [173] J. Frost, *Introduction to statistics: an Intuitive guide for analyzing data and unlocking discoveries*. Statistics By Jim Publishing, 2019, oCLC: 1239320265. → Cited on page 80.
- [174] COSIC Group, “Fast, Furious and Insecure: Passive Keyless Entry and Start Systems in Modern Supercars,” Sep. 2018. [Online]. Available: <https://www.esat.kuleuven.be/cosic/fast-furious-and-insecure-passive-keyless-entry-and-start-in-modern-supercars/> → Cited on page 82.
- [175] B. Wrozek, “Cyber Kill Chain Methodology,” OPTIV, Tech. Rep., 2017. → Cited on page 88.
- [176] P. Pols, “Designing a Unified Kill Chain for analyzing, comparing and defending against cyber attacks,” Ph.D. dissertation, Cyber Security Academy (CSA), 2017, 00000. → Cited on page 88.
- [177] B. Kaiser, P. Liggesmeyer, and O. Mackel, “A New Component Concept for Fault Trees,” in *Eighth Australian Workshop on Safety Critical Systems and Software (SCS 2003)*, ser. CRPIT, P. Lindsay and T. Cant, Eds., vol. 33. Canberra, Australia: ACS, 2004, pp. 37–46. → Cited on page 88.
- [178] L. Gressl, M. Krisper, C. Steger, and U. Neffe, “Towards an Automated Exploration of Secure IoT/CPS Design Variants,” in *Computer Safety, Reliability, and Security*. Springer International Publishing, 2020. → Cited on pages 88 and 220.
- [179] R. Dewri, N. Poolsappasit, I. Ray, and D. Whitley, “Optimal security hardening using multi-objective optimization on attack tree models of networks,” in *Proceedings of the 14th ACM conference on Computer and communications security - CCS ’07*. Alexandria, Virginia, USA: ACM Press, 2007, p. 204, 00164. → Cited on pages 88 and 89.
- [180] L. Gressl, M. Krisper, C. Steger, and U. Neffe, “Towards Security Attack and Risk Assessment during Early System Design,” in *2020 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. Dublin, Ireland: IEEE, Jun. 2020, pp. 1–8. [Online]. Available: <https://ieeexplore.ieee.org/document/9138896/> → Cited on pages 89 and 236.
- [181] S. H. Houmb, V. N. Franqueira, and E. A. Engum, “Quantifying security risk level from CVSS estimates of frequency and impact,” *Journal of Systems and Software*, vol. 83, no. 9, pp. 1622–1634, Sep. 2010. [Online]. Available: <https://linkinghub.elsevier.com/retrieve/pii/S0164121209002155> → Cited on page 89.

-
- [182] M. Krisper, J. Iber, T. Rauter, and C. Kreiner, “Physical Quantity: Towards a Pattern Language for Quantities and Units in Physical Calculations,” in *Proceedings of the 22nd European Conference on Pattern Languages of Programs*. Irsee Germany: ACM, Jul. 2017, pp. 1–20. [Online]. Available: <https://dl.acm.org/doi/10.1145/3147704.3147715> → Cited on page 96.
- [183] M. Krisper, J. Iber, and J. Dobaj, “Use-Cases for Uncertainty Propagation in Distributed Control Systems,” in *Systems, Software and Services Process Improvement*, X. Larrucea, I. Santamaria, R. V. O’Connor, and R. Messnarz, Eds., vol. 896. Cham: Springer International Publishing, 2018, pp. 368–379, series Title: Communications in Computer and Information Science. [Online]. Available: http://link.springer.com/10.1007/978-3-319-97925-0_30 → Cited on page 118.
- [184] J. Dobaj, C. Schmittner, M. Krisper, and G. Macher, “Towards Integrated Quantitative Security and Safety Risk Assessment,” in *Computer Safety, Reliability, and Security*, A. Romanovsky, E. Troubitsyna, I. Gashi, E. Schoitsch, and F. Bitsch, Eds., vol. 11699. Cham: Springer International Publishing, 2019, pp. 102–116, series Title: Lecture Notes in Computer Science. [Online]. Available: http://link.springer.com/10.1007/978-3-030-26250-1_8 → Cited on page 140.
- [185] M. Krisper, J. Dobaj, and G. Macher, “Patterns for communicating numerical uncertainty,” in *Proceedings of the 24th European Conference on Pattern Languages of Programs - EuroPLop ’19*. Irsee, Germany: ACM Press, 2019, pp. 1–15. [Online]. Available: <http://dl.acm.org/citation.cfm?doid=3361149.3361160> → Cited on page 170.
- [186] M. Krisper, “Problems with Risk Matrices Using Ordinal Scales,” *arXiv:2103.05440 [q-fin]*, Mar. 2021, arXiv: 2103.05440. [Online]. Available: <http://arxiv.org/abs/2103.05440> → Cited on page 208.