Christoph Aigner, BSc

Requirements Specification of a Systems-Engineering Tool: Example on Effort Estimation using Neural Networks

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Information and Computer Engineering

submitted to

Graz University of Technology

Supervisor

Assoc.Prof. Dipl.-Ing. Dr.mont. Franz Pernkopf

Signal Processing and Speech Communication Laboratory

Graz, June 2018

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 master's thesis.

Date	Signature

Acknowledgment

I would like to express my gratitude to Dipl.-Ing. Dirk Denger and Dr. Andreas Braun for giving me the opportunity to conduct my thesis at the battery department and the Systems Engineering Laboratory of AVL List GmbH.
Likewise, I am grateful to Assoc.Prof. Dipl.-Ing. Dr.mont. Franz Pernkopf from the Speech Communication Laboratory at Graz, University of Technology for affording me the realization of my thesis at this company. Thank you for always lending me an ear, providing suggestions and keeping up my motivation.
Thanks to all members of the Global Battery Competence Team under the responsibility of Dr. Volker Hennige who gave me insight in and feedback to the departments proceedings. Moreover, I am particularly grateful to Dipl.-Ing. Paul Schiffbänker, Dr. Wenzel Prochazka and Dr. Andreas Könekamp who helped me to promote my work when staff changes occurred within the department and who helped me to overcome some obstacles.
I am also very thankful for the initial guidance for the thesis structure by Dipl.-Ing. Johannes Fritz, BSc. Furthermore, I want to thank the members of the Systems Engineering Laboratory for all inputs and the participation opportunities in several workshops.
Eventually, I am especially grateful to my family, partner and friends who not only encouraged and supported me during this thesis, but over all the years.

Graz, June 2018
Christoph Aigner

Abstract

Product development in the modern automotive industry has to deal with a vast quantity of knowledge. Additionally, shorter cycle times for development come with the economic competition. An attempt to cope with developments, that are more and more complex, is systems engineering. The idea of systems engineering is to organize all entities which are involved with the product development, i.e. engineering teams of different departments and organizations which e.g. cope with project acquisition, requirements engineering, coordination of teams, etc. Two issues which emerge in this context are tackled within this thesis. Thereby, the battery development department of AVL List GmbH serves as the research environment. First, the volatility of gained knowledge from former product developments has to be handled. There may be many tools to document and store process information, but much information might be lost as not being retrievable and usable in a productivity-improving way. A design for an application that enables engineers to properly work with all kinds of information in the forerun of and during product development is being introduced to support model-based system engineering. This thesis outlines how existing applications may be combined and which benefits can be expected by doing this. Furthermore, functions that support the development processes are stated. Second, many different data items and large data quantities in various relations to each other make it hard for engineers to draw conclusions. For some of the introduced application functions, possible applications of machine learning are mentioned. Regarding a specific function - the estimation of expected development costs from product requirements - a proof of concept is developed by means of an offer assistance prototype application. The prototype application covers the preparation of sparse datasets as well as the usage of several neural network methods to estimate the costs for product development. The results of the cost estimator revealed problems regarding the documentation of product requirements and offers. Nonetheless, the expected costs could be estimated such that engineers are supported at defining budgets for product developments.

Contents

1 Introduction

Systems engineering may be regarded indispensable in the course of creation for many of today's technical systems. The basis of systems engineering is represented by so called systems thinking which, to put it simple, is the organization of system knowledge in subsystems [Haberfellner et al., 2012]. The reason for doing so stems from the large organizational effort of complex systems' development, design, implementation and decommission. The relevant system knowledge not only includes technical aspects, but rather the engineering environment as a whole. This includes on the one hand the coordination of different teams and companies around the world. These are occupied with many tasks in requirements engineering, logistics, reliability engineering, simulation and many more. On the other hand, methods on how to approach certain kinds of engineering processes shall be optimized or newly conceived. Especially concerning development methods, research opportunities emerge.
Regarding these research opportunities, machine learning must be considered. Recently, machine learning pops up in many fields of modern society, may it be to automate processes, to support decision making or the prediction of future data development. There are manifold applications which already employ machine learning. Let us take a look at some examples. The distribution of deliveries by postal services is supported by automated analysis of destination addresses. Thereby, a system was trained to recognize handwritten characters from a digital image. In the medical sector, applications which narrow the chance of illnesses from a number of symptoms and measurements exist. This works by estimating probabilities for illnesses from the information at hand. Also the speech intelligibility in our daily phone calls may be improved by some machine learning methods and even text messages are read to us by an artificial voice whose system was trained how to pronounce syllables in different languages. Formally spoken, the various machine learning methods are generally meant to provide a function f which maps an input $\boldsymbol{x} \in \mathcal{X}$ to an output $\boldsymbol{t} \in \mathcal{T}$. This is done by exploiting patterns which are known, or develop and adapt during the usage of these methods. Also within engineering processes, many tasks consume much time for retrieving, sorting and estimating data and information. This means that engineering environments provide many opportunities for automating processes using machine learning methods.

1.1 Motivation and Problem Statement

The battery development skill team of AVL List GmbH works on the development of customer-tailored solutions for battery system designs. The development of battery system designs thereby must be settled within customer requirements for quality, time, associated cost and ingenuity. As the development of battery systems is a rather new field in the company, many processes, e.g. procurement and exchange of up-to-date information, must still be optimized or introduced anew which must be taken as a huge challenge [Haberfellner et al., 2012]. Additional challenges come up in this engineering environment. The number of competitors in the field of battery development grows with the high demand [Lugger, 2016]. The high demand can be linked to current political goals of many countries regarding reduction of greenhouse gas emissions [United Nations, 2015] as well as them wanting to get more independent from fossil fuels. Furthermore,

the competition intensifies with short time-to-market intervals.

A systems engineering laboratory was installed by the company in order to research and develop methods and tools which support its engineering and management processes. The laboratory is staffed with students from several universities, as it is intended that their new ideas may break with accustomed processes and bring innovation. Considering the battery development teams needs, a cooperation was deemed valuable and the project *MoBat* (model-based systems engineering in battery development) was created. Previously conducted research within this project delivered important steps in the discovery of potential improvements for battery developments. This includes a description and depiction of a generic battery model [Lugger, 2016] and a functional battery model. The most crucial piece of preparatory work regarding this thesis were the findings in a process failure mode and effects analysis (pFMEA) [Schiffbänker, 2016]. These findings give information about problems in battery development that occur in the course of preparation and during the development stage.

1.1.1 Current Shortcomings

A selection of most crucial shortcomings in [Schiffbänker, 2016] lists the following issues:

1. Overall project objectives are not sufficiently defined, coordinated and accessible.
2. Tasks are not formulated sufficiently.
3. Deliverables are not sufficiently specified or tailored to customer needs.
4. Already gained knowledge from previous projects is not being used.

It is certain that these findings interrelate and come with high costs. The retrieval of knowledge from previously conducted research and developments is often not possible or allowed. Also without a clear definition of objectives, the definition of a set of tasks suffers. Consequently, the results of engineering processes in the form of deliverables may differ from customer expectations.

Additionally, the financial assessment of project expenses regarding working hours is named a major deficiency of the project objectives. It must be made in order to do an economic pricing of an offer, but the assessment is time-consuming, as there is no straightforward way for an engineer to calculate it from the project objectives. This is due to quite different objectives given by customers and little experience, because battery system development is a young field of engineering at the company. Also, there were no precise working hours recorded for tasks of previous projects[1], thus being completely in the dark regarding their extent.

1.1.2 Requirements

There are two requirement categories that are dealt with in this thesis. First, this thesis is to present a design for a system which copes with the currently most problematic shortcomings and incorporates model-based systems engineering ideas. This is a theoretical part which mostly concerns ideas for handling the communication and data distribution that is related to development projects. Information from distributed sources shall be organized and kept coherent implying that all stakeholders from project acquisition to project development share the same up-to-date knowledge. This way, the effort of maintaining and distributing data and updates shall be decreased. It is expected that the time savings potentially results in decreased engineering and management costs.

[1] This would track employees doings in a very detailed way and thus is not allowed at AVL List GmbH.

The second and major part of this thesis covers the design and implementation of a machine learning application which makes use of neural networks in order to estimate the expected working hours for a development. This can be converted into costs by what financial risks shall be made assessable. Coincidentally, much working time of an employee, who is committed to compile this estimation manually, may be saved. As the input for the estimation, given as customer requirements, is rather inconsistent, this poses a great challenge.

1.2 Organization and Aim of the Thesis

The major chapters of this thesis feature ideas on a systems engineering application and a prototype application for supporting the financial assessment during the offer creation for battery system developments. In order to get into the matter, a chapter to introduce related backgrounds is given. Eventually, a prospect will show what future research and proceedings should be concerned with.

In order to assist with the comprehension of the key topics of this thesis, Chapter 2 on Background and Related Work gives a short background to systems engineering and methods of learning. This means, first, an introduction into the theory of systems engineering, giving an overview and describing the document-based and the model-based approach. Second, methods of learning are covered for explaining what is meant by learning within the machine learning context and which challenges may occur. Last, a set of neural network architectures is illustrated in detail as these were chosen to be part of the prototype application that is shown in Chapter 5.

The ideas for a *MBSE-System Concept for Battery Development* are presented in Chapter 3, whereby MBSE is the abbreviation for model-based systems engineering. The environment of such an application and targets to fulfill within the application design are given here. The content of this chapter is complemented by more detailed remarks on requirements for the application system in Appendix A.

The subsequent Chapters 4 on *Offer Assistance Prototype - Data* and 5 on *Offer Assistance Prototype - Implementation and Experiments* both relate to the estimator which is built for estimating the working hours regarding a battery development. The prototype is labeled to be an offer assistance, because the offering process at the battery department of AVL shall be supported by it. The former chapter presents the data situation and has statements on how it is dealt with. The latter chapter shows the implementation idea of the application by usage of neural networks including their architecture configurations. Afterwards, the possible results of the actual implementation are depicted and discussed. Additionally, Appendix B has a manual for users and developers of the prototype application.

The thesis closes with a prospect which addresses remarks which on the one hand would help to improve the further development in this topic. On the other hand, future ideas which did not find space within the extent of this work are suggested.

2 Background and Related Work

This chapter shall provide the reader with the basic necessary background for comprehending the contents of this thesis. Therefore, the idea of systems engineering is looked at in Section 2.1. Subsequently, Section 2.2 represents an introduction to the term of learning with regards to machine learning and issues which have to be considered. Finally, Section 2.3 presents the concept of artificial neural networks on the basis of neural networks which are involved in the prototype application which is introduced in Chapter 5.

2.1 Systems Engineering

A system may be described as the entirety of elements which have a relation to each other [Haberfellner et al., 2012]. These elements can also be seen as subsystems. The elements of a system function together to accomplish a need. Regarding engineering in modern industries and the development of a product, i.e. a system, subsystems are assigned to specific disciplines which stand for major parts, services or facilities of the system. Weilkiens states that technical progress leads to complex systems [Weilkiens, 2008]. Additionally, increasing complexity is named as one of the biggest challenge in modern product development projects, which includes that targets relating to time, cost and quality may not always be met [Haberfellner et al., 2012] [Gausemeier et al., 2013].
In order to successfully handle the increasingly complex and interdisciplinary tasks of product developments, systems engineering was introduced. This approach incorporates so called systems thinking, whereby the interactions between the system and its environment are analyzed and the system elements and respective relations characterized. By this, boundaries are set out for the system and its elements, thus leading to a system model. Development of the system elements then can be assigned to specific teams which do not have to care about the system as a whole. The coordination of teams must be taken care of by a system engineer (or team), e.g. functional specifications and requirement specifications or targeted milestones and temporal sequences must be communicated. The following Section 2.1.1 sketches how systems engineering was initially executed on the basis of documents. Then, more important in this thesis, Section 2.1.2 shows what model-based systems engineering is considered to be and what must be thought of when adopting it in the industry.

2.1.1 Document-based Systems Engineering

The Idea

Traditionally, systems engineering is pursued by using textual specifications and documents. These documents are accessible to different stakeholders like customers, developers or testers if necessary. Examples for documents are requirement specifications or functional specifications [Minke, 2015].

Pros and Cons

There are advantages and disadvantages affiliated with this approach. On the one hand, it is convenient that content may be filed in designated repository structures and well-defined documentation schemes lead to a better understanding of project content. Furthermore, the quantity and quality aspects of information help in comparing the project to others. Finally, the centralized storage and version management are not complicated to establish [Minke, 2015].
On the other hand, many disadvantages are tied to the document-based approach. Pieces of information are hidden inside texts and are not quickly retrievable. Often information is spread amongst multiple documents and cross references may be missing because of work intensive maintenance [Alt, 2012]. Consequently, it is hard to trace specific project developments and to keep information coherent regarding changes in requirements. This concerns the information management of a running project and the reuse of information alike [Schiffbänker, 2016]. Furthermore, the documentation quality may suffer because of the intricate way of working with documents [Friedenthal et al., 2012].
It is obvious that the given disadvantages call for a systems engineering approach that is capable of providing a consistent, traceable and reusable documentation. The answer shall be given by model-based systems engineering which is explained in Section 2.1.2

2.1.2 Model-based Systems Engineering (MBSE)

The Idea

As the term model-based systems engineering (MBSE) already hints at, the engineering approach is extended by a model which is extensible and reusable in developments with distinct requirements. The model initially contains ground knowledge on the relationships and links in complex systems and thus helps the engineers to understand the various system elements [Eigner et al., 2014]. Using this support, the assignment of system requirements to system elements and engineering disciplines is facilitated.
The following needs are associated with such a model. An up-to-date graphical and semantic model is to be used [Eigner et al., 2014]. It shall be the point of reference which may be edited and read by all entitled stakeholders of a project leading to a simplified exchange of data. A well-worded definition of the model purpose was given in [Holt and Perry, 2013]:

> An approach to realizing successful systems that is driven by a model that comprises a coherent and consistent set of representations that reflect multiple viewpoints[2] of the system.

It must be noted that it may sound confusing of speaking of a single model to consider. There are different sub-models which serve the development of certain system elements, but it shall be possible to bring their information together if needed. E.g. a functional model of a system including its parameters has been created via SysML[3] and one of its elements is simulated in a Matlab[4] model by adding values to these parameters. From the simulation, an engineer could select the system element via an appropriate application and request all simulation results in a defined range of values or date. Also, the usage of the model does not mean that engineers have to stop using documents, but it constitutes the major source of information [Minke, 2015].

[2] A viewpoint (or view) enables the display of a specific purpose. E.g. it may may be shown which sub-elements a system element consists of or which parameters are in- and outputs of system elements.
[3] Systems Modeling Language (SysML): A graphical modeling language based on UML and XML which was specifically developed for modeling technical systems.
[4] A numerical calculation program for solving and graphically depicting mathematical problems.

Pros and Cons

A list of potential benefits when using MBSE was compiled in [Friedenthal et al., 2012]:

1. Enhanced communications
   - Shared understanding of the system across the development team and other stakeholders
   - Ability to integrate views of the system from multiple perspectives
2. Reduced development risk
   - Ongoing requirements validation and design verification
   - More accurate cost estimation for the system development
3. Improved quality
   - More complete, unambiguous, and verifiable requirements
   - More rigorous traceability between requirements, design, analysis, and testing
   - Enhanced design integrity
4. Increased productivity
   - Faster impact analysis of requirements and design changes
   - More effective exploration of trade-space
   - Reuse of existing models to support design evolution
   - Reduced errors and time during integration and testing
   - Automated document generation
5. Leveraging the models across life cycle
   - Support operator training on the use of the system
   - Support diagnostics and maintenance of the system
6. Enhanced knowledge transfer
   - Capture of existing and legacy designs
   - Efficient access and modification of the information

Nonetheless, there also exist disadvantages concerning MBSE. Initially, models must be created, whereby a large quantity of relationships of model elements has to be set. Sometimes, these mappings are not assignable in a simple way and require further discussion. Then, the transfer and integration of processes, methods and tools into an MBSE environment takes time and additional costs must be considered. This also includes training of engineers to be able to utilize their new development environment.

Adoption Considerations for Organizations

The introduction of MBSE is in need of a clearly set path on how to adopt the idea in an organization. This is necessary as there is cost related to the change which should be reasonable compared to the expected benefits. Moreover, the initiative to bring improvements by redefining work processes and introducing new tools will not be supported one-hundred percent by all concerned stakeholders. The changes must be clearly presented by initiative responsible persons and largely be agreed on by the stakeholders. A five-step improvement process was specified in [Friedenthal et al., 2012] and features the following actions:

1. *Monitoring and Assessing:* Initially, an assessment on currently used SE/MBSE practices should be done in order to identify issues, improvement goals and expected costs. This represents a baseline including aspects of stakeholders' processes, methods and tools. This way, metrics and objectives for an assessment of the introduction of MBSE can be defined and the resulting value by adoption in projects can be measured.

2. *Planning of the improvement:* It is specified how improvement goals are achieved including the improvement process. A detailed plan containing a schedule, responsibilities and resources must be approved by the management. Stakeholders take in responsible personnel of the improvement team, and customers and subcontractors if relevant, for defining and tuning project management, systems engineering, development.

3. *Definition of changes to processes, methods, tools and training:* Changes in the organization are defined, documented and validated. Finally, the affected stakeholders approve these changes.

4. *Piloting the approach:* A pilot project shall be selected and planned. Therefore, an eager team of participants, which is provided with commitment and essential resources by the management, is most suitable. The realization is continuously optimized while running. The pilot project ends with an in-detail evaluation and considerations for possible modifications.

5. *Incremental deployment of changes:* The outcomes of the pilot study indicate requirements for the deployment of MBSE to projects. Examples for these are training methods for engineers and criteria for selecting projects to be supported by MBSE. Observed criteria may be time, maintenance effort or internal support regarding modeling.

Finally, it is important to understand that this improvement process is not finished once all points have been worked on. It is a rather repetitive procedure. When, the first improvement initiative was realized, a new initiative should be started which again starts by monitoring and assessing the new situation.

State of Application in the Industry

Nowadays, many organizations that have ties to the automotive industry have research running on MBSE. Thereby, the largest area of application is in function development, i.e. simulations and CAD tools are used for supporting the development of the various disciplines that are needed for the systems. Often, a SysML model was created initially for breaking down the system into the essential elements and to visualize their relations and functions.
Although, modeling languages like SysML are generally known, they are only used in a couple of departments of organizations [Fritz, 2013]. A number of reasons for this were found in [Zingel, 2013]: It is stated that the generic character of modules obstructs the usage in classical mechanical engineering. Additionally, the information difference of used software tools and generic SysML descriptions is to be blamed. On top of this, SysML diagrams lack an appropriate visualization of management information like status, resources and costs. In [Broy et al., 2010] another factor is named: deficient language specifications limit reasonable modeling.

MBSE and Industry 4.0

Previous statements revealed that MBSE is not embedded to full satisfaction into the development and production changes of automotive industry organizations, yet. Furthermore, the implementation is no easy endeavor. This means there must be a drive which justifies the effort that is invested for pushing the development of MBSE. The answer is to be found in the close

relationship to the *Internet of Things* and data services: Industry 4.0.
There is a large range of consumer products which are highly tailored to customer wishes. Besides, a highly dynamic behavior of demand is given, thus putting production, development and logistic processes to the test. Handling this with a fixed and centrally controlled process structure is hardly possible according to the German federal ministry of education. They state that cyber-physical systems (CPS) could provide a network of interacting elements which may support engineers in becoming agile problem solvers [BMBF, 2017]. The idea of such a network perfectly fits to the goal of MBSE.
The similarities between the concepts of model-based systems engineering and Industry 4.0 were collected in [Alt, 2014]:

- Comprehensive exchange of information between suppliers, customers and producers is the basis of Industry 4.0. The usage of a central model repository like in MBSE would satisfy this need.
- Production changes are used in Industry 4.0. The idea of functional models can be adopted to their specification.
- The production of individual and customized products in a way of mass-production is desired in Industry 4.0. The definition of customer products can be handled like requirements in MBSE, whereby a diversity of variants must be realizable.
- Different models and tools describe products and their states. Industry 4.0 sees a merging of the tools and models, which would be possible by the requirements interchange format (ReqIF) which was defined by the Object Management Group (OMG) [5].

Considering these relations with MBSE, the goal of Industry 4.0 does not seem too far away. Joining currently used methods and technologies is the key for the advent of a new industrial era.

2.2 Methods of Machine Learning

Machine learning relates to the exploitation of more or less complex mathematical models in order to automatically find dependencies and structures in a set of data. These models are often created for data processes which are hard or practically infeasible to be defined by explicit programming. Thus, algorithms which make data-driven predictions and decisions are used. The term *learning* refers to adapting the mathematical models using sample inputs. There are three major categories of learning:

- Supervised learning
- Unsupervised learning
- Reinforcement learning

The first category's principle are in the focus of further explaining as it is relevant in the context of this thesis. In the process of *supervised learning*, dependencies between model inputs and outputs shall be modeled, thus providing means for tackling many practical problems as already mentioned in the introduction of this thesis.
The subsequent sections provide information on *learning* and important topics emerging thereby.

[5] OMG, besides the International Organization for Standardization (ISO), is responsible for standardizing the modeling language UML. UML together with the modeling language XML builds the basis for SysML and as such plays an important role in today's MBSE approaches.

First, necessary technical terms for understanding supervised learning are introduced. Second, various concepts for fitting a polynomial to existing data will be outlined. Third, attention is turned to the issue of *overfitting* which refers to aligning the model too much to given data points. By this, it loses the so called generality, leading to worse results when using completely new data.

2.2.1 Supervised Learning

In order to use supervised learning, a so called training dataset is needed. It contains N data examples $\{(\boldsymbol{x}_1, t_1), \ldots, (\boldsymbol{x}_N, t_N)\}$ whereby the input data is denoted by $\boldsymbol{x} \in \mathcal{X}$ and the output data by $t \in \mathcal{T}$. The output data must not be a scalar like given here, but may also be a vector. For simplicity reasons, the following explanations consider a scalar output. As already mentioned before, the target of *supervised learning* is to produce a *decision function* $f : \mathcal{X} \mapsto \mathcal{T}$, mapping the model inputs $\boldsymbol{x}_n$ to respective model outputs t_n. The training inputs for the machine learning model are often named *feature vectors* $\boldsymbol{x}_n \in \mathbb{R}^d$ with their positions holding the d *features* of the data to be processed for getting an output. The outputs are termed *target values* or also *labels* as learning the model targets for finding rules to achieve these target values. Thus, the training dataset is also called *labeled data*. There are two general kinds of problems that may be approached. In *regression problems*, outputs are real valued and a function which models their distribution is intended to be found. When speaking of *classification problems*, the outputs express *categories* or *classes* and the function f is called *classifier*. The prototype in Chapter 5 will target a model for a function for certain values and figures, which means *regression* will be the main matter in the following.
As already hinted, *learning* or *training* is the process of generating a function f from the training dataset. It is called as such, because it is to improve in performance for every training data example presented to the refining model function. It must be stated that the final model function normally predicts target values from unknown inputs. Thereby, the *generalization* property of the model function is of importance. A formalized expression of generalization assumes that the data used on the model underlies a joint distribution $p(\boldsymbol{x}, t)$. An *error function* (also *loss function*) $e(t, \hat{t})$ measures the quality of the prediction $\hat{t} = f(\boldsymbol{x})$ in comparison to the true target value t. There are different error functions available, but an often used one regarding regression problems is given by the sum-of-squares error function e_{TSS}, i.e. for one sample:

$$e_{TSS}(t, \hat{t}) = (t - \hat{t})^2 \tag{2.1}$$

whereby the sum of the difference is squared, thus delivering a positive error surface. The current explanations only considered the training of models, but more importantly, a model must still perform accurately for data out of the training process. As a matter of fact, the real data distribution of the training set is most of the time unknown. In order to evaluate the generalization performance of a model, a dataset is kept out of training and used on the model afterwards. This means, the available labeled data must be split into a *training set* and a *test set* (also *validation set*). The latter set tests the generalization error of the trained model.
Next, so called *hyperparameters* are needed in many machine learning methods. Their choice may influence the model complexity and the adaption speed and thus are of high importance. Often, these need to be tweaked manually and an optimum may usually not be found. In order to find promising hyperparameters, the model performance with different parameter settings is validated by a special validation set. This special validation set is not to be used in later performance tests as it would falsify the true generalization error of the model.

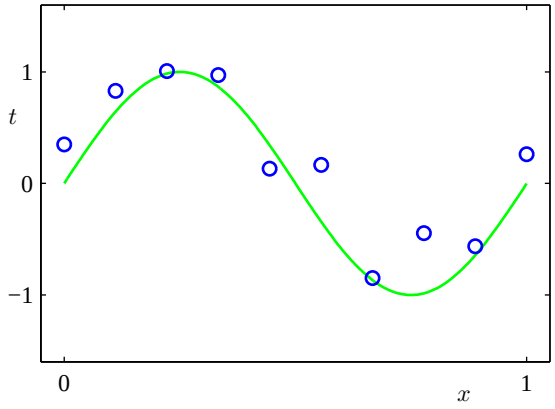


Figure 2.1: This plot shows an exemplary training dataset. The $N = 10$ blue circles refer to an observation of the input x at the respective target variable t. The green sinusoidal was used for generating the data. The task is to predict a value t for another value of x, but the sinusoidal is treated like not known to us [Bishop, 2006].

2.2.2 Fitting Polynomial Curves

We take a look at a regression problem to see how fitting of curves works considering an example like shown in [Bishop, 2006]. The given data is real-valued, whereby a prediction of the target variable t is desired to be made from the input variable x. We assume N sample data pairs $\{(x_1, t_1), \ldots, (x_N, t_N)\}$ from which a function $f(x_n) \approx t_n$ shall be learned for making predictions. The function f is an arbitrary polynomial. Figure 2.1 shows a sinusoidal example. The sample data (blue circles) was generated from a sinusoidal (green) plus added noise as there would be basically no non-noisy observation in real datasets. From the sample data, the model function f must be learned. The sinusoidal or any other polynomial f can be formally written as

$$f_{\boldsymbol{\theta}}(x) = \sum_{k=0}^{K} \theta_k x^k \tag{2.2}$$

whereby $\boldsymbol{\theta} = [\theta_0, \ldots, \theta_K]$ refers to a weight vector. The order of the polynomial is $K + 1$ and $\boldsymbol{\theta}$ holds equal many weights.

An issue, which emerges here is how to set the order of the polynomial. This cannot be answered in a fully satisfying way. It has already been proven that N data points can perfectly fit a $(N - 1)$-order polynomial, so that the error function $e(\boldsymbol{\theta}) = 0$ [Bishop, 2006], but an order which is too large is not of advantage in general. There may be the case that data points may fit perfectly, but at the same time generalization may be very poor. Figure 2.2 shows fitting examples for different polynomial orders. The original curve that is tried to be fitted is denoted green, the fitting attempt is marked red. In Figures 2.2(a) and 2.2(b) with orders $M = 1$ and $M = 2$, underfitting as of low curve adaptability is clearly the case. Figure 2.2(c) shows a quite fitting example with order $M = 3$. Finally, Figure 2.2(d) depicts an excellent fit to the data points, but the found polynomial ($M = 9$) represents the original curve just poorly - this is overfitting. It must additionally be said that overfitting is not only due to polynomial orders, but is related to the size of the training data. If more training samples are involved in the generation of a model function, there is less chance of overfitting as the polynomial is less tuned to the random noise of the target values [Bishop, 2006]. All in all, the polynomial order is a hyperparameter that must be found by testing several orders in a model.

By tuning the parameters of the vector $\boldsymbol{\theta}$, we aim to achieve a function $f(x_n) \approx t_n$ which also generalizes well. In order to make the function f fit the training data, an error function is usually used. The function takes the $\boldsymbol{\theta}$ weights contained in f as inputs and calculates how much of a fit can be achieved. Like already said in Section 2.2.1, the sum-of-squares of errors e_{TSS} could be such an error function:

$$e_{TSS}(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} (f_{\boldsymbol{\theta}}(x_n) - t_n)^2. \tag{2.3}$$

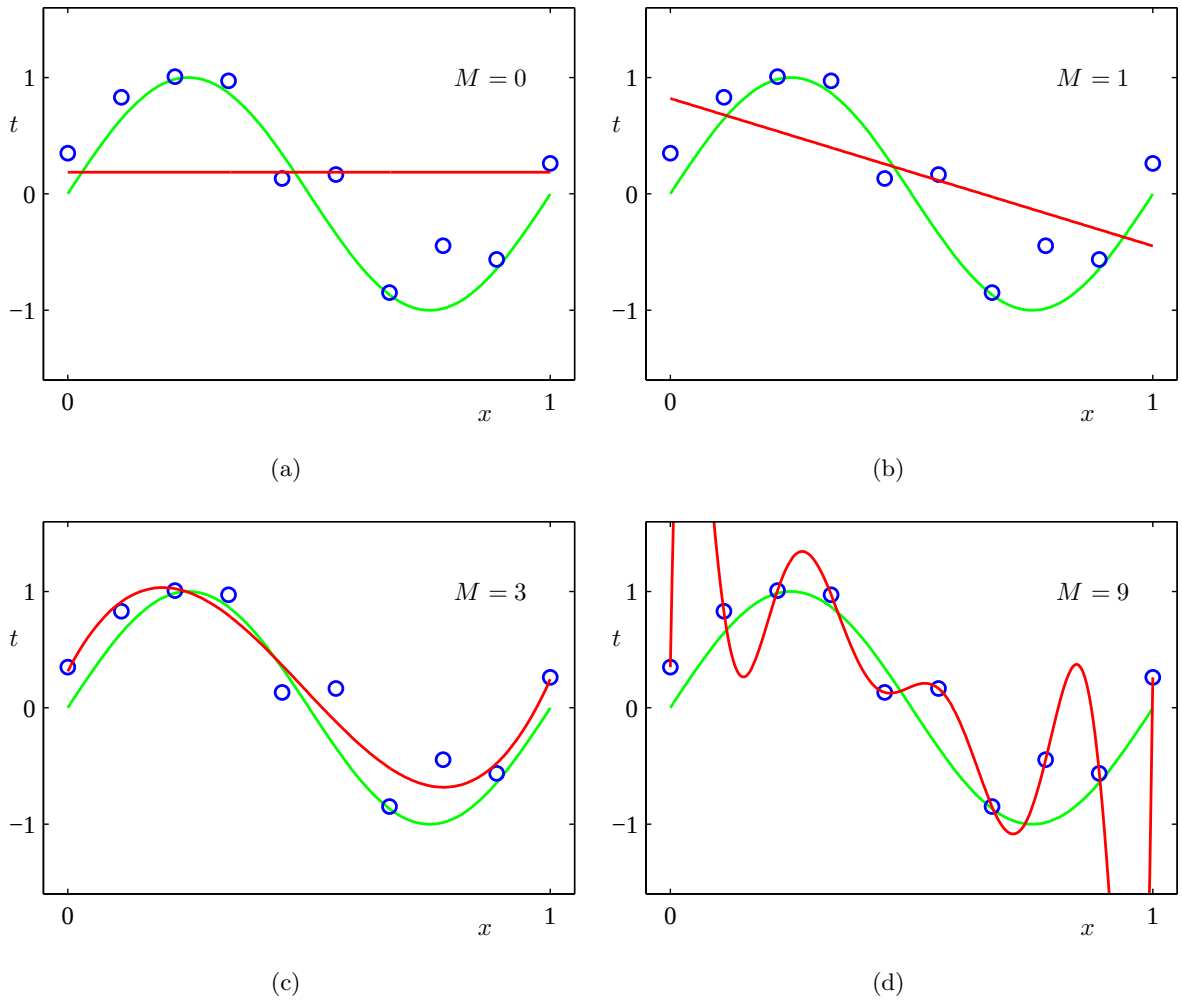


Figure 2.2: Plots of polynomials with orders M (red) which were fitted to the data set given in Figure 2.1 [Bishop, 2006].

This means, finding a function f corresponds to finding parameters that minimize the error function.

2.2.3 Dealing with Overfitting

In general, overfitting is a much worse problem than underfitting, because many stronger deviations from the true data distribution may be achieved. Moreover, a complex regression model in combination with scarce training data is vulnerable to overfitting. A model that includes many parameters is described to be more expressive than one with fewer parameters, i.e. a higher parameter count is also capable of modeling structures which cannot be obtained otherwise.
It is important to understand that not only the order contributes to the complexity of a polynomial. There are also weights of the polynomial that have to be considered. When overfitting, their magnitudes are commonly very large in comparison to non-overfitting polynomials [Bishop, 2006]. This means, it is desired to keep parameter magnitudes low. A way to achieve this is a technique called *regularization*. The idea is to introduce a regularization term $R(f)$ to the used error function which penalizes high parameter magnitudes and by this lowering the model complexity. Again considering the sum-of-squares error, the function can be modified in the

following way:

$$e_{TSS}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}(f_{\boldsymbol{\theta}}(x_n) - t_n)^2 + \lambda R(f_{\boldsymbol{\theta}}) \,, \tag{2.4}$$

whereby λ determines how important the regularization term is in comparison to the original sum-of-squares term. Obviously, the regularization term cancels out by $\lambda = 0$. Contrarily, an infinitely large λ results in the model taking on the zero function. This means, λ is another hyperparameter that must be tuned. In return, this offers a way of coping with high polynomial orders along with rarely available training data, i.e. if λ has a reasonable value, a simple and expressive as possible result is achievable.

A problem which occurs many times is that more than one hyperparameter is used. Whilst the effort for selecting test values and evaluating the performance for one parameter can be kept within reasonable limits, it is difficult for more parameters. The simple reason for this is that the number of possible combinations grows exponentially with each new hyperparameter. The process of testing and evaluating more than one hyperparameter is known as *grid search*. As we see, it is reasonable to be only in need of a few parameters. There are some simple ways to deal with this issue. One way which may sometimes be better than grid search in some applications and models is the *random search* [Bergstra and Bengio, 2012]. Another way would be to initiate the search by using a macro-grid and then to go on searching in a micro-grid which is bounded between promising parameter values. Also possible is the usage of a logarithmic value spacing of hyperparameters, e.g. 2^i where i is an element of a certain integer interval. More sophisticated ways are possible like shown in [Bergstra et al., 2013] which describes that parameter configurations may be determined by observing previously used parameters.

Having looked at hyperparameter search, we continue with the idea of *early stopping* which directly interferes with the training and validation process. Note that this method only is applicable to machine learning methods which use techniques that are intended to iteratively improve their models. In general, the validation error of a model drops for some training iterations until reaching a minimum. Continuing to train the model leads to further adjustment to the training data and the model function starts to overfit. As a result, the validation error increases again. Here it must be said that the overall objective of learning the model function is not the solution to the given optimization problem, but to acquire a regression which has the best average fit to the validation set. This means, the model state at the iteration which provides the lowest validation error must be remembered. The point in time to stop the training of the model cannot be defined straightforward, since the model's validation error does not always drop continuously to a global minimum. Because of this, some decision rules were devised. There are numerous rules and criteria to choose, but only a brief excerpt of possibilities makes sense in the context of this thesis. The easiest of these is to stop training if the validation error has not dropped for a defined number of iterations. Another rule is given by the idea of *generalization loss* which is one of the concepts suggested in [Prechelt, 2012]:

$$g(m) = \frac{e_{val}(m)}{min_{m' \in \{1,\dots,m\}} e_{val}(m')} - 1 \,, \tag{2.5}$$

whereby m denotes the current iteration and $e_{val}(m)$ the corresponding error. The term calculates a value which expressed how much the validation error deteriorated in comparison to the lowest validation error until this point. The training process can be stopped early, if a defined value is exceeded.
The early stopping approaches provide reasonable means for finding a fitting model solution for problems, but there is a final statement about the quality of solutions to be made. Commonly,

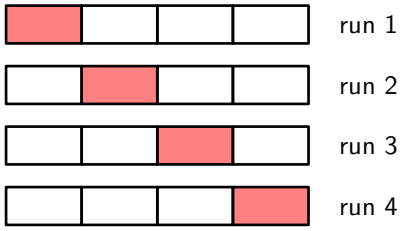


Figure 2.3: For S-fold cross validation, the available data is split into S groups (here $S = 4$, of equal size in the simplest case). $S - 1$ groups are used for training a set of models and evaluated by the remaining group (red). For all S choices, the procedure is repeated. Then, the performance scores from the S runs are averaged. [Bishop, 2006].

hyperparameters which make up a qualitatively fitting model are desired, but by the usage of early stopping, it cannot be always said if the quality of a solution comes from the general model design or the stopping criteria.

Until here, some approaches to deal with overfitting were presented, but all have in common that they had to be conceived because of a lack of available data for training and testing. It is desired to use as large as possible quantities for training, but this comes with the drawback of a noisy estimate of the model's performance [Bishop, 2006]. By using *cross-validation*, a potent technique to train and assess the performance of the model is given [Seni and Elder, 2010]. Its basis is that training and testing capability is not lost by a fixed separation of data into training and test sets. Like the example shown in Figure 2.3, all available data may be used for training and validation. The total data is split into different partitions S of which one is a validation set (red) and all others serve the training purpose. After a training run, another set is chosen for validation and all others for training. By usage of the validation results, an best regularized model can be selected. If there is a really scarce availability of data, as many partitions as there are are data points may be considered - the *leave-one-out* technique.
An obvious disadvantage of cross-validation is the execution of S training runs which can be very time-consuming, especially considering training procedures which already took long before using cross-validation. Additionally, multiple complexity parameters could come up with this kind of technique, meaning many training runs in order to explore settings of their combinations [Bishop, 2006]. With the selection of a best hyperparameter combination, it is common to train a model with the whole dataset again.

2.3 Artificial Neural Networks

For the offer assistance prototype application that is described in Chapter 5, it was defined to make use of neural networks in general. A selection of network architectures was met to be tested for usage: feedforward neural networks, radial basis function (RBF) networks and generalized regression neural networks (GRNN). The choice of network types was met as these represent approaches with different qualities and are rather plain in complexity. The low complexity of models is advantageous with regards to less model parameters that have to be tuned. As the major objective is to investigate the general applicability of some network models to the given offer information datasets[6] that have very sparsely built dependencies, yet, only rough results are needed. The objective of fine tuning network parameters would contradict this goal. The following subsections shall give a brief introduction to the used neural network architectures.

2.3.1 Feedforward Neural Networks

A feedforward neural network consists of nodes that belong to an input layer, a variable number of hidden layers and an output layer. Their node connections do not build cycles or loops [Zell, 1997]. The information in this network is processed in the forward direction from input to

[6] Details on these datasets are given in Chapter 4.

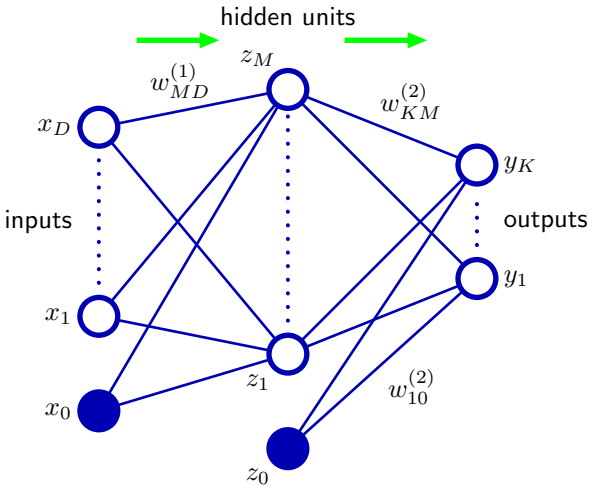


Figure 2.4: General structure of a feedforward neural network. The input, hidden and output variables are given as nodes. The weights are denoted by links in between the nodes. The bias parameters are represented by linkage from variables x_0 and z_0. The arrows denote the direction of the information flow at forward propagation [Bishop, 2006]. The structure of the network could also feature more than one hidden layer.

output nodes through the hidden ones. Figure 2.4 gives an overview of the network structure. The reason for taking this kind of network is its rather simple and basic neural network structure which may be quickly implemented. As of one strict direction of propagation, computation time may be relatively fast. If there are appropriate results, a simple approach could be favored over a more sophisticated one.

Principle

The network's function can be seen as a series of functional transformations [Bishop, 2006]. For this purpose, M linear combinations of the input variables x_i, $i = 1, \ldots, D$, are built:

$$a_j = \sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \tag{2.6}$$

where $j = 1, \ldots M$, and the exponent (1) denotes the parameter affiliation to the network's first layer. The w_{ji} parameters are generally referred to as *weights* and the w_{j0} parameters as *biases*. The constructed values a_j are known as *activation* which are transformed via differentiable, nonlinear *activation functions* $h(\cdot)$,

$$z_j = h(a_j)\,, \tag{2.7}$$

to obtain the *hidden unit* values [Bishop, 2006]. These values again are used to build linear combinations to obtain the output unit activations

$$a_k = \sum_{j=1}^{M} w_{kj}^{(2)} z_j + w_{k0}^{(2)} \tag{2.8}$$

with $k = 1, \ldots, K$, where K is the number of outputs. Equally like in Equation 2.6, the w parameters are called weights and biases. The output unit activations also are transformed via activation functions which leads to the network outputs y_k [Bishop, 2006]:

$$y_k = h(a_k)\,. \tag{2.9}$$

The activation functions may be chosen from a range of generally nonlinear functions, because they enable the network for computation of non-trivial problems. A non-trivial problem is also present in the experiments of Chapter 5 as the influences from information inputs not always add linearly to the solution, but scale according to complex information relations. The

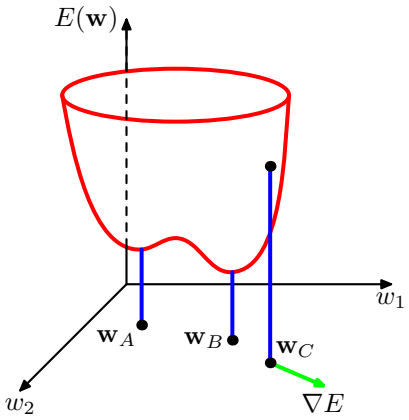


Figure 2.5: Geometrical exemplary view of an error function $E(\boldsymbol{w})$. The error is a surface over a 2-dimensional weight space. A local minimum is given in point $\boldsymbol{w}_A$ and a global minimum is given in point $\boldsymbol{w}_B$. At the arbitrary point $\boldsymbol{w}_C$, the local gradient of the error surface is given by the vector ∇E. The gradient vector points towards the steepest slope of the error function [Bishop, 2006].

nonlinear activation functions are often represented by sigmoid functions which are normally used for multiple binary classification problems. A softmax activation function is generally used for multiclass problems [Bishop, 2006]. Exemplarily considering the network in Figure 2.4 with a logistic sigmoid activation function $\sigma(\cdot)$ for the output layer, the nonlinear function for calculating the network outputs y_k from a quantity of inputs may be built:

$$y_k(\boldsymbol{x}, \boldsymbol{w}) = \sigma\Big(\sum_{j=1}^{M} w_{kj}^{(2)} h\Big(\sum_{i=1}^{D} w_{ji}^{(1)} x_i + w_{j0}^{(1)} \Big) + w_{k0}^{(2)} \Big) \tag{2.10}$$

where $\boldsymbol{x}$ is a vector containing all input values and $\boldsymbol{w}$ is a vector containing all weights and biases [Bishop, 2006].

Training

In order to determine and adjust the network parameters, training of the network is required. This includes the adjustment of weights and biases of the network. There are many training methods for neural networks of which the *backpropagation* is used here. This method is often used in feedforward networks and known for accuracy and computational efficiency [Bishop, 2006]. There is extensive literature available which considers the derivation and background to this approach, but only the procedure and a rough overview on its background is necessary in the scope of this work.
The reason for training is the parameter optimization, i.e. finding weights which minimize a chosen error function $E(\boldsymbol{w})$. This is no trivial task, because of the amount of possible weight permutations. Figure 2.5 depicts an exemplary geometrical picture of such an error function. At the smallest values of the error function, i.e. local and global minima, the gradient of the error function is zero. There is no chance of finding an analytical solution to the equation $\nabla E(\boldsymbol{w})$, so an iterative numerical procedure like backpropagation is necessary [Bishop, 2006].
The classical backpropagation procedure makes use of the gradient descent algorithm in order to reach a minimum of the error function. In this work, *stochastic gradient descent* is used, because it features the possibility of escaping local minima [Bishop, 2006] and yet leaves a rather simple implementation for this prototype application. It was proposed in [LeCun et al., 1989] as an on-line version of gradient descent. The idea is that the error function based on maximum likelihood for a set of independent observations is made up of a sum of terms for each input pattern n:

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w}) \, . \tag{2.11}$$

For this approach, updates to the weights are made for each data point respectively by

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{(\tau)} - \eta \nabla E_n(\boldsymbol{w}^{(\tau)}) \tag{2.12}$$

where $\eta > 0$ indicates a learning rate and τ labels the iteration step. This means for every step and data point, an error minimum is approached by making a weight adaption in the direction of the negative gradient [Bishop, 2006]. The computation of the gradients needs an error function, which is chosen to be a sum-of-squares error (as it gives a positive error surface):

$$E_n = \frac{1}{2} \sum_k (y_k(\boldsymbol{x}_n, \boldsymbol{w}) - t_{n,k})^2 \tag{2.13}$$

where the subscript n refers to the pattern and the subscript k the output node. With this knowledge, the backpropagation procedure could be derived as covered by ([LeCun, 1988] [Bishop, 2006]). The relevant application steps for each backpropagation iteration may be summarized in the following way regarding a feedforward network in Figure 2.4:

1. An input vector $\boldsymbol{x}$ is forward propagated through the network, i.e. through hidden units to output vector y_k.

2. The errors δ_k are calculated for all output nodes, where

   $$\delta_k = y_k - t_k \tag{2.14}$$

   with t_k being the target values for the output nodes.

3. The errors δ_k are propagated backwards using

   $$\delta_j = h'(a_j) \sum_k w_{kj} \delta_k \tag{2.15}$$

   in order to get δ_j for each hidden unit of the network.

4. The derivatives used for adjusting the network weights are evaluated by

   $$\frac{\partial E_n}{\partial w_{ji}^{(1)}} = \delta_j z_i \tag{2.16}$$

   respectively

   $$\frac{\partial E_n}{\partial w_{kj}^{(2)}} = \delta_k z_j \ . \tag{2.17}$$

The algorithm furthermore is in need of initial weight values. There are a couple of common methods available which target almost uniform weight distributions per layer. The choice of initialization method is of minor relevance in the scope of this work, as its influence would only be minor in comparison to the influence of sparsely available datasets that are introduced in Chapter 4.
In order to achieve a network model which generalizes well, an *early stopping* method will be used in the experiments given in Chapter 5.

2.3.2 Radial Basis Function Networks

Radial basis function networks make use of radial basis functions (RBFs) as activation functions. The network outputs are a linear combination of RBFs of input and node parameters. Figure 2.6

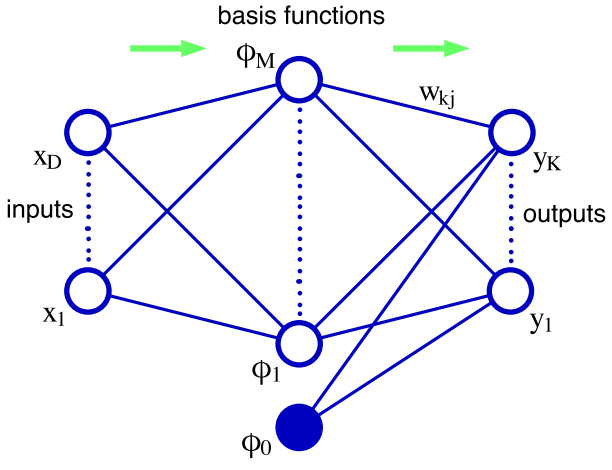


Figure 2.6: General structure of a radial basis function network network, corresponding to Equation 2.19 [Bishop, 1995]. The arrows denote the direction of the information flow from input to output. The basis functions work like the hidden units in a feedforward network, but without receiving weighted information from the input. Instead, the distance between the input vector $\boldsymbol{x}$ and the basis function center vector $\boldsymbol{\mu}_j$ are evaluated on the basis function ϕ_j. The weights w_{kj} are given as connections from the basis functions to the outputs. The biases are denoted by weights stemming from an additional basis function with a fixed output of 1.

shows the layout of such a network. Again, the rather straight forward design was a reason for choosing this network method as well as the facts that the RBF network should generally offer good generalization and strong tolerance to input noise [Yu et al., 2011]. The latter aspect may play an important role for the application in this work's experiments regarding the available datasets (see Chapter 4).

Principle

A general summary of the RBF neuron function would be that each neuron represents a so called prototype vector. Input vectors to the neuron are compared with the prototype vector and the neuron produces a result between 0 and 1 as a measure of similarity. If input and prototype vector are equal, 1 will be the result. Increasing distance between these two leads to the value converging 0 as the RBF neuron response is a bell curve. The prototype vector may be selected from the training sets, but may also be computed randomly. It is also called the neuron's center, as it is the value at the center of the basis function.
The radial basis function idea for the network's RBF neurons is that they model a function $f(\boldsymbol{x})$ which precisely fits the target values such that $f(\boldsymbol{x}_k) = t_k$ for $k = 1, \ldots, K$ patterns [Powell, 1987]. The function expression for these requirements is given as a linear combination of radial basis functions

$$f(\boldsymbol{x}) = \sum_{k=1}^{K} w_k h(||\boldsymbol{x} - \boldsymbol{x}_k||) \tag{2.18}$$

where w_k is the weighting of the RBF result to the output neuron with every radial basis function centered on a data point [Broomhead and Lowe, 1988] [Bishop, 2006].
As of generalization problems, some modifications have to be made to Equation 2.18 in order to get rid of the noise contained in the data which influences the results [Bishop, 1995]. This means, the number of basis functions shall be related to the mapping complexity which is desired to be represented. Until now, their number was related to the size of the dataset as the number of patterns in the data had to be equal to the RBF number. The modification requirements are [Bishop, 1995]:

- The number M of basis functions must not be equal to the number K of data points.
- The centers of basis functions may no longer only be chosen from input vectors. They shall be defined during training.
- Each basis function shall obtain an own width parameter, which is defined during training.

- The linear sum receives bias parameters. These adjust the offset between the average over the basis function dataset and related average of targets.

Thus, the radial basis function expression for neural networks is

$$y_k(\boldsymbol{x}) = \sum_{j=1}^{M} w_{kj}\phi_j(\boldsymbol{x}) + w_{k0} \tag{2.19}$$

where j indicates the basis functions and k the outputs. Respective weights connect basis functions and outputs, and w_{k0} denotes the bias to the outputs. x is the d-dimensional input vector. A common and popular kind of radial basis function is the Gaussian, here

$$\phi_j(\boldsymbol{x}) = \frac{1}{\sigma\sqrt{2\pi}} \mathrm{e}^{\left(-\frac{||\boldsymbol{x}-\boldsymbol{\mu}_j||^2}{2\sigma_j^2}\right)} \tag{2.20}$$

where $\boldsymbol{\mu}_j$ denotes the center vector of the basis function ϕ_j. σ is the variance of the Gaussian and also called *smoothing parameter* in this context. The basis function can be further stripped by removing the factor in front of the exponential, because it can be fully controlled by the weights w_{kj}.

Training

Training of the network parameters generally concerns two major parts. First, the radial basis function width and the positioning of centers must be determined. Second, the weights from radial basis functions to output neurons must be optimized. For regression problems, there were numerous algorithms presented for finding these parameters, but often rely on data characteristics [Huan et al., 2011]. The definition of optimal centers is an issue. With regards to the little available data and as only a general possibility for machine learning shall be shown in this prototypical approach, experiments with various numbers of RBF neurons will be performed, including randomized centers and widths of the Gaussians. The second part of training, concerning the weights which connect RBF neurons and outputs, is described in the following.

Considering the basis function parameters fixed, the second stage of parameter training can be started, connecting weights to the outputs. First, the bias may be absorbed into the RBF neuron layer by creating an additional basis function $\phi_0 = 1$:

$$y_k(\boldsymbol{x}) = \sum_{j=0}^{M} w_{kj}\phi_j(\boldsymbol{x})\,. \tag{2.21}$$

The matrix notation for this is

$$\boldsymbol{y}(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{\phi} \tag{2.22}$$

where $\boldsymbol{W} = [w_{kj}]$ and $\boldsymbol{\phi} = [\phi_j]$ [Bishop, 1995]. The weights now may be optimized by using the information given from an error function, e.g. the sum-of-squares error like stated in previous network method. This error function may be denoted here as

$$E = \frac{1}{2}\sum_n \sum_k [y_k(\boldsymbol{x}_n) - t_{n,k}]^2 \tag{2.23}$$

where $\boldsymbol{x}_n$ is the input vector to the network and $t_{n,k}$ is the related target value of output node k. $n = [1, \ldots, N]$ indicates the training sample.

The quadratic weight-dependent error function lets us find a minimum by solving a set of linear equations [Bishop, 1995]:

$$\boldsymbol{\Phi}^T\boldsymbol{\Phi}\boldsymbol{W}^T = \boldsymbol{\Phi}^T\boldsymbol{T} \tag{2.24}$$

where $\boldsymbol{T}_{nk} = [t_{nk}]$ and $\boldsymbol{\Phi}_{nj} = [\phi_j(\boldsymbol{x}_n)]$. The solution to $\boldsymbol{W}_kj$ is practically obtained by singular value decomposition:

$$\boldsymbol{W}^T = (\boldsymbol{\Phi}^T\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^T\boldsymbol{T}\,. \tag{2.25}$$

This kind of weight definition may be done, because of the linear combination of RBF neuron outputs. Nonetheless, also other training methods like the backpropagation may be used, although they may not always reach the error optimum (in case of nonlinear combinations) or may take many iteration steps.

2.3.3 Generalized Regression Neural Networks

As a third neural network method, a generalized regression neural network (GRNN) was selected. Unlike the previously introduced networks, there are no weights to be initialized at all, with only a Gaussian spread parameter to be adjusted. There are some advantages related to a GRNN shown in [Specht, 1991]. Learning is done by one pass through the training data. Furthermore, convergence to the conditional mean regression surfaces with more training data being available, but reasonable regression surfaces already build from few training samples. Also, the convergence to local minima of the error surface is not possible.

Principle

The generalized neural network structure contains four layers: the input, the pattern, the summation and the output layer [Specht, 1991]. Figure 2.7 shows this structure. Like in the previously introduced RBF neural network, an input vector is processed by RBF neurons, which are called pattern units in this scope. For this approach, the number of basis functions is kept equal to the number of data points. This follows from the subsequently stated explanations.
A Gaussian, like given in Equation 2.20, again serves as an activation function for the pattern units. These units deliver the activation value vector. This means, the input vector is subtracted from the stored vector representing each pattern. Then the squares of the differences are summed and fed to the nonlinear activation function. These values are distributed to the so called summation units. [Specht, 1991]
There are two kinds of summation units. One is the denominator unit, all others are numerator units. Every single numerator value is divided by the denominator in order to get output unit values, i.e. there must be $K + 1$ summation units with K as the number of output units. The summation function $f(\boldsymbol{x})$ for the denominator estimate is a simple summation of activation values from the pattern units:

$$f(\boldsymbol{x}) = \sum_{j=1}^{M} \phi_j(\boldsymbol{x}) \tag{2.26}$$

where j indicates the originating pattern unit. The numerator unit estimates are calculated by multiplying the activation values coming from the pattern units which were obtained in the training process with the respective training target values t_j:

$$\boldsymbol{t}f(\boldsymbol{x}) = \sum_{j=1}^{M} t_j \cdot \phi_j(\boldsymbol{x}) \tag{2.27}$$

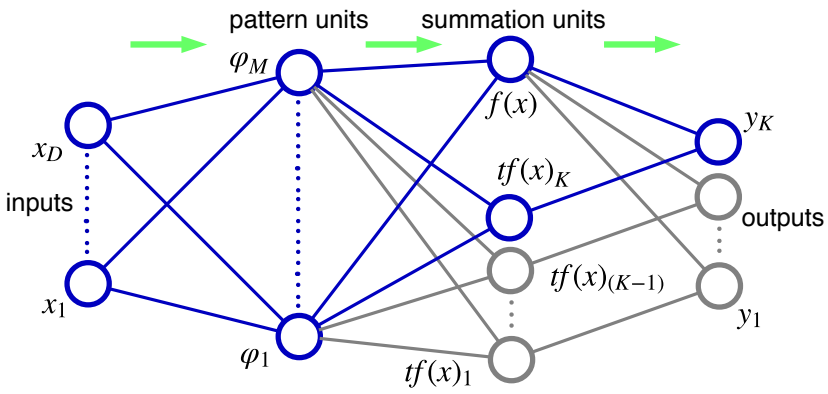


Figure 2.7: Structure of a generalized neural network. The arrows indicate the direction of information flow from input to output nodes. The connections drawn in blue describe the propagation from the input to one specific output node for reasons of clarity and comprehensibility, but are equally done for all outputs. The pattern units are the RBF units which calculate an activation value a_j, $j = 1, \ldots, M$ from the distance between input vector and pattern centers. The activation values are fed to the summation units, where a denominator value is calculated by $f(\boldsymbol{x}) = a_1 + a_2 + \cdots + a_M$ and numerator values are calculated by $\boldsymbol{t}f(\boldsymbol{x}) = t_1 a_1 + t_2 a_2 + \cdots + t_M a_M$, where t_j denotes the training target value of a pattern unit from a specific training sample. The division of a numerator and the denominator in the respective output unit yields the estimate of y_k [Specht, 1991].

where j is again the pattern unit index. Thus, the summation unit performs a dot product between the target value vector $\boldsymbol{t}$ and the activation value vector $\boldsymbol{\phi}$. The target values t_j are mapped to the specific pattern units as the idea of this network is that every training sample incorporates a mean for a new pattern unit to the network. The overall mathematical representation of an output node value y_k thus may be denoted as:

$$y_k(\boldsymbol{x}) = \frac{\sum_{j=1}^{M} t_j \cdot \phi_j(\boldsymbol{x})}{\sum_{j=1}^{M} \phi_j(\boldsymbol{x})} \tag{2.28}$$

where k denotes the corresponding output unit. This network method originated from the requirement of dismissing the necessity of assuming a specific functional form. Thus, the appropriate form shall be expressed as a probability density function (pdf) which is to be determined empirically. From the joint pdf of an input vector $\boldsymbol{x}$ and a target value y, the expected value may be calculated [Specht, 1991].

Training

A training of parameters like mentioned in previously introduced network methods of this chapter is basically not needed. The network parameters are determined directly from training samples (single-pass learning) and generalization of the structure is immediately given [Specht, 1991]. This means that the training samples (it would be better to call them construction samples) are taken as center values of the pattern units and their respective target values feed the $\boldsymbol{t}$ for numerator calculations.
The only freely adjustable parameter of this method is the variance σ of the Gaussians at the pattern units. For keeping the instant learning capability of the network, one σ is used for all pattern units [Specht, 1991]. The *holdout method*, like explained in [Specht, 1991], is used to find the best value of σ by testing different values of σ with different training samples.

3 MBSE-System Concept for Battery Development

This chapter introduces the general concept for a system of software applications. The system shall realize the practice of model-based systems engineering (MBSE) in the battery development environment and is referred to as *MoBat-System* henceforward. The presented capabilities of the system stem directly from the needs of future system users. The feasibility of each system capability was investigated beforehand, i.e. only suggestions which were regarded to be realizable are mentioned for the MoBat-System concept.
The initially presented system scope indicates the major purposes of the MoBat-System. These purposes are derived from the shortcomings in current battery development which are given in Section 1.1.1. The next section about the system environment gives a short high-level overview on the infrastructure and persons which are involved with the MoBat-System. Afterwards, a section about the system context provides information about the major structure and relationships of the MoBat-System. Thereby, the context is divided into internal and external context. The internal context relates to the capabilities which shall be provided by the MoBat-System and the external context concerns the system's linkage to existing software applications and databases. The final section of this chapter briefly introduces the user characteristics which lead to the requirements which must be thought of for the MoBat-System. Detailed remarks on the requirements which are mainly important to readers who have to deal with the system realization will be given in Appendix A.

3.1 System Scope

The findings of most crucial shortcomings in current battery development (Section 1.1.1) lead to certain system objectives. These are complemented by objectives which must be generally considered for such a system in order to be usable. All objectives together aim for a unified system to control and modify the process of battery development and also the related project acquisition process. These two processes go hand in hand with each other. The battery development process involves the boundaries and objectives which are collected during the project acquisition process by which first development-related data is acquired and the project extent is defined. The list of objectives to be tackled by the MoBat-System reads as follows:

- *Tracking and definition options for project data*
  The system shall provide means to create and edit objectives, requirements, tasks, functions and to define the relations amongst these elements as well as amongst their subelements.

- *Project scope support*
  The dependencies between objectives, requirements and tasks shall be exploited in order to suggest appropriate project tasks. The recommendation relies on the wishes expressed by the customer. Thus, a better assessment of workload shall be enabled.

- *Battery module data exploitation*
  Functions and components and their dependencies are the core elements of battery developments. Thus, this information must be extracted from the existing generic battery model[7] for usage in the MoBat-System. Furthermore, simulation parameters for a project are not trivial to obtain. A suggestion for their values by estimation from previous project simulation parameters shall be enabled.

- *Access rights management*
  Shares of data shall only be accessible to certain users of certain stakeholders which are involved in the processes of battery development and project acquisition. Means of authentication shall ensure an access control.

- *Coherent and unambiguous data management*
  Data on several databases and in several software applications shall communicate with the MoBat-System. The MoBat-System must ensure a coherent and unambiguous data usage.

- *Project acquisition support*
  Product managers must be able to operate with a mobile device only while talking about a project with a customer. A mobile application which supports the product manager in this situation shall be part of the MoBat-System. The application capabilities shall comprise support for quick mail writing by intelligent suggestions on recipients and further mail contents as well as for requesting and sending information from and to the server-sided MoBat-System.

3.2 System Environment

The MoBat-System is placed in an environment of various accessing actors and different software and hardware elements as well as has to be conform to demanded development techniques. Figure 3.1 depicts this high-level environment overview.
First of all, the MoBat-System must integrate the idea of model-based systems engineering and is part of the project *model-based systems engineering in battery development* (MoBat). It shall consist of software elements which are in need of an infrastructure in order to fulfill the demands on the system. This on the one hand covers software tools and databases which exchange information with the MoBat-System. On the other hand, access to the internet is a necessity as the MoBat-System is intended to distribute and receive data amongst various servers, work stations and mobile devices around the globe.
The right-hand side of Figure 3.1 is made up of three parties. The first party considers the *service engineers* of the MoBat-System. These are responsible for the development and the maintenance of the system according to the demands of the remaining two parties parties. The second party is called *production* and consists of persons who are responsible for the assembly of product developments by using the information which was created by the developers in a project. This means, they only have to read the information which was produced with the usage of the MoBat-System. The two described parties are excluded from the so called *user* party, because only this last group is intended for using the major capabilities of the MoBat-System like tuning product parameters. The system shall basically be designed to support the users during product development. Thereby, *product engineers* like product managers, project planning managers or development engineers, access the MoBat-System and provide it with project-relevant or generally relevant information for acquiring a project and developing the

[7] This model was created in the PTC Integrity Modeler and includes all known functions and components that may be involved in a battery system.

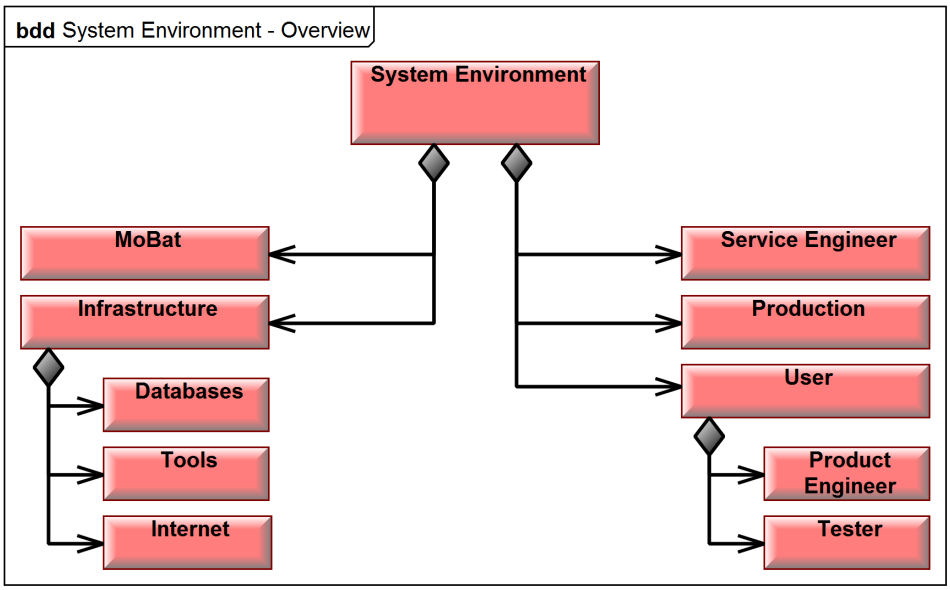


Figure 3.1: Embedding environment of the MoBat-System.

involved product. Eventually, *testers* are working with the MoBat-System. The testers own the same roles as the product engineers, but are mainly associated with using new additions to the MoBat-System in a special testing sub-system which shall run in parallel to the productive one.

3.3 System Context

The MoBat-System context covers the major structure of the MoBat-System as well as its relationships to elements outside of the system. This means, the external context concerns the system's embedding into the so called AVL tool-network (also Cluu tool-network). Thus, the MoBat-System shall be connected to different existing software applications, databases and other server-side file storage for bidirectionally transferring data. This also includes the MoBat DB which is part of the MoBat project and already has a basic design, but must be largely extended to hold the data described in Section A.1.1. The internal context gives an high-level overview on the functions that must be handled by the MoBat-System. Figure 3.2 shows the high-level structure of the system.
The MoBat-System consists of two major parts. The *core system* shall contain the major operations which the system shall be capable of. The second part is the so called *mobile system* and shall be realized as a mobile device application which will be used by some of the system users. The mobile system shall only include some of the functions that must be provided by the core system. The specification of the functional requirements to both sub-systems is given in Section A.1.
Input to the system shall be delivered in three ways. The first is a user interface which is intended to be realized as a web service. Second, input shall come from the mobile system. Third, input is received from the partly configured *Cluu Tool-Network*. This tool network provides company-wide means for systems engineering and is realized by the *Cluu* software application. Existing database and tool data is connected via a *Cluu* configuration (in-house name *data.Connect*). This software must be configured such that a data altering process in the network is performed consistently in all connected elements. The interfacing for the MoBat-System must be provided bidirectionally for being able to retrieve and modify system data.

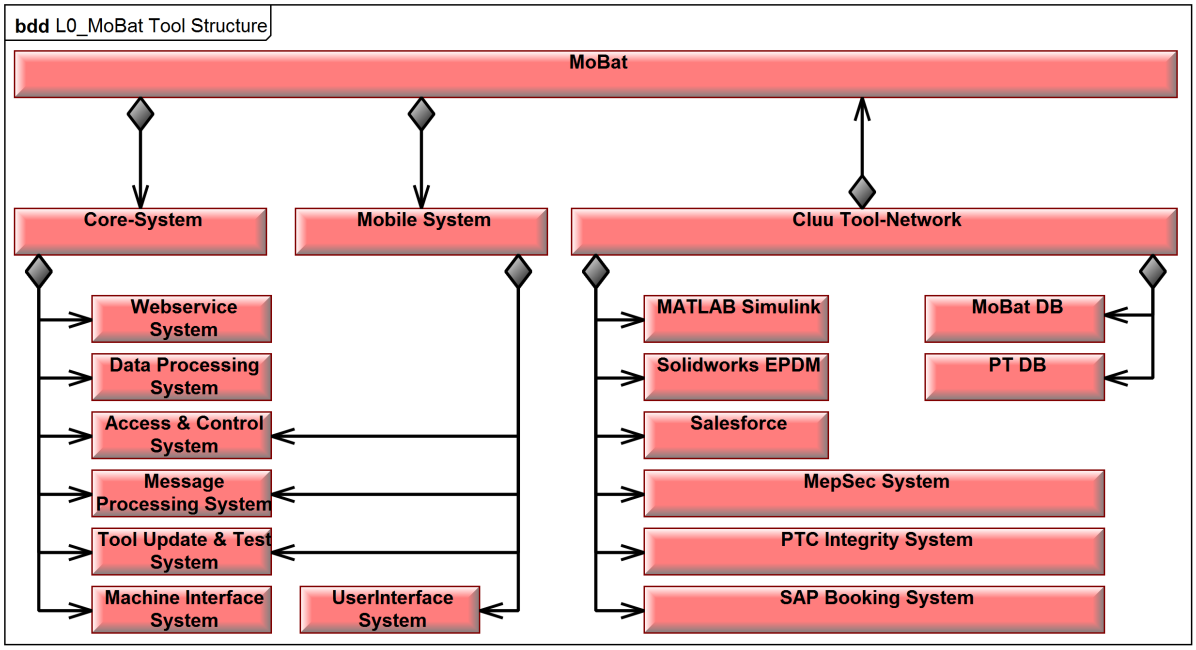


Figure 3.2: High-level structure of the MoBat-System (MoBat Box). The system includes a core-system and several mobile systems which are in need of depicted major functionality. The MoBat-System as a whole is part of a tool-network which is maintained with the software Cluu. Thus, the linkage two diverse databases and software systems shall be handled.

3.3.1 Internal System Context

Two applications provide means of usage for the MoBat-System. The more important one is the core-system as it shall include more functionality than the mobile system. As Figure 3.2 already showed, there are overlaps between the core and the mobile system. The mobile system will feature only some functions of the core-system, because it will almost only be used by a product manager in a meeting with a customer. Figure 3.3 gives an overview of the internal structure of the core-system. The sub-systems are briefly introduced subsequently. More detailed requirements for these sub-systems will be presented in Appendix A.

- Web Service System: The core-system must be accessible to the users via a web service, mainly due to the fact that this enables an architecture which is independent of used platforms, coding languages and protocols [Cavanaugh, 2006] and as the open standard prevents licensing costs. Additionally, web services can communicate with other applications that use different web services. The usage of protocols of the HTTP family may be exemplary suggested as of the just stated reasons. Thereby, it is crucial to implement a proper encryption and authentication for the web service. Furthermore, web services for communicating with databases and tools must be established for exchanging data with the AVL tool-network.

- *Data Processing System*: This system shall be relevant for processing all kinds of functional operations that are driven by battery productive data and not concerning mail services (message processing) or interfacing (provided by two other subsystems). This means, calculations and estimations of product parameters and process-relevant information shall be possible.

- *Message Processing System*: This system basically describes a mail client with enhanced functionality. The user is supported in quickly distributing information by mail.

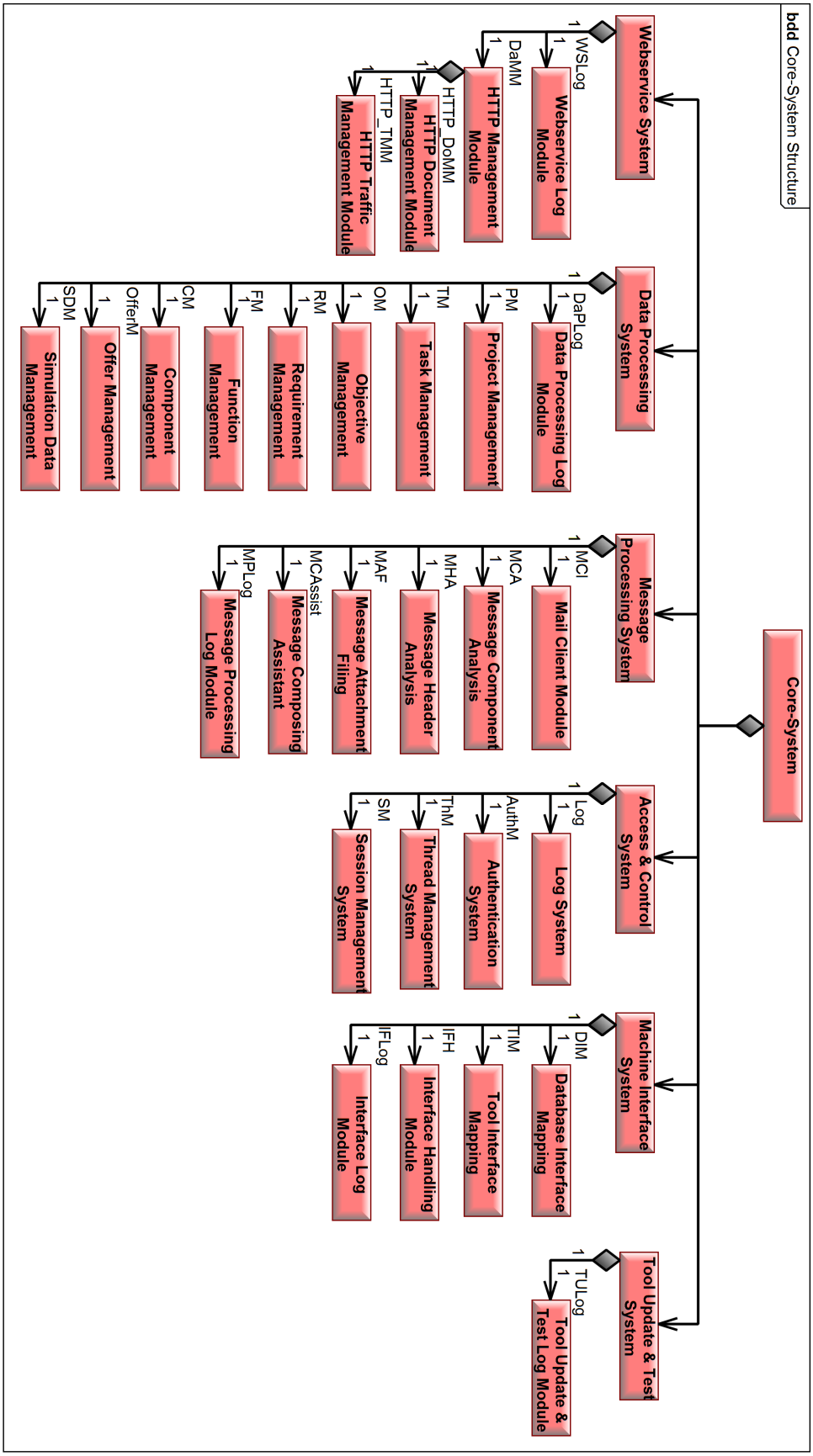


Figure 3.3: High-level structure of the MoBat core-system.

- *Access & Control System*: The system is assigned the tasks to handle sessions for its users and their authentication. It additionally has to provide thread management in order of the system being functional for a large number of sessions and data processing.
- *Machine Interface System*: The sub-system's purpose is to structure data in a way that it may be used with the tool-network as well as to decompose it for usage in the MoBat-System. This may include preparation in order to receive and send data to tool-external systems and databases. A handling module shall be used thereby to keep track of interfacing operations and to signal potential problems regarding these data exchange operations.
- *Tool Update & Test System*: A software tool is usually developed iteratively. The sub-system is to provide means to update the MoBat-System and test these updates without interfering with the productive system, but is using necessary sets of realistic productive data samples.

All sub-systems and their respective requirements are to be equipped with logging functions in order to ensure operations are traceable, e.g. probable failures or unauthorized access shall be noted.
The mobile system shall also use the functionality of an access & control system, a message processing system and a tool update & test system, but adjusted to fewer requirements in comparison to the core-system. Nonetheless, the functionality shall include the same sub-functionality like shown in Figure 3.3. The user interface for the mobile application is not in need of a web service as it is intended to exchange information with the core-system by mail protocols.

3.3.2 External System Context

The MoBat-System shall be integrated into the AVL tool-network like shown in the structure in Figure 3.2. The tool-network must be enabled to handle data which comes from the following and existing MoBat-relevant sources:

- *MoBat DB*: This database originates from previous MoBat proceedings and as such may be modified solely for project purposes. Its content must be modified for holding data which is not considered, yet. Section A.1.1 provides the necessary data items.
- *Powertrain DB*: This DB holds data which relates to the parameters of different kinds of powertrains, e.g. traction battery, fuel cell and combustion engines. This means value settings for different function and component attributes of different projects can be extracted and stored using this database.
- *MATLAB Simulink*: With this software tool, a functional battery model was built considering varying parameter values for different projects. As this is only a tool with no database in the background, a way to import and export data to and from it has to be considered.
- *MepSec System*: The system is used for managing access rights of AVL employees for various software within AVL. Access information is stored by an associated database.
- *PTC Integrity System*: From the *PTC Integrity* product group, the *Lifecycle Manager* and the *Modeler* are used. Information which is generated with these software tools is stored in a single database. The Lifecycle Manager, like Salesforce, is used for managing requirements of a project as well as related tasks. The Modeler was used for the already mentioned generic battery model.
- *Salesforce*: Customer information and associated projects within AVL reside in this system. New project data is created from the moment the AVL product manager is consulted by

the customer. Project data refers to objectives and requirements for a new project. The information is stored in a database.

- *SAP Booking System*: Within this context, the booking system issues the working time that AVL employees spend on projects and deliverables as well as costs of material and further purchases for the product development. Working time is booked via the AVL-wide established electronic time sheet application *ESZ*. Furthermore, figures on the real cost of deliverables are assigned and analyzed for projects via the AVL-wide established application *proCalc*. Also, reports on the financial progress and outcome of projects are compiled here. Information used by these applications is stored to the company's SAP databases.

- *SOLIDWORKS Enterprise PDM*: This data management solution stores CAD models of all components that may be used for the battery development as well as battery constructions. Also bills of material (BOM) are extracted from here. The information is stored in a database.

Furthermore, software tools which cannot be dealt with by the Cluu-Tool-Network in a reasonable way must be considered for the MoBat-System:

- *Battery Wiki*: The wiki website is used as a tool for common and gained knowledge as well as *lessons learned* regarding projects.

- *Microsoft Sharepoint*: Project reports and offers are stored in Microsoft Sharepoint directories.

Eventually, from all these sources, information shall be collectable by the MoBat-System in order to do calculations, e.g. computations on parameters or compilation of reports. The cases where the listed items are used, are given in Appendix A.

3.4 User Characteristics and Requirements

Several user roles, with partly overlapping usage characteristics, are developing, operating and using the system. These are divided in the following user groups:

- System Maintenance: System Service Engineer, Server Administrator, Support Staff, etc.

- Product Management: Product Manager, Project Planning Manager

- Product Development: Development Engineer, System Engineer, Lead Engineer, etc.

- Production: Production Engineer, Technical Staff, etc.

Their usage characteristics must be known in order to design the MoBat-System to fit their needs. This means the necessary functional requirements are obtained from their demands to the system. The following sections introduce the use cases which must be considered for the system users. These are important for defining much of the requirements that the MoBat-System has to fulfill. The requirements specifications is given in Appendix A, as this chapter is only intended to provide a major understanding of the concept. In order to fully understand the use cases, Section A.1.1 includes a collection of functional system objects and their description.

3.4.1 System Maintenance Use Cases

The system maintenance group is responsible for maintaining the system and to add new features to the MoBat-System. The group consists with people who are responsible for the system development as well as system administrators who are required to deploy the MoBat-System on AVL servers. Figure 3.4 gives insight in the groups activities. The group is charged with the

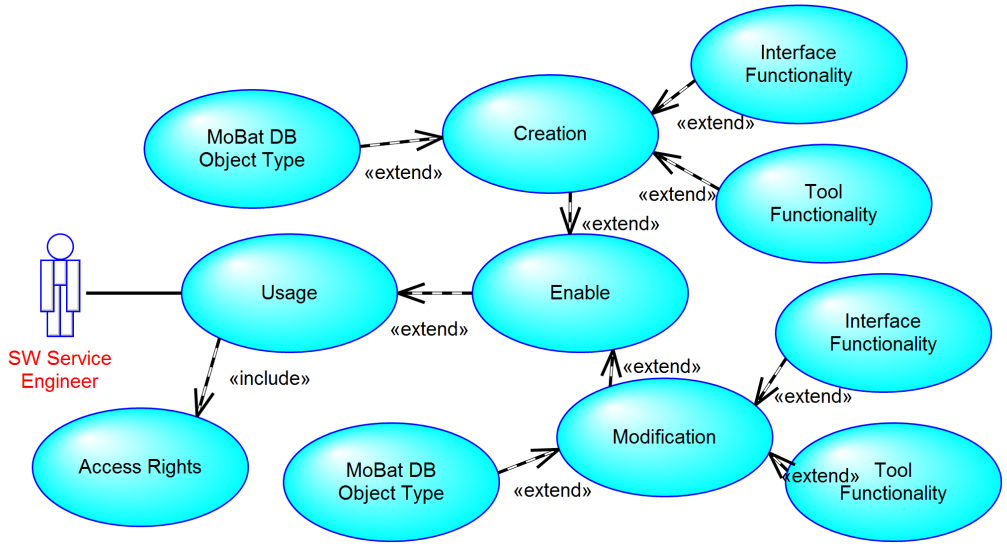


Figure 3.4: Overview on use cases of the system maintenance.

following competences:

- The group implements new tool functionality and provides appropriate interfacing if needed. Additionally, the enhancement of the MoBat DB regarding new major system objects is realized.
- The modification of tool functionality, interfaces and the MoBat DB must be enabled.
- In order to enable users to properly work with the MoBat-System, user support must be provided.
- The access to the activities is bound to appropriate access rights.

As explained beforehand, changes must be first implemented in a test system, which potentially also AVL-external personal must be able to access. The access to the productive system must be reserved to AVL employees.

3.4.2 Product Management Use Cases

The product management group is occupied with the project acquisition and management. Both activities are highly dependent on customer objectives, i.e. wishes must be investigated by direct communication with the customer. Especially, needed change requests and different ideas for development targets must be resolved to the customer's satisfaction. The group also serves the coordination of project contents with the development group and must also consult them for discussions on the feasibility of the project. The MoBat-Application must provide support for the actions that are shown in Figure 3.5.
The use cases view, create and modify own different specializations. The use cases apply to negotiations and requirement specifications with the customer, but also to interaction with the battery development group's area of responsibility.

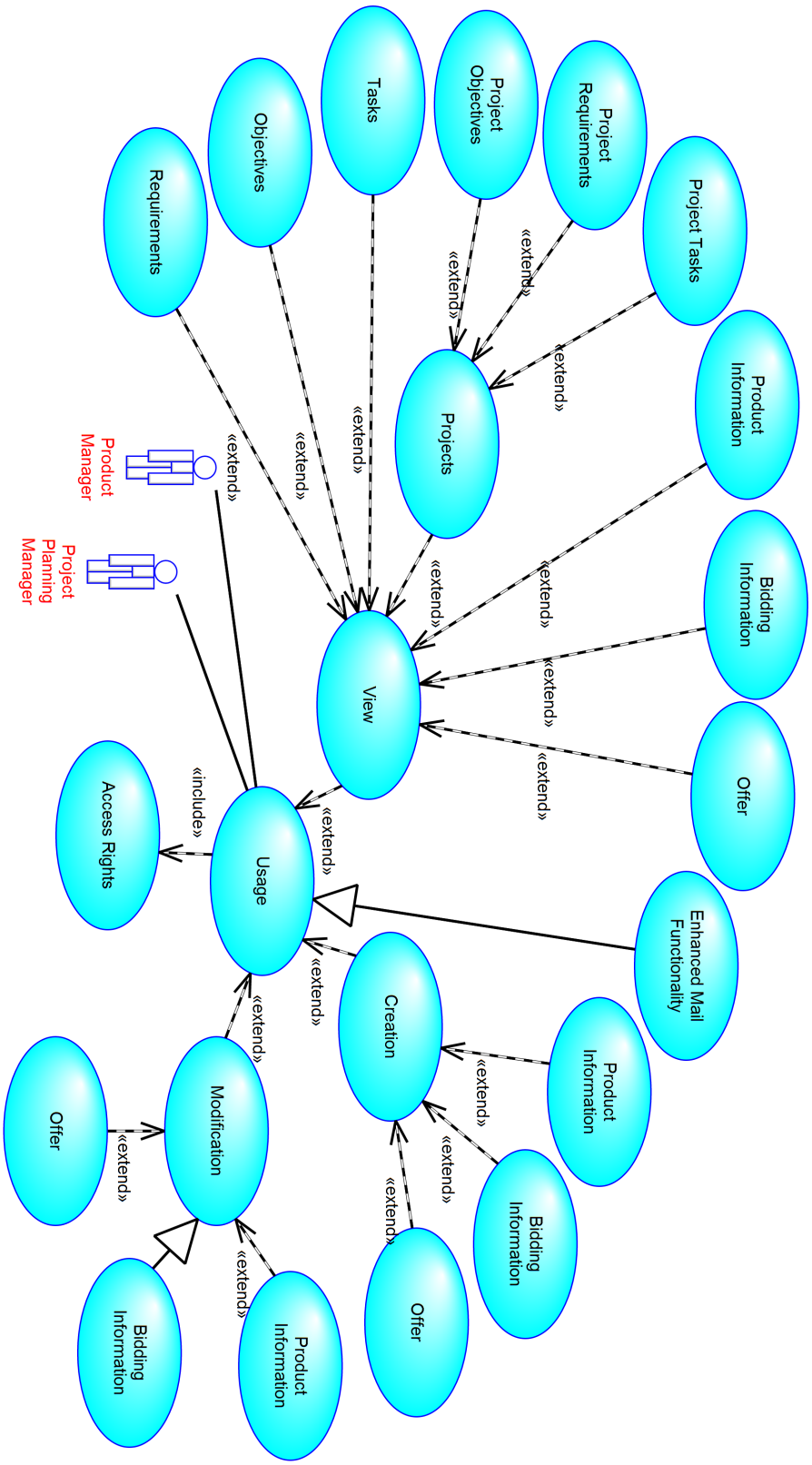


Figure 3.5: Overview on use cases of the product management.

- Persons of the product management group must be able to see which objectives, requirements, tasks, product information, bidding information and offering templates are generally available for appropriately advising customers on the know-how of the company.

Furthermore, this is important regarding the feasibility of customer wishes and thus for knowing how to deal the customer.

- The status of objectives, requirements, tasks and deliverables of a certain project must be accessible. This helps to understand the progress of a project and to inform the customer about the proceedings.
- The enhanced mail functionality (especially mobile app) shall be usable. This supports the purposeful communication with the right persons.
- Product information and bidding information must be created and managed. This refers to the range of products in general, but also related to customers and running projects, i.e. an overview on AVL's products can be gained.
- General project information shall be accessible, i.e. responsible persons and used functions and components.
- The access to the activities is bound to appropriate access rights.

3.4.3 Product Development Use Cases

The product development group members accomplish the development of a product according to the customer wishes. Thereby, project information on former projects must be taken into account. The development group may also be consulted by the product management group for giving an estimation about the feasibility of a project as well as the potentially occurring costs. The product development group represents the major user group of the systems engineering application and thus is also the major group for maintaining development information. Figure 3.6 holds an overview about the product development group's use cases. The product development group may view, create and modify almost every information item that is related to product development if all access rights are owned:

- The management of objectives, requirements, tasks, functions and components must be enabled. This includes the creation of template objects as well as the management and modification in a specific project. A key use case herein is the is the linkage of dependencies between the system objects and the filling of attributes and properties.
- Components and simulation parameters are often coming from different software tools and users must be enabled to import this data to the system.
- Tracking information must be accessible, i.e. what is the state of accomplishment of certain tasks and what is left to do for a deliverable. Also a change protocol of the objectives, requirements, tasks, functions and components as well as simulation parameters must be considered for this use case.
- The ability to edit and viewing access according to certain projects and also of templates is always bound to respective access rights.

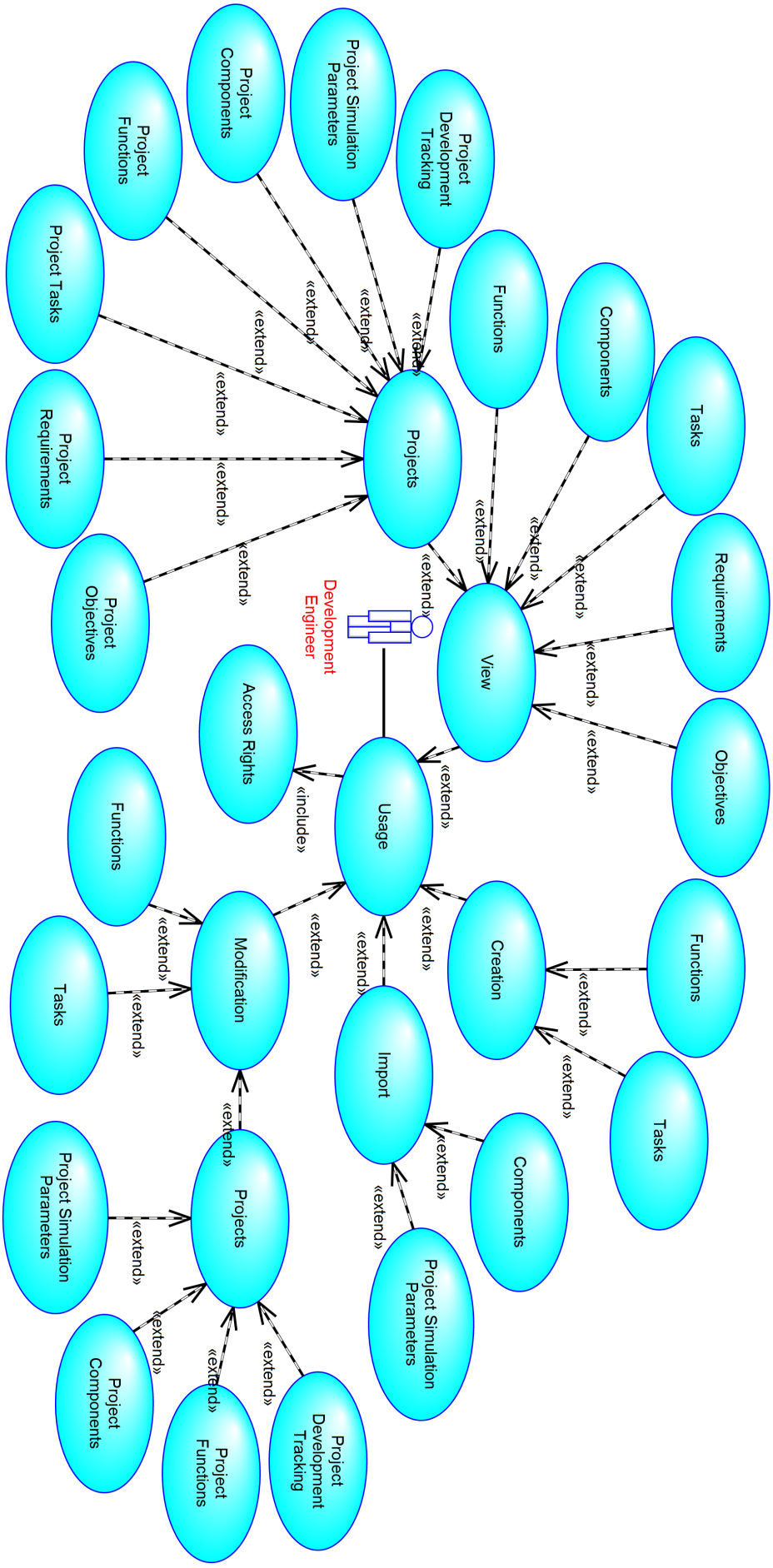


Figure 3.6: Overview on use cases of the product development.

3.4.4 Production Use Cases

The production group engineers realize the product designs specified by the development group. For a lossless transfer of product parameters, specific project data must be accessible, e.g. the components to assemble and their parameter settings. The range of functionality of the MoBat-System is minor here as the assigned tasks generally concern the construction of samples and prototypes. The investigation of performance is again conducted by the development group. For the sake of completeness, Figure 3.7 shows the production group's use cases.

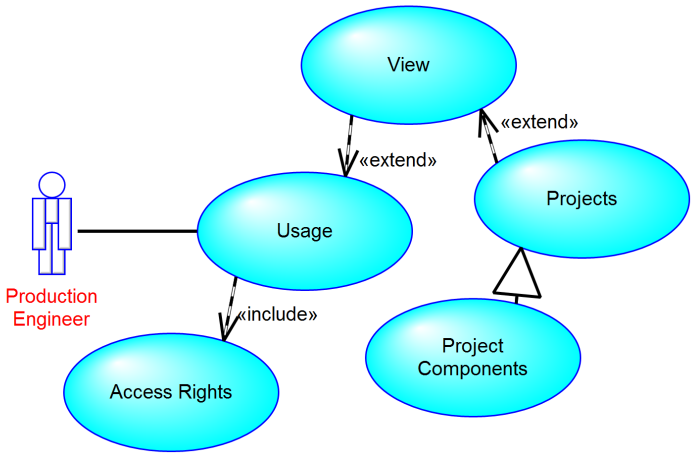


Figure 3.7: Overview on use cases of the production.

4 Offer Assistance Prototype - Data

As already mentioned in Section 1.1.1, there are shortcomings in estimating the workload and related costs only from the customer objectives for a product. A reason for this is that customers request rather individual solutions, i.e. few development constraints which must be satisfied and many degrees of freedom for the remaining development scope. Apart from that, many countries for which the battery system developments are designed, have a different legislation regarding vehicle standards. This leads to varying effort for developments. In order to help with the assessment of the development and cost effort for a product, the need of a software for assisting the cost estimation emerged.
The requirements for such a prototype were not defined in advance to this thesis, as the definition of data and access to it only emerged in the course of this thesis. However, only the intention of providing support by means of machine learning, more precisely by neural networks, was communicated. Since a solution for the effort estimation demands data not available from the start of this work, its compilation and preparation took much time in the progress of this thesis. This chapter summarizes the approach for the preparation of the relevant data. The data is then used for supervised learning in the offer assistance prototype in Chapter 5.
Section 4.1 gives an overview about the difficulties in the compilation of the data series. This is followed by a brief description on the finally gathered data in Section 4.2. The way how the data in these sources is prepared is shown in Section 4.3 and finally considerations about the data normalization are mentioned in Section 4.4.

4.1 Objectives and Related Issues

Although the objective of designing and implementing a prototype application which would estimate development and cost effort was introduced as a goal, its accomplishment was hardly achievable due to lack of data. This lack of data on the one hand originates from incomplete datasets. On the other hand, restrictions inside the company limited the access to desired datasets. Thus, the following paragraphs provide a short summary on the progression of finding data to work with.
The battery development team was established few years prior to this work. Therefore, there were only few projects accomplished at a certain maturity level [8]. For the research in this thesis, the maturity levels A-sample and B-sample were considered, because for these levels the largest group of projects was available.
The documents on selected projects provide customer objectives, tasks, deliverables and pricing. Unfortunately, the information was given in individually compiled documents. Furthermore, a standardized form and naming of the content for the considered information items was missing, leading to a largely manual search through these records.
The identification of relationships between a project's tasks and the assessed working hours was an objective of the offer assistance prototype. Ideally, a breakdown of costs per task and

[8] The maturity level of a development describes the placement of a product on a scale from A (concept) to E (series production). This scaling describes the in-house standard at AVL, but industrial partners and contestants use similar descriptions for the progress of their products.

involved departments could have been estimated for new projects. Unfortunately, the access to relevant files was denied. Additionally, investigations showed that tasks with similar names in different projects often had been assigned quite varying kinds of sub-tasks. A partial solution to this issue was found with the access to the project bookings to cost centers, which are not as detailed, but provide the true expenses which emerge in each project.
All in all, only a small quantity of only 6 datasets could be obtained. Each dataset is a set of files which were compiled in the course of a project that comprises a single battery development. This low quantity already implies that less expressive results can be expected. Nonetheless, orientation may be provided for comparison with the engineer's estimation of workload. This way, a rough assessment on the quality of the model can be achieved.
Finally, the access to data and in consequence the information given in the data was only granted few weeks before the thesis deadline. As such, the focus was laid on simple methods for preparing and working with the data. Since there were only a few datasets available, only the general applicability of machine learning was expected to be demonstrated. Regarding this, the application of simple methods was a reasonable focus.

4.2 Data Sources

There were two kinds of documents made available for the extraction of data that is needed for the offer assistance prototype.
First, offer information on accomplished projects is available in *PDF* files or *Word* files. In general, paragraphs on four different contents are found in the inspected offers:

- *Introduction*: Here, the objectives for the desired product are described.
- *Scope of work*: Tasks to conduct the development of the product in a project are given.
- *Deliverables*: Packages of (partial) solutions and intermediate reports on the product development's progress are defined.
- *Price offer*: A rough breakdown of prices according to an itemization as requested by the customer is compiled. The prices do not necessarily have to correlate with the expenses for a project.

Unfortunately, the information is mostly hidden in text passages and files provide more or less information depending on the objectives. Furthermore, the files were not compiled from a template and therefore miss a common structure or labeling of paragraphs and tables. Also, terms regarding the scope of work, deliverables and pricing as well as their contents do differ. Finally, the offers were written in English or German language. All these aspects make it impossible to extract the provided information in a simple automated way, because it would need much effort in language processing and as such is out of the scope of this work.

Second, the bookings to cost centers per project could be provided by SAP excerpts in form of *Excel* files. These files contain various information, but thereof, two items are important:

1. *Cost Type*: The cost type describes the principal purpose of a booking, e.g. development hours or consumption of applicable material.
2. *Value in Euros*: There are many cost center bookings to be considered of which every booking has a certain monetary value.

The usage of cost centers are equal over all projects. Thus, this data may be extracted in an automated way.

In the experiments in Chapter 5, *Basics*, *Scope of Work* and *Deliverables* will be the inputs to several neural network models. The final outputs of the models will then be the *Prices* of offers as well as the overall bookings to *Cost Centers* in a project. The form of these data items is presented in Section 4.3.

4.3 Data Preparation

In order to make a neural network learn from the provided project documents, some of their information must be adapted to fit to the inputs and outputs of the network models. To provide the information in such a form, a categorization approach must be applied to the offer information data. Therefore, the four categories *Basics*, *Scope of Work*, *Deliverables* and *Prices* were defined according to the contents which were described in Section 4.2. Expressions in German language were translated to English. The processes of sorting the data are depicted in Section 4.3.1. Furthermore, information on the bookings to cost centers is given in an appropriate form for automated processing already and thus needs only little preparatory work.

4.3.1 Categorization of Offer Information

Basics

This category collects information on the major objectives of battery developments. These objectives occur in most offers. The selection of objectives that are most important in an offering was discussed and finalized in a group of development engineers and product managers from the battery development skill team. The collection of objectives gathered is listed in Table 4.1.
The offer title is parsed for certain terms. These terms indicate different quality levels of battery developments. Furthermore, the data on customer, offer title, vehicle concept, project duration and application level was rated as especially valuable. The procedures to deal with missing data are presented in Section 4.3.3.

Scope of Work

The scope of work (SoW) of a project describes the work packages that are to be done. There is no explicit naming convention for work packages, thus the naming of work packages may differ a lot. Furthermore, the content of similar work packages may not be consistent, leading to a lower or higher number of various sub-items. In order to obtain a reasonable measure for the work packages, a scoring system was introduced.
The scoring system is intended to weight major categories from task naming. An example for a category would be the term *simulation* which exemplary covers the terms *FEM-simulation*, *CAE-simulation* or *simulation*. Another, less apparent example than the previous one would be the term *project management* covering the terms *supplier meeting*, *kick-off meeting* and *project management*. For each item related to a certain category, the score of the category increases by 1. The following categories were chosen from the available offers:

- Cell
- Electrical Engineering
- Mechanical Engineering
- Procurement
- Production
- Project Management
- Prototyping
- Quality Management
- Simulation
- System Engineering
- Testing

Input Identifier	Selection	Description
Customer	single	The customer of the project.
Offer title	multiple	The title of the project offer contains several keywords that are parsed: concept, outline, draft, design, development, layout, simulation, sample build, detail, prototype and test. The parsing process is not case-sensitive.
Vehicle concept	single	One of the concepts electrical vehicle (EV), plug-in hybrid electrical vehicle (PHEV) or hybrid electrical vehicle (HEV) may be chosen.
Project duration	single	The project duration in days.
Placing option	single	Indication if the project has the option for a follow-up project or not.
Application level	multiple	Describes the level(s) of application which shall be covered: cell, module or pack[9].
Number of variants	multiple	The number of variants desired for each application level.
Cell type	single	The cell type may be a pouch cell, a cylindrical cell or a prismatic cell.
Planned production volume	single	Units of the top application level that are likely to be produced.
Energy capacity	single	Capacity of the top application level in kWh.
Min. voltage output	single	Minimum voltage output of the top application level in V.
Max. voltage output	single	Maximum voltage output of the top application level in V.
Max. power output	single	Maximum power output of the top application level in kW. This value is measured at $25°C$ for $10s$ and at $50\%SoC$[10].
Avg. power output	single	Average power output of the top application level in kW. This value is measured at $25°C$ for $30s$ and at $50\%SoC$.

Table 4.1: The basics data refers to the objectives which the offer assistance prototype uses as starting point in the estimation of expected costs for a project. The inputs are either denoted by a single choice or by a multiple choice of figures or words. If there is no knowledge about a data feature, nothing is specified and the missing value is handled like shown in Section 4.3.3.

[9] Packs consist of a number of modules and modules of clusters of cells. These stages of application have different development demands, e.g. for electrochemical stability, junction, spacing.
[10] SoC: state of charge of the battery.

Deliverables

The deliverables of a project are the outcomes in a project that are shared with the customer. Like in the scope of work category, the naming of deliverables and the content differ a lot amongst different offerings, but cannot be assigned to a value which corresponds to the effort. Due to this fact, the same scoring system that was described for the work of scope items was applied, leading to the following categories:

- Cell
- Electrical Engineering
- Mechanical Engineering
- Procurement
- Production
- Project Management
- Prototyping
- Quality Management
- Simulation
- System Engineering
- Testing

Prices

Again, the data situation forced the simplification of data in the form of a scoring system. Categories were defined to cover terms, e.g. the term *testing* covers the terms *tests* or *thermal characterization testing*. The terms are related to prices. These prices are accumulated per category. The categories listed below were determined as recurring ones:

- Engineering
- Procurement
- Production
- Prototyping
- Simulation
- Testing

4.3.2 Preparation of Cost Center Data

Since the information on project bookings to cost centers is given in a standardized way, the preparation may be done with a straightforward software approach. The cost centers of provided files are scanned. Then, the monetary values of all bookings to the cost centers are summed up per project. The calculations are stored in *CSV* files whereby each line refers to a different project and columns indicate the cost centers.

4.3.3 Data Completion and Encoding

Some of the data introduced above must be prepared additionally for the applicability with neural networks, in order to be proper inputs and outputs for learning the networks. The offer data from scope of work, deliverables and pricing was already simplified by introducing the categories using certain integer or decimal values per category. The same applies to the cost center data. The challenge is to complete and encode some of the basics data in Table 4.1.
Furthermore, missing data is problematic, as every data item will be represented by at least one neuron (see Chapter 5). According to [Bishop, 1995], the simplest solution would be to discard data samples assuming a large quantity of data samples and a small share of affected patterns in the dataset. Due to the small amount of data, this is not applicable. Another approach would suggest to drop features with a large proportion of unavailable values from our model input. This is ruled out as there is a chance that important factors for our model would be excluded which leads to bias and introduces errors [Horton and Kleinman, 2007]. Other methods like seeking a maximum likelihood solution using expectation-maximization by processing the missing values via integration over corresponding features [Ahmad and Tresp, 1992] might be possible.

The following data items have their missing data handled by making assumptions on the data:

- *Customer*: Different customers are involved per project. In order to get a hold of the customer input, a one-hot encoding is used. Thus, the customer information is a binary value vector, where each entry stands for a certain customer. The input which corresponds to the project customer is set to 1, the other ones are set to 0.
  The customers of projects are assigned these input positions as they must be considered to have different expectation and maturity levels. An example therefore would be that customer A generally wants to develop an electrical vehicle for a certain country with regards to a certain level of safety standards and usage characteristics. Customer B wants to develop similar vehicles as customer A, but normally operates on a larger market with different kinds of regulations. This leads to a certain development maturity which in general has to be considered in the model via the customer.

- *Offer title*: The project title may contain certain keywords. The title keywords of a project are represented by a one-hot vector. The related project title keywords are denoted by value 1, all others are 0.

- *Vehicle concept*: Several concepts of vehicles are in need of different battery dimensioning and functionality. From the application-featured suggestions - EV, PHEV, HEV - one must be chosen. These inputs are added to the neural network model as a one-hot vector, where the entry related to the vehicle model is 1 and the other ones are 0.

- *Project duration*: The project duration was assumed to correlate with the overall costs of a project. Thus, the durations were taken from projects in which they were specified and their mean average hours determined as inputs for projects which did not include this information.

- *Placing option*: An integer scalar which is 0 if no placing option was specified and 1 otherwise.

- *Application level*: If no level was specified, a battery pack[11] is assumed for development.

- *Number of variants*: If no number of variants was specified, a single variant is assumed for development. The number of variants is an integer value.

- *Cell type*: There are three major cell types used. The selection of a specific cell type is stored as a one-hot vector with respective values being set to 1, while others are 0. Unfortunately, the specification of cell type is not always given. Here, simple statistical values were applied, i.e. the probabilities of the cell types being used in projects where a cell type was selected are taken as a reasonable input for the neural networks. For instance, if the prismatic cell is used in 40% of projects, the corresponding neuron is assigned the value 0.4.

Finally, there is a group of data features where no reasonable assumption can be made. There are *Planned production volume*, *energy capacity*, *min. voltage output*, *max. voltage output*, *max. power output* and *avg. power output*. Only a rather general option of filling the missing values can be applied for them. Thus, a mean average is calculated over all data samples of a feature which provides a value for the respective missing data features. This mean average is used for the missing values. Although problems may come up with this approach, it is still better than the usage of extrema [Bishop, 1995].

[11] A battery pack generally consists of a single or several connected modules. The modules themselves are made of interconnected battery cells.

4.4 Normalization

The normalization of the data for the neural networks is important as the data values partly differ in several orders of magnitude. The normalization has the effect that their relative importance for network output calculations are assumed even at the start of the network training [Bishop, 1995].
The importance of normalization is especially given for the radial basis function networks which are used in the experiments. The activation of a basis function is dependent on the Euclidean distance l between the input vector $\mathbf{x}$ and the basis function center $\boldsymbol{\mu}_j$ in a D-dimensional input space:

$$l = ||\mathbf{x} - \boldsymbol{\mu}_j||^2 = \sum_{i=1}^{D} (x_i - \mu_{j,i})^2 \ , \tag{4.1}$$

The distance l may be non-sensitive to a feature which has a much smaller range of values than others [Bishop, 1995].
Normalization between 0 and 1 was applied as this was fitting to the provided kind of data. The calculation of the normalized value x' from a non-normalized value x is denoted as:

$$x' = n_l + \frac{(x - d_l)(n_h - n_l)}{d_h - d_l} \ , \tag{4.2}$$

where n denotes the normalization range with maximum (n_h) and minimum (n_l) and d the real data range with maximum (d_h) and minimum (d_l) [Heaton, 2011]. The denormalization regarding an obtained x' works by simply solving Equation 4.2 for x.
This choice of normalization in an interval between 0 and 1 was favored compared to preparing the data in zero-mean unit variance values. This is, because the former normalization delivered better results in some brief tests which were conducted in advance to the experiments in Chapter 5.

5 Offer Assistance Prototype - Implementation and Experiments

The offer assistance prototype is intended to support the estimation of costs which are expected from certain objectives for a project. Thereby, the general applicability of machine learning – explicitly neural network models – is shown. Section 5.1 first presents the implementation approach, i.e. how the data is handled within the supervised learning process. Then the definition of cross-validation and testing procedures is given. Section 5.2 continues with the specification of the parameters for the neural network models. Afterwards, the results which were obtained by testing the prototype are presented in Section 5.3, followed by a discussion about these outcomes in Section 5.4.

5.1 Implementation

As discussed in Chapter 4, the given data is a challenge for the construction of a proper prototype system. The approach tries to consider the relations between the input and output data in different stages instead of learning an estimator which directly maps objectives to the costs of a project. The reasons for this are explained in Section 5.1.1. The realization of the stages using different neural network models for the prototype is given in Section 5.1.2.

5.1.1 Approach

When preparing the calculations for a project, only basic information and objectives for the product to be developed are expressed by the customer. From this information, a scope of work, which is expected to be needed for fulfilling the objectives, is defined. Furthermore, the extent of tasks and initial wishes of the customer influence the extent of deliverables. Finally, the workload, the deliverables and the general project objectives and desires control the cost which is related to the project, i.e.

- certain basic objective extents may be costly for accomplishment.
- the scope of work directly influences development costs as of time and effort invested.
- the reasonable preparation and presentation of deliverables takes a considerable amount of time.

This connection is intended to be expressed with the approach depicted in Figure 5.1. This approach has the advantage of including the effort or extent of scope of work and deliverables into the cost estimation, but this could also be done in a direct approach and implicitly by the model complexity by including all information at once. The real advantage is given in the sense of intermediate control for learning the overall model and tuning its accuracy. In particular expert knowledge can be injected at each stage of the estimations. This is described in a short example:
Given new project objectives, an extent of the scope of work may be estimated with an existing

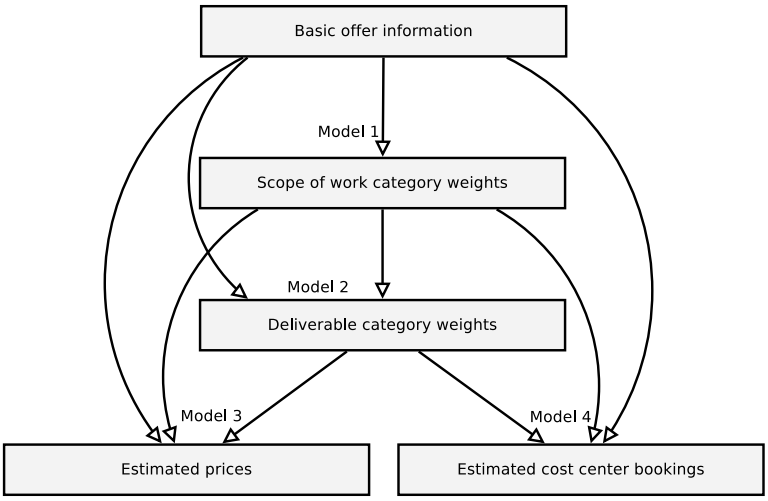


Figure 5.1: Models and dependencies of the offer estimation approach. The basic offer information is used for estimating the extent of the scope of work. Then the (potentially) adjusted scope of work estimate may be included for the estimation of the deliverable effort. The final models for the estimation of cost centers and prices include all previous estimations.

model. Since there were not too many data samples available for training yet, an engineer is occupied with determining the scope of work. The engineer may adjust the workload and let the program estimate the next model, i.e. the deliverables. Thus, more information for the estimation of overall costs is induced.
It is assumed that more information means more data which can be included in the calculations, thus refining the influencing factors. When data of more projects will be collected in the future, the adjustment of efforts like in the previous paragraph's example may be left out and a cost estimation is directly drawn from one general model.

5.1.2 Model Realization

The estimation models are created by the neural network architectures which were introduced in Section 2.3. These models are used with different kinds of activation functions, number of hidden units, as well as other related network parameters. The idea of using several network architectures is to find architectures and configurations which provide the best models for the modules which were introduced in Section 5.1.1.
The general process of selecting models for the different modules is shown in Figure 5.2.

A test set is chosen from the $j = 6$ available project datasets. This set is chosen for the evaluation of the performance of the finally selected models. The remaining $k = j-1$ datasets are used for training and validating the models. Since there is very few data available, there are all possibilities of dataset combinations used for training and validation of the network architectures. Validation is done by leave-one-out cross-validation with different network configurations, which is a fairly well-known technique for small datasets [Seni and Elder, 2010] as it preserves the maximum modeling and testing capability in this case. These configurations are presented in detail in Section 5.2.
Why was a special test set held back for the testing of already validated models? This has to do with the fact that the general applicability of machine learning models shall be shown regarding very scarce and non-homogeneous data. Newly acquired projects will not necessarily provide more and complete information for an estimation algorithm and will also possibly diverge in many aspects from the data of known projects. The same reason applies to the datasets currently available. Thus, the idea is to show which testing error may be expected even for the best models after the selection by cross-validation.

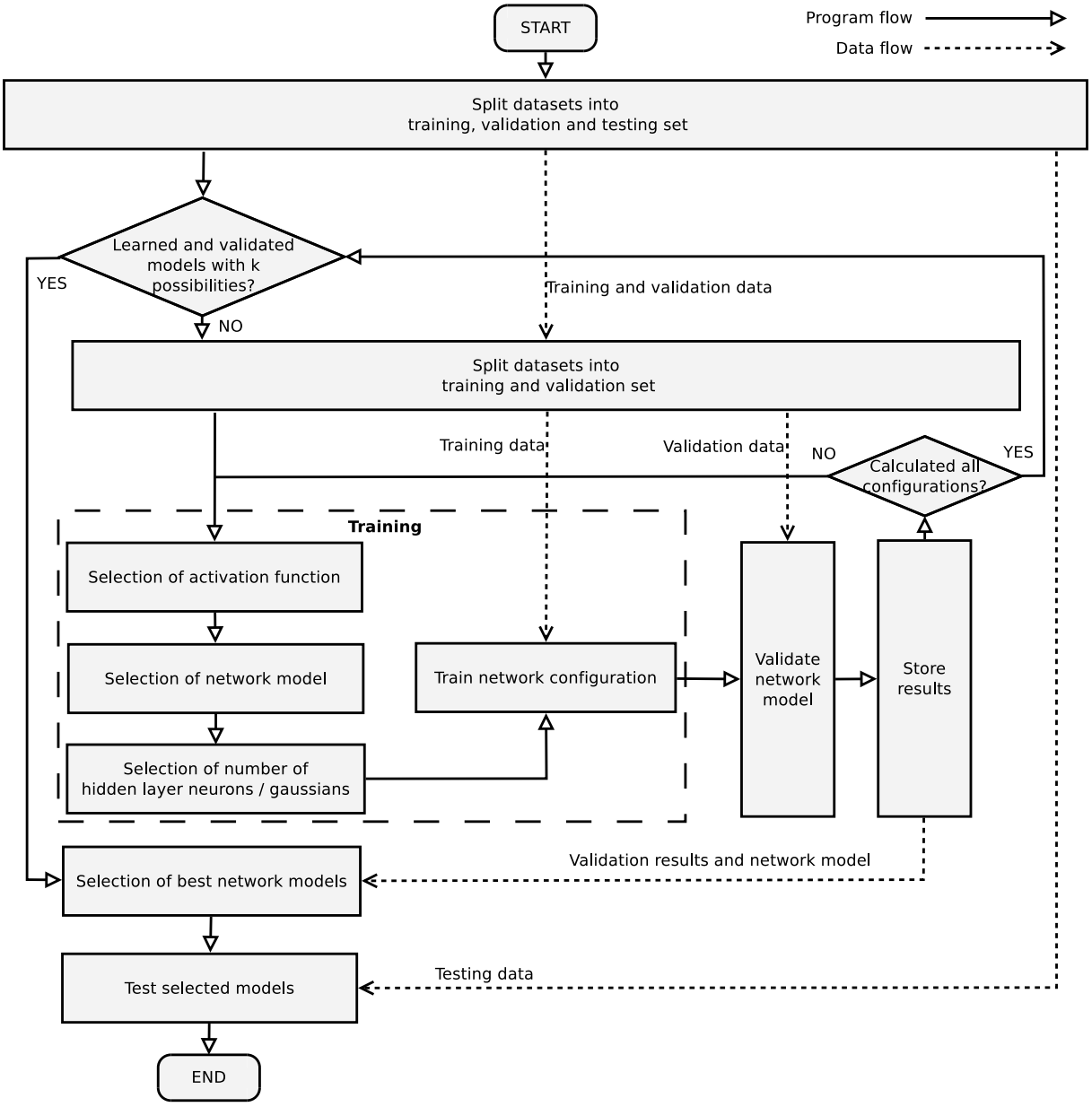


Figure 5.2: Overview of the network modeling and testing process. The shown steps are fulfilled for every module model. The available datasets are split into testing, training and validation datasets. Then, training and validation of the model networks with different configurations is done via leave-one-out cross-validation. The network with the best validation error is selected and tested with the test dataset. Note that the selection of an activation function is limited to the feedforward network, because the other used network methods use Gaussians for the calculation of an activation value.

5.2 Network Model Parameters

Network structure First, the general structures of the used network architectures are defined:

- *Feedforward neural network*: The network owns one hidden layer which is fully connected to input and output layers.
- *Radial basis function (RBF) network*: The network owns one hidden layer of radial basis function (RBF) neurons which is fully connected to input and output layers.

- *General regression neural network (GRNN)*: The inputs of the network are fully connected to the RBF neurons. These neurons are fully connected to the summation neurons.

The number of input and output neurons depends on the model which is targeted. The number of hidden layer neurons is variable except for the GRNN. The feedforward network and the RBF network are set up with $[10, 15, 20, 25, 30]$ hidden layer neurons each. The GRNN works with a number of RBF neurons that are solely dependent on the number of training datasets. The training of the first two network models considers a variety of hidden layer neurons, since this leads to network models of distinct complexities and thus different levels of generalization. There exist some rules of thumb to approximate the number of hidden layer neurons, but these often relate to specific tasks or to the number of training samples. These are not applicable here, because of the lack of data samples and also do not necessarily include the entire complexity of neural networks [Sarle, 2014].

Weight initialization For the initialization of network weights of the feedforward network, the Nguyen-Widrow algorithm [Nguyen and Widrow, 1990] is used which is applicable for neurons having a bias and a bounded activation function. Weight values are selected close to uniform distribution of the input space in the interval $[-1, 1]$. Nguyen-Widrow initialization is of advantage as it has been shown that it may shorten learning time while also having low computational complexity while being task independent [Halawa, 2014].
The RBF network and the GRNN are not in need of weight initialization as they are not dependent on a gradual weight adaption using backpropagation.

Activation function For the feedforward network, the *logistic sigmoid* activation function σ was chosen as the nonlinear activation function of the hidden layer. The output layer is assigned the *identity* activation function ι, i.e.

$$\sigma(x) = \frac{1}{1 + e^{-x}} , \tag{5.1}$$

$$y = \iota(x) = x . \tag{5.2}$$

This choice of activation functions is assumed reasonable, as the hidden layer activations provide nonlinear operations which may be needed regarding the inputs. The identity activation was intended to just sum up the signals from the hidden neurons whereby the outputs were expected to adapt in a good way.
Regarding the Gaussian activation functions of RBF networks and GRNNs, a variance of $\sigma = 1$ was fixed. The center values are set to 1.Unfortunately, there was no was no time for fine tuning these activation function parameters.

Training The used training methods for the networks have already been introduced in Section 2.3, i.e. the feedforward network weights are trained via backpropagation. The feedforward network is trained for several cycles. One training cycle is done by a batch of $k - 1 = 5$ samples. The sample's errors are averaged and backpropagated through the network for the adaption of the weights. The training process is stopped if a sum-of-squares error of 0.05 is not achieved within 200 training cycles. This is a very simple early-stopping approach. The selected error on the one hand seemed appropriate, because it became apparent that further improvement was only hardly possible. On the other hand, the available training samples include rather inhomogeneous data, but are used repetitively. Thus a certain level of generalization was assumed to be maintained with an error that is not too low.

5.3 Results

First, the validation performance of the different neural network architectures and their configurations must be considered. As described in Section 5.1.2, a validation set is used for providing this information. Figure 5.3 shows the results. Thereby, the mean-square error of different network architectures is denoted against the presented models. It is evident that many network models are not really trainable to a low error considering the constraints of 200 training cycles for decreasing the error to at least 0.05.
Nonetheless, there are also rather good models possible. This mainly is the case due to the estimation of the cost centers in model 4 in which about half of the errors are below 0.05. Model 4 does not by chance deliver the best results. The explanation to this follows in the discussion in Section 5.4.

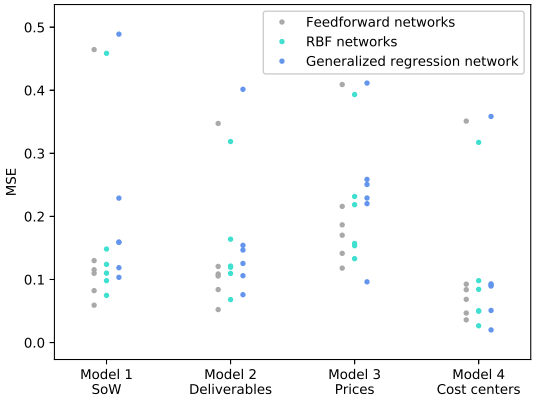


Figure 5.3: The mean squared validation errors of the offer estimation models. For each model and network type, a set of 5 errors is given, which stem from the 5 validation datasets used for obtaining the mean-square error (MSE). For the feedforward networks and RBF networks, which use a varying number of hidden units, the best outcomes regarding the specific validation datasets are depicted. Obviously, a validation dataset which is very uncommon in comparison with the others came up, thus leading to a much higher validation error.

Furthermore, the validation performance distribution of network models is quite similar. For the scope of work (SoW), deliverables and cost center models, there are 4 to 5 models per network architecture producing errors which are closer to each other and the remaining models show higher errors. This distribution of errors can be backtraced to especially one dataset which owns a significantly different scope than the others. If this dataset is selected to validate the performance of a network model, considerable deviation can be expected.
Then, the best models are selected as the estimators of the prototype application and the best models after validation are tested on the testing dataset which was held back. Then, the distribution of percentual errors of categories in the models is investigated. Thereby, the model outputs are compared with the normalized true values of the testing dataset. The test delivers results that are partly expected from the previously mentioned outcomes. These results are shown in Figure 5.4.

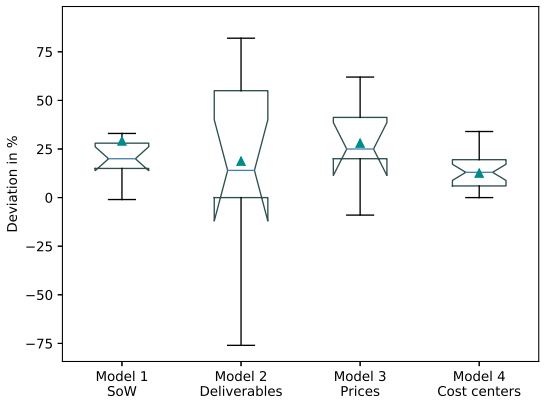


Figure 5.4: The boxplots depict the estimation deviation of offer and cost center information. The normed target outputs are subtracted from the estimated outputs, thus leading to these deviations in %. The box notches indicate the 95% confidence intervals of the median. The triangles represent the mean of the errors per boxplot. As expected from previous results, the cost center estimation is significantly the best with ca. 75% of estimations having a surplus of 0% to 20% to the targets and a very small confidence interval of 8% (length of notch). The spread of all other estimation models is by far higher.

This Figure shows a significantly better estimation accuracy of the best cost center model compared to the other best models. This relates to a small interquartile range (IQR) and the narrow confidence interval (notched part) of the model as well as the small extent of the whiskers, which are set at a maximum of 1.5 times of the IQR. Thus, 75% of the cost center estimations only have a surplus of 0% to 20% to the target values. Admittedly, the notches of the deliverable model and the cost center model overlap, raising questions on the significance of the cost center estimation. However, since the confidence interval of the deliverable deviation is represented by a wide notch, the relevance of the deliverable results may be negligible. Nonetheless, some outliers are also present for the cost center model which may negatively influence the estimation. For all other boxplots in Figure 5.4, a worse confidence interval and a right-skewed distribution is given, i.e. the model outputs must be expected to have a larger deviation and are less likely to be limited to a certain error region. Furthermore, the flipped appearance of these notch plots indicates that the confidence of their means have a higher variance than what is represented by the output data. The respective first quartiles are indicated by the horizontal line in the flipped box part.
An important issue are outliers and further extreme values of the estimations. These mainly originate from information dependencies which are only weakly or not at all existent in the training data. This could be problematic regarding cost estimations via the cost center results. An example may be stated by a normalized cost center which is of value 0.5, denoting 100,000 € that may have been misestimated about 0.2 units, meaning 40,000 € estimation error. Figure 5.5 shows the deviation of the major cost centers for the test dataset. The shown cost centers combined stand for more than 99% of booked working hours and other expenses in a project.

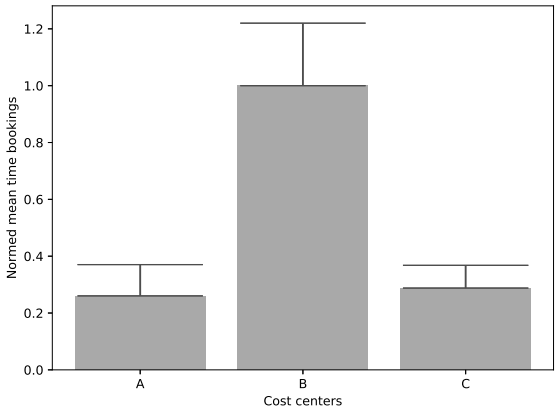


Figure 5.5: Deviation of major cost centers. The positions A, B and C stand for the three cost centers which make up for more than 99% of accumulated costs. Cost center B represents by far the strongest volume of bookings and was taken as a norm of normalization for all cost centers[12]. The black error bars indicate the estimation deviations of the test results.

It can be observed that the deviations may be large compared to the true costs. For the testing dataset a surplus of 0.21 was estimated for cost center B. The estimations for cost centers A and C did not overshoot this strongly, but still in the range of 0.1 to 0.15 which are even higher overshoots regarding relative values. Nonetheless, it must be said that the calculations with the test data resulted in positive mean deviations, i.e. the extent of costs was exceeded on average and thus the financially needed investment for the project accomplishment was not underestimated.

[12] AVL terms restrict sharing the figures and cost center naming of project development.

5.4 Discussion

The overall results must be interpreted with care as there are only few datasets for model training, validation and testing available. Nonetheless, some valuable conclusions can be drawn which relate to the general usability of machine learning in the context of an automated offer creation.

First, the limited number of datasets and their inconsistent representation of contents have a strong impact in the process of building reasonable network models. This may be inferred from Figure 5.3 which shows validation errors for different models. As already stated in the results, datasets which are quite different from common data, i.e. more sophisticated and of larger extent, in comparison to other datasets lead to a higher error if used as a validation dataset. If such a dataset is integrated into the training set, the outcomes of the models are also influenced, but the validation errors apparently do not suffer. As a most crucial fact, it must be stated that only the best achieved cost center models seem to deliver estimations which are usable at the moment, i.e. they are ca. 20% away from the true value.

The better estimation results of cost centers compared to other model results especially can be pinned down to consistent datasets of cost centers. The cost centers for bookings are all the same in the available projects and bookings were done to equal cost centers for equal accomplishments. Quite the opposite was noticed for the estimations of scope of work, deliverables and prices. There is no consistent usage of terms and assignments. Internally, a more precise breakdown of prices would be of advantage. The inconsistency had to be dealt with by several approaches which were presented in Section 4.3. Unfortunately, it is inevitable to introduce noise to the datasets, because the clustering of data into categories was not always possible with 100% certainty and the assumptions to missing data could vary from real values. The importance of clear data types emerges from the experimental results and is strongly recommended.

Nevertheless, despite the lack of training data and partly large varieties fairly well estimations were obtained. The continuation of machine learning projects including a large quantity of datasets should increase the estimation quality. The amount of costs to be expected for a development would already be a major advantage for AVL. Thereby, the extent of costs which were calculated by engineers are comparable with the predictions of the algorithm. This way, the uncertainty of the costs of a project can be reduced.

6 Conclusion and Prospect

This thesis concludes with two major impressions about the realization of systems engineering in the form of a software tool and the applicability of machine learning in such an environment. The first impression stresses that there are still many obstacles left on the way to a systems engineering application which truly enables computer supported collaborative work (CSCW). Some problems must be resolved before. Currently, a strong restriction is the access to data, e.g. in-detail offer figures. There are good reasons for restricting the accessibility of data within the company. Nonetheless, for the sake of improving development methods, it would be wise to reconsider some of these policies. Furthermore, many steps of preparation must still be done for making use of the specified systems engineering application. Regarding this, Section 6.1 provides input on some of the obstacles which could be removed in the battery development department.
The second impression concerns the discovery of many opportunities which seem realizable in the near future regarding the current knowledge base in battery development. It was a matter of this thesis that ideas for the improvement of product development are not only of theoretical nature, but that they are feasible in practice. The main improvements might be found in a clearly structured IT environment including tools and databases. This would provide the foundation to effectively make use of machine learning techniques. This thesis closes with an outlook on the possibilities which may still emerge in the combined field of systems engineering and machine learning in Section 6.2.

6.1 Obstacles and Solutions

A lack of standardized documentation is one of the key drawbacks which was recognized in the course of this work. This especially relates to missing templates or definitions. Some examples subsequently show some of the issues which could be tackled and what may be gained by solving these issues.
A first example targets an aspect which is important for the extent of a battery development project – the maturity level of a customer. It was often stated that this maturity is dependent on several influences: general knowledge and capabilities of the AVL customer, legislative standards of countries for which the customer needs designs, requirements by the customer's product consumers, etc. The maturity level should have been an input to the offer assistance prototype, but in fact, no definition about how to determine this level was provided. Furthermore, even experienced engineers can not provide a ruling as the matter is too complex. Another example relates to the tasks that exist for solving different development problems. There is often room for improvements, especially regarding a consistent documentation. Until now, new tasks were introduced by the development teams. Unfortunately, there was no institution for refining and coordinating the introduced tasks since the battery development department was founded. As such, dependencies between tasks as well as potential overlaps of task competences were not represented. When fixing this, chains of tasks may be built for accomplishing certain targets, milestones or deliverables. The installation of these task dependencies then provides the possibility of applying stochastic sequence models in order to find suitable tasks for various battery

developments.
This also solves another problem. It is hard to develop machine learning methods on data which is hidden in the texts of the offers. Having nicely structured tasks helps to extract relevant information to support data-driven algorithms. Examples of task chains may help to present different offer possibilities to a customer.

6.2 Outlook

Finally, it is important to understand that much of the functionality of a systems engineering tool does not just apply to the field of battery development. The joint usage of such a tool would provide quicker exchange of information with all engineers in the company. Furthermore, a reasonable and coherent basis for storing data may be built which provides a foundation for using machine learning on a large scale. For example, algorithms which suggest tasks for certain development processes could be applicable in other departments of the company.
However, before introducing functionality like the offer assistance application in teams or company-wide, proper functioning must be validated and confirmed. This is to ensure a highest possible user acceptance. In order to provide these requirements, an investment into research and more resources are needed. One option would be to introduce specific flagship projects. These flagship projects could target work processes which are unintuitive and/or maybe too sophisticated for many employees. A similar approach indeed already exists in the systems engineering laboratory, but the potential of working on a single project in a team was not enforced yet. Advantages would be: quicker development cycles, specialists could directly interact in projects instead of many meetings to roughly explain in-detail knowledge.
Additionally, it is essential that new software tools require a proper introduction to the people who need to work with these innovations. A smooth introduction takes time and software training is needed. If the employees do not have the impression of wasting time which could be used for other productive work, much acceptance can be gained.
Finally, as mentioned in Section 5.4, the applicability of neural networks for supporting a process in battery development could be partly shown. The careful reader may have recognized many cases where improvements can be done, e.g. the presented model structure was developed due to the lack of data to work with. Furthermore, a segmentation into several models can be envisioned, i.e. requirements could be derived from objectives in a first machine learning model, then task chains can be built from these requirements in another model and so on. All in all, data will be the key for further developments regarding methods and techniques of machine learning. Thus, the coordination of access to all required data will have to occupy a high priority in the management of the company.

A

MBSE-System Concept - Requirements

This chapter specifies the requirements needed for the software application (MoBat-System) in order to be beneficial for the work of the battery development skill team. Generally, the given requirements are adoptable for other skill teams and departments at AVL or are only in need of some adaptions.

A.1 Functional Requirements

This section states the functional requirements which must be handled by the system. These also comprise other application systems and databases that are reasonable to include. An overview on the different systems which enable the functionality and major user characteristics has already been given in Section 3.4. A major number of these requirements is considered to be applied to the core-system. Thus, requirements generally must be implemented for the core-system. There are also requirements which have to be implemented in the mobile system and in the core-system. In order to ease the retrieval of requirements for the mobile system, Table A.1 can be consulted.

Section	**Heading**
A.1.2	Assignment of User Rights
A.1.3	Project Offer Assistance
A.1.4	Message Assistance
A.1.5	Project Document Filing
A.1.12	System Test and Enhancement Management

Table A.1: Mobile system requirement sections.

A.1.1 Productive Data

This section is intended to give an overview on the fundamental objects and terms of the MoBat-Application and their attributes in the course of product development. The breakdown of these elements is essential for fully understanding the requirements for this systems engineering application. The system objects that are introduced here already exist partly in the *MoBat DB*, which is designed to be extensible for additional data objects. For the major system objects *project*, *objective*, *requirement*, *task*, *function* and *component*, it must be stated that the attribute field *comment* matters a lot as it will give the possibility of registering and linking information like advantages but also problems with the object. These are often referred to as *lessons learned*.

Project

A project gathers instances of different system objects in order to describe the project. A project may be a single product or a collection of products to be developed. Thus, as sometimes, product is a more descriptive term than project, it may sometimes stand for a whole project. Table A.2 holds explanations on the project attributes.

Attribute	**Description**
ID	Unique identifier of the project.
Name	Name of the project.
Customer	Customer who is involved in this project.
Customer Maturity	Maturity level of the customer.
Management Responsible	Person who is management responsible for in this project.
Development Responsible	Person who is responsible for product development in this project.
Planning Responsible	Person who is responsible for planning the project.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.
Objectives	Objectives that are defined for this project.
Requirements	Requirements that are defined for this project.
Tasks	Tasks that are defined for this project.
Functions	Functions that are defined for this project.
Components	Components that are defined for this project.

Table A.2: Project attributes.

The term *customer maturity* demands some more explanation. As there is a large variety of customers who have different experience regarding the stages of a battery development, they are assigned a maturity level. The maturity level indicates which level of advancement is desired for a product. There maturity level is usually referred to as a sample in the range from A to F, where A is the most simple sample[13]. The maturity of a sample is dependent on the expected location-specific development extent. This means that some regions in the world have very high standards which often go hand in hand with costly demands while others have not.

Objective

An instance of an objective object can be assigned to a project. An objective is a customer's rough definition of a product property. Table A.3 holds explanations on the objective attributes.

[13] According to customers, different sample classifications are possible and must be converted to AVL terms.

Attribute	Description
ID	Unique identifier of the objective.
Name	Name of the objective.
Description	Description of the objective.
Requirements	Requirements that can be derived from the objective.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.

Table A.3: Objective attributes.

Requirement

An instance of a requirement object can be assigned to a project. A requirement is derived from a customer's product objective and describes needs of the project in order to develop the product. Requirements and objectives are often hard to distinguish. The decision between objective and requirement assignment has to be discussed and met by the users of the application. Table A.4 holds explanations on the requirement attributes.

Attribute	Description
ID	Unique identifier of the requirement.
Name	Name of the requirement.
Description	Description of the requirement.
Objectives	Objectives from which the requirement is derived.
Tasks	Tasks that can be derived from the requirement.
Components	Components that can be derived from the requirement.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.

Table A.4: Requirement attributes.

The table also shows a component being a potential part of a requirement. Usually a component may be chosen by looking on the function which a product has to fulfill and an appropriate component is chosen from this, but the case that the customer wants a specific component to be applied in the product may occur.

Task

An instance of a task object can be assigned to a project. A task describes a part of or a whole work package that has to be processed in a project. Tasks are generally derived from the product requirements. Table A.5 holds the task attributes and their definitions.
Tasks and requirements are very important for the product developers to understand the goal

and the steps towards the solution of a product development. Thus they must be well defined and put into relation to each other by the final users of the system.

Attribute	**Description**
ID	Unique identifier of a task.
Name	Name of the task.
Stakeholder	Groups of persons (departments, skill teams or parts of these) who share involvement in the task.
Description	Description on which input the task shall convert to which output.
Content	Detailed contents which are covered by the task.
Phase	Phase which the task is assigned to.
Concept	Development concept which is pursued by the task.
Sample	The sample, indicates the maturity level of the overall product.
Output Type	Defines if the output is simply an output for another task or if the output is also marked a deliverable.
Deliverable Date	A date when the deliverable is due is stated if the output type is a deliverable.
Milestone	Milestone which the task is assigned to.
Preceding Tasks	Task(s) that must be fulfilled in order to start this task and are settled directly before this task in the chain of tasks.
Requirements	Requirements from which the task is derived.
Functions	Functions that can be derived from the task.
Median Duration	Median duration of all previously finished applications of this task. This may be regarded as a duration estimate of the task.
Project-related Duration	Attribute that is related to a certain project and states the work time invested in this task.
State	Binary state that is related to a certain project which indicates if the task is finished or in progress.
Comments	Attribute that is related to a certain project which holds special textual information on the project task.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.

Table A.5: Task attributes.

Some attributes demand additional explanation on their meaning. The *phase* describes which stage a task is assigned, e.g. innovation and strategy, feasibility, concept, development, series

preparation, ramp up or series. The *concept* indicates which vehicle concept is related to the task, i.e. for battery development: electrical vehicle (EV), plug-in hybrid electrical vehicle (PHEV) or hybrid electrical vehicle (HEV). The sample which indicates the maturity of the overall product to be developed is specified as an alphabetic character from A to F. The *output type* states if the task is a *deliverable* or not. Generally, a collection of tasks that depend on each other result in a deliverable. That means preceding tasks are accomplished serially or in parallel and lead to a certain task. Any task may be selected as a deliverable if AVL and the customer meet an agreement on it. Usually, a collection of deliverables can be seen as a *milestone*, which marks a major step in development.
Eventually, the *Description* and *Content* fields of the object differ in the following way: The description only gives a rough overview on what shall be accomplished, e.g. the waterproofness of components must be verified. The more detailed content would exemplarily list possible components, that they would be analyzed by simulation and that components with proper simulation results are tested in real test environments.

Function

An instance of a function object can be assigned to a project. A function describes a certain functionality of a battery module. The function needs battery module components for its realization. Table A.6 holds explanations on the function attributes.

Attribute	**Description**
ID	Unique identifier of the function.
Name	Name of the function.
Description	Description of the function.
Tasks	Tasks from which the function is derived.
Components	Components that can be derived from the function.
Parameters	Parameters that are assigned to the function.
Comments	Attribute that is related to a certain project which holds special textual information on the project function.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.

Table A.6: Function attributes.

The parameters to be assignable to the function object is depending on the kind of function. As there is a large variety of functions, parameter objects must be set up separately in order to ensure a consistent collection of parameters.

Component

An instance of a component object can be assigned to a project. One or more components contribute to a certain functionality of a battery module. Table A.7 holds explanations on the component attributes.
The parameters to be assignable to the component object is depending on the kind of component.

Attribute	Description
ID	Unique identifier of the component.
Name	Name of the component.
Description	Description of the component.
Functions	Functions from which the component is derived.
Requirements	Requirements from which the component is derived.
Parameters	Parameters that are assigned to the component.
Comments	Attribute that is related to a certain project which holds special textual information on the project component.
Editor	Last editor of the template object.
Project Editor	Last editor of project assigned content.

Table A.7: Component attributes.

As there is a large variety of components, parameter objects must be set up separately in order to ensure a consistent collection of parameters.

Further Data Items

Hereinafter, a short definition of terms used with the application is presented. These are intended to exist as standardized forms. If filled, they are stored to *Microsoft SharePoint* locations of assigned projects. Their configuration data must be stored within configuration files which can be created, edited and used for forms filling within the MoBat-System.

- *Bidding information*: This kind of information concerns experiences made in the bidding process for a project. Usually, a potential customer calls for offers on a project with certain objectives. The product management of AVL has to include knowledge on previous quotations and if they were successful in the project acquisition as well as specificity on the customer for the bid creation.
- *Offering*: An offering includes the basic objectives and requirements, work packages and deliverables and the bid for a project. AVL wins a bidding process, if an offering is signed by the customer.
- *Product information*: Previously finished projects at AVL lead to learned lessons on specific developments, e.g. constraints of a product construction or further potentials. A collection of this knowledge is usable for negotiations with customer as well as for the estimation of the feasibility of projects.

A.1.2 Assignment of User Rights

Involved databases and tool systems:

- MepSec System

Core System Requirements

The system must provide the functionality to receive user rights for system usage. A unique AVL employee may be assigned to the system as a user. For identification, a user's full name and the AVL mail address are used. For better recognition the user selection may be supported with respective AVL position, associated AVL department and AVL subsidiary company name. This information has to be consistent with the global company *Microsoft Outlook* user list[14].
A user must be selectable by a rights assignment responsible person in order to add, modify or delete their user rights. User rights are separated into *Administrative Rights* and *Content Rights*. The distribution of these rights must be assignable with the MepSec system. General rights and how they may be distributed by whom are given in Section A.6.3. Rights within the system functionality are explained alongside those. Finally, it must be enabled to create user rights templates and store them to the system.

Mobile System Requirements

The access to the mobile system is generally possible if a company phone has received the mobile application install (see Section A.6.2 for install requirements). Unlike in the core system, the mobile system must not receive user rights because of install requirements. The right of usage for the mobile application is given by logging in to the phone. The application functionality may be seen as a cohesive functionality which partly provides some mean of communication with the system.

A.1.3 Project Offer Assistance

Involved databases and tool systems:

- Salesforce
- SAP System (proCalc)
- Microsoft SharePoint
- PTC Integrity System (Lifecycle Manager)
- MoBat DB

Core System Requirements

This functionality shall exist for supporting the project planning process with a tool to estimate work packages, deliverables and cost for a certain project and customer. This way, the customer shall receive an offer that is tailored to their needs by reuse of information of previous projects. Furthermore, the cost for compiling the offer may be reduced as several days of working hours are conserved.
The offer generation is initiated by a dialog between AVL and the customer and the request for quotation regarding certain objectives. The estimation procedure takes the objectives for a product which are imported into the MoBat-System from a standardized form or user interface. For the new request for quotation, the system prompts the *Salesforce* system to generate a new project which is identifiable with an AVL-wide unique ID assignment. The project entry is assigned data on customer, objectives and requirements as they are given in the request. If the request already contains an AVL project ID, the ID must be adopted. The same data proceedings apply to the *PTC Integrity Lifecycle Manager* (PTC LM). Regarding the *SAP* system, specifically the *proCalc* offer calculation tool, a project with the same project ID is to be generated. A new calculation form automatically may be requested by usage of the mail-driven interface of the *proCalc* tool.
Furthermore, a project repository, named with the project ID is to be set up in the *Microsoft*

[14] A global contact list is distributed via the company's *Microsoft Exchange Server* install.

SharePoint web application of AVL. The repository is to be located in a *SharePoint* location of the AVL department which is majorly concerned with the project. The initial access rights to the repository are to be coordinated by the product manager or product planning manager who coordinates the start-up phase of the project.
An exemplary solitary application[15], estimating the cost according to bookings for certain projects, was implemented as a proof of concept within this thesis. The detailed explanations on the approach may be viewed in Chapter 5. The application's functionality may be incorporated by this functional system requirement, but must be enhanced at large for providing true usability in daily engineering. Additions to the applications idea are that the information must be applied to a learning system (machine learning system) which makes use of offering and quotation data of earlier requests by the customer and must include task and deliverable information that may be derived from the request objectives and requirements. A necessity for usage of these information items is a standardization of data and its storage in the *MoBat DB*. The *proCalc* task items for the project may then be estimated regarding working hours, cost as well as by RASI information (see Section A.5.1 for the definition of RASI). Finally, a list of task items with the task dependencies shall be suggested by the application. The tasks which really shall be used in the project must be choosable by an application user. Thus, a quality control and a even more customer- and project-accurate selection may be applied by experienced engineers.

Mobile System Requirements

Additionally involved tools:

- Microsoft Office for Android and iOS

The mobile system is to provide standardized forms to collect objectives and requirements from the customer. These forms must be compatible for usage with the *Microsoft Office* tools on *Android*- and *iOS*-driven devices. These operating systems must be of the newest generation at the moment of ordering the development. The same holds for the case that another major operating system is introduced for the company's mobile devices. This ensures that the MoBat-System may interpret and process the data in an automated way.

A.1.4 Message Assistance

Involved databases and tool systems:

- Microsoft SharePoint
- MoBat DB

Core System Requirements

In the process of acquiring a project as well as during project development, status files must be sent to groups of people according to the state of the project, i.e. acquisition, development, reached milestone, etc. Support for these necessities is to be offered by the system's message assistance functionality which enhances standard mail functionality.
The service is intended to exploit data which shall be contained within the e-mail message. This data at least refers to the project ID or unique project name (project identifiers), the name of the message's author and additionally to different technical terms in the message body as well as potential attachments of the message. These information items shall be analyzed, whereby a preselection of receivers as well as a subject suggestion shall be made. E.g. a mail from a

[15] This application currently uses some basic inputs and weighted work package and deliverable data (because of sparsely correlated data) of a handful of finished projects and estimates the expected cost center values.

project manager containing a request for quotation may be automatically assigned the subject *Request for quotation (customer name)* and the e-mail address of the project planning manager as recipient. Information on personnel involved in respective projects may be obtained from *MoBat DB* entries by using the project identifiers.
If attachments are selected from the *Microsoft SharePoint* repository for being sent, potential timestamp information which is contained in the file names (see Section A.1.5) must be removed. Additionally, the functionality provides the option to set default settings for a certain user. These include often used projects with corresponding recipients and so on.
Regarding the receipt of mails, the system shall own a so called system recipient. This is an artificial system user provided by the system including an own mail address. Information which is sent to this user may be processed by the system. Jobs and further information for the system to process, may be wrapped in the header of the e-mail[16]. A close partner function of the message assistance is the automated filing of received files which is specified in Section A.1.5. The general process for the message processing and automated document filing works in the following way:

- Message sending
  1. Creation of a mail subject from initially required information.
  2. Extraction of possible mail recipients and selection of the system recipient(s).
  3. Definition of mail header information, so that the system knows which way the message has to be processed.
  4. Sending of the message.
- Message receipt
  1. Receipt of the message by the artificial system user.
  2. Evaluation of the mail header.
  3. Unambiguous filing of possible mail attachments to the right *Microsoft Sharepoint* directory.
  4. Linkage of documents considering their project identifiers to respective *Salesforce* entries.

Mobile System Requirements

The core system requirements are to be adopted for the mobile system. A difference for the mobile user is the fact that this system must request project-related persons' contacts from the *MoBat DB*. Information could be provided by a user's request mail to the system which stores this information to the mobile system. This request for information could be realized by a command in the mail header which tells the core system how to proceed. The mail must also contain a project ID for this purpose. Furthermore, it would be reasonable to be able to request contacts which are assigned to a specific department and hold a certain position at AVL.

A.1.5 Project Document Filing

Core System Requirements

Involved databases and tool systems:

- Microsoft SharePoint

[16] Maybe the spam control of AVL mail servers has to be adapted for enabling enhanced messages to pass.

The filing process works hand in hand with the presented message assistance functionality. Regarding the unique project identifiers delivered by message headers of a received e-mail, a *Microsoft SharePoint* location for the project may be identified. If none is existent, yet, a new repository shall be created and named with the project identifiers. According to different states of the project, different subdirectories must be accessible for storage, i.e. acquisition, development, reached milestone, etc. The documents that are attached to the mails must be filed to these particular project directories. The document names must be joined a timestamp in order to be stored unambiguously. An exemplary new file name would thus be of the format:

<name> + <_YYYY-MM-DD> + <_HH-MM-SS> + <.extension>

where name states the original file name, the first added term provides the date by year month and day (e.g. 2017-12-13) and the next joined term the time of day by hour, minute and second (e.g. 23-59-59).

Mobile System Requirements

The requirements of the core system can be directly transferred to the mobile system with one difference. The mobile application is to provide an own storage management and does not interfere with the documents filed on AVL servers. The unambiguously naming conventions and storage concept may be applied likewise for the application's filing directory.

A.1.6 Battery Development Data Management

This functionality provides means of viewing and modifying information on projects. Information on the project objects is visible to all users who own content rights for a project. Modifications shall only be possible for object attributes and parameters which users are enabled for. The rights for modifications are described subsequently in the respective sub-functionality.

Project Development Management

Involved databases and tool systems:

- MoBat DB
- PT DB

This functionality refers to operations with project development content. Project content which is assigned to a project must be accessible from the *MoBat DB*. Project-relevant data on component attributes must be accessible from the *PT DB*. On a request for project data, the system checks which kind of access rights the requesting user has to the project data. If there is no access right registered for the user, no data is loaded from the MoBat database. The data items held by the database were already addressed in Section A.1.1.
The project development management offers the possibility to add, modify, delete or just load project information depending on user rights. Table A.8 shows the objects of a project which are used with this functionality and which objects are partly modifiable in this context with respect to the RASI distribution of context rights.
An important feature of this functionality is the permission of accessing similar projects which are permitted to be viewed (nonetheless the origin of data is anonymized as a protection to prevent certain customer's projects from being targeted). The similarity of projects may be rated by an overlap count of similar or equal objectives and requirements as well as value similarities and deviations of constraints and boundary data. An idea for spotting similarities in data would possibly be subspace clustering or correlation clustering as these may avoid the so called curse of

Content Project	Modification Rights of Project			
Objects	**Responsible**	**Accountable**	**Support**	**Informed**
Project Management Responsible	yes	veto	no	no
Project Development Responsible	yes	veto	no	no
Project Information	yes	yes	no	no
Project Objectives	yes	veto	no	no
Project Requirements	yes	yes	no	no
Project Tasks	yes	yes	no	no
Project Functions	yes	yes	yes	no
Project Components	yes	yes	yes	no

Table A.8: Default modification rights of project development content users for project objects.

dimensionality [Kriegel et al., 2012], which refers to various phenomena occurring at analyzing and organizing data in high-dimensional spaces.
A necessity regarding the exploitation of different project's information is a project-related opt-out clause for the customer. As it may be the case that a customer does not wish to feed their development data into the assisting system, the usage of this information for other customer's projects may be ruled out. Likewise, the usage of other customer's project information is to be denied for this project. Like this, customers have the decision to buy a development advantage, e.g. sped up development and reduced costs, or stick to knowledge which was developed only for them.
The attributes and parameters of modifiable project objects and attributes are described in the following paragraphs.

Project Development Responsible Person: The responsibility for the project development may only be yielded and assigned to another person by the currently responsible person. The project accountable person must approve this. Furthermore, the development responsible person is eligible to appoint content users and their access rights.

Project Requirement: Project requirements are generally set in the beginning or prior to the development phase. Nonetheless, the need of changes may emerge in the project advance. The development responsible person is authorized for applying modifications after the approval by the accountable person.

Project Task: The task's *accomplishment state* is a binary state. It is set to *not accomplished* by default and may be set to *accomplished*. Also functions for this task may be edited or newly defined in the project. Removal and adding of tasks may only be performed by the project responsible person in coordination with the accountable person.

Project Function: Project functions may be selected by related rights holders. A selection suggestion must be provided by derivation from project tasks. Considering selected functions, components may be assigned by selection from a recommendation list or menu. The information for these selection options is provided by the *MoBat DB*.

Project Component: Project components may be selected by related rights holders. A selection suggestion must be provided by derivation from project functions.
The component parameters are to be extracted from the battery data of previous projects in the *PT DB*. Their default values are an estimation from these projects and must be marked as such in order to prevent potential confusion of developers. When parameter values are set

or edited in the system, the system must initiate modification of these parameters in the *PT DB*.

Generally, the content responsible person and the content accountable person always both have to agree on every change. The modification rights are subjected to the content rights given in Section A.6.3 and thus the approval by content responsible person and accountable person may be delegated to a third person.
Time stamps on all project modifications as well as records on modification-responsible users must be kept with the project data related to each parameter and attribute.
Eventually, it has to be stated that lessons learned records on all system objects (comments field of objects) of previous projects must be accessible to system users unless these were locked by related customers.

Simulation Data Management

Involved databases and tool systems:

- MATLAB Simulink
- MoBat DB
- PT DB

The simulation data management provides the functionality to store and provide simulation parameters for feeding the already existing MATLAB Simulink battery simulation model with project-relevant data.
A single project to work on must be selectable from the simulation data context menu of the MoBat-System. This is enabled by the possession of related access rights. A number of accomplished projects that used similar functions and components like the selected project must be identifiable. Like the adjacency between projects in the project development management, an estimation of simulation parameters may be achieved by exploiting similarities to previous project data (subspace clustering, artificial neural networks). The data which may be used here are constraints and key data of functions or components to be accomplished for the battery. These are to be taken from the *MoBat DB* and the *PT DB* as well as have to be stored there when applied to a product.
The *MATLAB Simulink* tool, is a solitary application on AVL's computers so far. Thus the MoBat-System must also provide the function for the user to export and import data from standardized forms and into the system and backwards. The standardized form shall be read and created by a *MATLAB* function. This way, the manual effort for the user is kept low.

Project Development Tracking

Involved databases and tool systems:

- SAP System (ESZ, proCalc)
- MoBat DB
- PT DB

Tracking with regard to project development has two issues:

1. Comparison of project related time exposure for tasks with the time consumption of equal tasks in similar projects.
2. Investigation of the accomplishment share of the project regarding its tasks, deliverables and milestones.

Time Exposure Comparison A project must be selectable from a choice of similar finished projects. The similarity may be found like described for the development management and

tracking in previous sub-sections. Adjacencies of tasks, functions and components are promising for finding a close project. These projects working hours on tasks are to be extracted and set into relation to the tasks of the current project. The data for the project similarity calculation may be taken from the *MoBat DB* and the *PT DB*.
The collection of information on the time consumption on a projects tasks can be handled with the help of the *ESZ* tool which records employees working hours. Entries in this tool are dealt to certain cost centers. The entries are incorporated by the *SAP Booking* system servers. As the *ESZ* tool allows comments to its entries, a identification of working hours for certain tasks may be enabled by issuing the project ID and task ID for each entry. The MoBat-System must be enabled to extract the comment information from AVL's *SAP Booking* system servers without cutting down on the system user's personal rights. The bookings for certain projects may be extracted by feeding the SAP system one or more project-specific booking numbers (these are also used for entries in the *ESZ* tool). These booking numbers can be found in the company's *proCalc* user interface.
As the explicit association of employee involvements in specific tasks would be possible, the system functionality has to provide means of anonymizing this information for the usage in the system and for protecting it from anyone's access. Only the project-related information is important here. It could be useful to introduce sanctions, if any interference in the personal rights of employees is intended by exploitation of this functionality by any enhancements.

Accomplishment Investigation The project task instances must be extracted from the *MoBat DB*. The information on tasks must be sortable by task sequence and task phase as well as task milestones. The sorting requirements for the functionality go hand in hand with the functionality of a user interface that is presented in Section A.4.2. The representation of the project progress must also be available as a Gantt chart of tasks. The tasks' accomplishment states and durations as well as deliverables and milestones must be shown in the chart. The task elements must be selectable for closer investigation of all their attributes.

A.1.7 Objective Management

Involved databases and tool systems:

- MoBat DB

The objective management offers the possibility to add, modify and activate or deactivate objective objects according to user rights. The rights for this functionality are context usage rights for adding and modification. The proceedings by these usages must be accepted by a context responsible person. Deactivation rights are only permitted to specialized system administrators, who are characterized in Section A.6.3, by following a request by a context responsible person.
The creation of a new objective object initially requires all its attributes to be populated, except for the objective ID, which shall be generated uniquely and automated, and the requirements.
The modification of objectives must be possible for all their attributes, but the unique ID. The objectives' attributes are given in Section A.1.1.
Furthermore, time stamps on objective creation, modification and deactivation as well as records on users who are responsible for these actions are kept. These records also must include objective attribute changes.
Eventually, the implementation of the functionality is in need of a search function that is capable of finding an objective regarding a certain objective name, objective ID or keywords of the objective. The search request's results must be represented in a list of potentially sought objectives. The list must feature sortability with respect to objective names and IDs as well as by requirement names and IDs associated with the objective. Featured search options that are non-related to attributes are the creation/modification time stamps of the objective and the

separation in active and deactivated objectives. If these search options do not cancel out each other, all the search filter options must be applicable at the same time.

A.1.8 Requirement Management

Involved databases and tool systems:

- MoBat DB

The requirement management offers the possibility to add, modify and activate or deactivate requirement objects according to user rights. The rights for this functionality are context usage rights for adding and modification. The proceedings by these usages must be accepted by a context responsible person. Deactivation rights are only permitted to specialized system administrators, who are characterized in Section A.6.3, by following a request by a context responsible person.
The creation of a new requirement object initially requires the name and the description attributes to be populated. The requirement ID is to be generated uniquely and automated. The remaining attributes may be filled later.
The modification of requirements must be possible for all their attributes, but the unique ID. The requirements' attributes are given in Section A.1.1.
Furthermore time stamps on requirement creation, modification and deactivation as well as records on users who are responsible for these actions are kept. These records also must include requirement attribute changes.
Eventually, the implementation of the functionality is in need of a search function that is capable of finding a requirement regarding a certain requirement name, requirement ID or keywords of the requirement. The search request's results must be represented in a list of potentially sought requirements. The list must feature sortability with respect to requirement names and IDs as well as by objective names and IDs, task names and IDs and component names and IDs that are associated with the requirement. Featured search options that are non-related to attributes are the creation or modification time stamps of the overall requirement and the separation in active and deactivated requirements. If these search options do not cancel out each other, all these search filter options must be applicable at the same time.

A.1.9 Task Management

Involved databases and tool systems:

- MoBat DB

The task management offers the possibility to add, modify and activate or deactivate task objects according to user rights. The rights for this functionality are context usage rights for adding and modification. The proceedings by these usages must be accepted by a context responsible person. Deactivation rights are only permitted to specialized system administrators, who are characterized in Section A.6.3, by following a request by a context responsible person.
The creation of a new task object initially requires the following attributes being populated by the user: name, description, stakeholder, phase, preceding tasks. The task ID is to be calculated uniquely and automated as well as the median duration of the task. The remaining attributes may be filled later. A newly created task is always set into review mode. Only after content responsible persons of all involved stakeholders mark the task as accepted, a task is set operational.
The modification of tasks must be possible for all their attributes, but the unique ID, the median and the project-related duration which are all calculated by the MoBat-System. The tasks' attributes are given in Section A.1.1. Moreover there is a number of attributes that may only

be edited for a project-assigned instance of the task, which encompasses the output type, deliverable data, milestone, preceding tasks (e.g. if an item in the chain of tasks is considered omittable) and the state.
Furthermore time stamps on task creation, modification and activation or deactivation as well as records on users who are responsible for these actions are kept. These records also must include task attribute changes.
Eventually, the implementation of the functionality is in need of a search function that is capable of finding a task regarding a certain task name, task ID or keywords of the task. The search request's results must be represented in a list of potentially sought tasks. The list must feature sortability with respect to task names and IDs as well as by stakeholder names and IDs, requirement names and IDs, phase, concept and task dependencies that are associated with the task. Featured search options that are non-related to attributes are the creation/modification time stamps of the task and the separation in active and deactivated tasks. If these search options do not cancel out each other, all the search filter options must be applicable at the same time.
As the task management function contains a lot of attributes which are a challenge to manage, because of their amount and dependencies, Section A.4.2 gives a suggestion for the creation of a user interface.

A.1.10 Function Management

Involved databases and tool systems:

- PTC Integrity System (Modeler)
- MoBat DB

The function management offers the possibility to add, modify and activate or deactivate function objects according to user rights. The rights for this functionality are context usage rights for adding and modification. The proceedings by these usages must be accepted by a context responsible person. Deactivation rights are only permitted to specialized system administrators, who are characterized in Section A.6.3, by following a request by a context responsible person.
The creation of a new function object initially requires the name, the description and the parameter attributes to be populated. The function ID is to be generated uniquely and automated. The remaining attributes may be filled later. A newly created function is always set into review mode. Only after content responsible persons of all involved stakeholders mark the task as accepted, a function is set operational.
A population of functions shall be supported by the extraction of information from a generic battery model that was modeled with the *PTC Integrity Modeler* [Lugger, 2016]. This model features a generic set of function and component options for batteries.
The modification of functions must be possible for all their attributes, but the unique ID. The functions' attributes are given in Section A.1.1. Moreover the parameter attribute may only be filled with values when a function instance is assigned to a project.
An essential attribute for the physical function description is the parameter attribute. Parameters vary in number, but often these physical values are also used with other functions. Thus, a collection of parameters must be creatable and modifiable and parameter consistency amongst functions must be ensured.
Furthermore time stamps on function creation, modification and deactivation as well as records on users who are responsible for these actions are kept. These records also must include function attribute changes.
Eventually, the implementation of the functionality is in need of a search function that is capable of finding a function regarding a certain function name, function ID or keywords of the function. The search request's results must be represented in a list of potentially sought functions. The list must feature sortability with respect to function names and IDs as well as by task names and

IDs, component names and IDs and parameters that are associated with the function. Further search options that are non-related to attributes must be featured, i.e. the creation or modification time stamps of the component and the separation in active and deactivated components. If these search options do not cancel out each other, all these search filter options must be applicable at the same time.
Likewise, a search for function parameters must be possible. The parameters must be searchable and sortable by keywords.

A.1.11 Component Management

Involved databases and tool systems:

- PTC Integrity System (Modeler)
- SOLIDWORKS Enterprise PDM
- MoBat DB
- PT DB

The component management offers the possibility to add, modify and activate or deactivate component objects according to user rights. The rights for this functionality are context usage rights for adding and modification. The proceedings by these usages must be accepted by a context responsible person. Deactivation rights are only permitted to specialized system administrators, who are characterized in Section A.6.3, by following a request by a context responsible person.
The creation of a new component object initially requires the name, the description and the parameter attributes to be populated. The parameter attributes shall be importable from the *SOLIDWORKS Enterprise PDM* system in which the CAD models and physical attributes of the components are created and stored. This includes the component ID which already is unique in this EPDM system. Some parameters may have fixed values and must be disabled from modification at the component creation in the MoBat-System. An assignment of potential functions to this object must be performed by extraction from the generic battery model which was modeled in the *PTC Integrity Modeler* [Lugger, 2016]. The remaining object attributes may be filled later.
An essential attribute for the physical component description is the parameter attribute. Parameters vary in number, but often these physical values are also used with other components. Thus, a collection of parameters must be creatable and modifiable and parameter consistency amongst components must be ensured.
The modification of components must be possible for all their attributes, but the unique ID and fixed component parameters. The component attributes are given in Section A.1.1. Moreover, modifiable component parameters may only be filled with values when a component instance is assigned to a project. All parameter values must be stored and edited in the right project instance of the *PT TB*.
Furthermore time stamps on component creation, modification and deactivation as well as records on users who are responsible for these actions are kept. These records must also include component attribute changes.
Eventually, the implementation of the component is in need of a search function that is capable of finding a component regarding a certain component name, component ID or keywords of the component. The search request's results must be represented in a list of potentially sought components. The list must feature sortability with respect to component names and IDs as well as by function names and IDs and parameters that are associated with the component. Further search options that are non-related to attributes must be featured, i.e. the creation/modification time stamps of the component and the separation in active and deactivated components. If these search options do not cancel out each other, all these search filter options must be applicable at the same time.

Likewise, a search for component parameters must be possible. The parameters must be searchable and sortable by keywords.

A.1.12 System Test and Enhancement Management

A test environment, which is to be used for implementing or porting and testing new MoBat-System features, must be realized. The system is intended to use a clone of the system (or parts of it according to needs) which is operated with test data. Untested feature enhancements may be tested in this environment and are to be ported to the productive system if tests have been successful and clearance by the system's administrators has been given.

Core System Requirements

Involved databases and tool systems:

- MATLAB Simulink
- Microsoft SharePoint
- MepSec System
- PTC Integrity System (Modeler)
- Salesforce
- SAP System (proCalc, ESZ)
- SOLIDWORKS Enterprise PDM
- MoBat DB
- PT DB

The core system must work in cooperation with many other systems and users. Proceedings in the testing environment must especially be traceable regarding the proper functionality of the following parts of the system:

- System security
- Flawless database processes
- Operationality of interfaces
- Massive number of users and sessions (> 5000 sessions) as well as parallel processes
- Process correctness

The functionality therefore shall provide means for adding further features to the core system. Furthermore, an automated interface check shall give insight on their proper functioning for data transmission. Also, a simulation of a large number of parallel user and system sessions has to be feasible.

Mobile System Requirements

Generally, the mobile system is easier to handle than the core system as it may be realized by testing the whole revised application on a mobile testing device. If new processes in communication with the core system are intended, a core system enhancement has to be done according to core system testing environment rules.

A.1.13 Standardized Forms Management

- Microsoft Word
- Microsoft SharePoint

There already exist forms which are used for documenting data which is coming from a customer as well as for the creation of offers and reports. Unfortunately, these do not always provide a consistent setup, yet. The system must offer means of providing forms, importing their data as well as filling and exporting them regarding a certain setup.
The creation of forms shall be configurable within the system by documents compiled in *Microsoft Word* (these may contain content within the document elements). The document elements and structure must be analyzed by the system. The creation responsible user must then be enabled to assign labels, e.g. INTRODUCTION, TASKS etc., to these elements. The structure and elements must be saved within a configuration file. Configuration files are to be stored to a distinct *Microsoft SharePoint* location.
Created configurations must be selectable within the system to be used as a template. The webservice furthermore must provide means of content selection for filling the template with information. Example: The selected template contains an element. This elements shall be filled with content. Therefore, the webservice offers the possibility for finding a task and to add it to the element. If a certain task is selected, its name, description and content is added to the template.
For the creation of projects reports, the same proceeding as stated in the example must be available, but objects of a project in progress are used. The access to this project data is dependent on the rights provided for respective users.

A.1.14 Battery Wiki Linkage

- Battery Wiki

Regarding the overall knowledge of batteries, there is an approach by the battery development skill team to collect knowledge on various types of battery developments and battery knowledge with respect to customers in the so called *Battery Wiki*. A Wiki helps to gather, manage and control corporate knowledge [Almeida and Rocha, 2011] and as such may maintain extensive explanations and comments for the MoBat-System's objects.
Furthermore, the system must include means of using the Wiki-provided search functionality directly from its own webservice and must provide direct links to search results in the Wiki. The reason for this is to reduce the usage of different user interfaces, which subjectively reduces the number of used software tools.

A.2 Usability Requirements

The field of usability requirements comprises a range which in total exceeds the scope of this work. Thus, a short mention of most crucial issues for this system's usability is given. A well-defined source for in-depths usability requirements which may be relevant for this system may be found in the **ISO 9241: Ergonomics of human-computer interaction**. The system development contractor is expected to have sound knowledge about this standard as well as must have experience in implementing it.
The main goal of these requirements is to reduce the click count. This shall ensure to use the system functionality without losing the general overview on the system functions. As this piece of information may only be described very inaccurately without actual users testing the system, an easy-to-extend user interface architecture shall be planned for implementation. The contractor who is in charge of the system implementation shall adapt the user interface to customer wishes by having an AVL-defined testing group evaluate the menu guidance and target achievement. Therefore, mock-ups must be provided to AVL.

A.2.1 Menu Hierarchy

The menu navigation shall be as flat as possible. The major user-involved functional system features are suggested to have the following hierarchical structure on the home page of the MoBat-System web service:

- Messaging (Message Assistance)
- Offer Estimation (Project Offer Assistance)
- Battery Development Data Management
  - Project Management
  - Simulation Data Management
  - Project Development Tracking
- Development Objects Management
  - Objectives
  - Requirements
  - Tasks
  - Functions
  - Components
- Standardized Forms Management
- Test Environment
- User Settings

For the mobile application implementation, no suggestion is given here and full rein is to be granted to the mobile system developer. As of the less complex features, a plain application can be expected and no further costs by extensive user interface testing and adaptions should be the case. Therefore, only a single test including an adaption option shall be performed by a product manager, as product managers are the persons who are mostly concerned with the application.

A.2.2 User Information

As the system provides a large number of functions and has many system objects defined with each of these owning very individual attributes, the system user has to be supported with help menus and explanations for these items. These user-assistance items must be available considering the following web service sections:

- *Menu navigation*: For each menu's items (function and sub-function) (e.g. in the menu introduced in Section A.2.1), an information field shall be implemented. While hoovering over the field with a cursor or by touch-activating it with a mobile device, an information box must be displayed. The box shall give a short listing on sub-functionality related to the menu item.
- *Functionality menu*: The functionality menus must provide an introduction to their possibilities. This must be covered by instructional overlays (coach marks) which are automatically triggered with the first usage of the functionality or may be requested by the user. Also, the overlay must be terminable and offer the possibility to continue where it was terminated or to start anew. The overlays must introduce groups of sub-functions and menu items. The overlay approach is chosen for this user information as it increases

a user's cognitive load with not too many instructions showed at once. The fact that the functionality may not be used while the overlay is active makes users memorize instructions. Also, only few overlays for most important interactions shall be shown with actions given one after another [Harley, 2014]. Also, the focus of few most important overlays shall lie on only a few interactions and these one after another.

- *Object attributes*: The attributes of the system objects introduced in Section A.1.1 must provide information fields like described for the here described *menu navigation* help. The information boxes related to these are to describe the meaning and an example for the related item as well as give indication about potential quantities and units.
- *Processing information*: A user who wants to commission a task to the system must be clarified about the potential and the likely processing time for the task as well as about rough processing steps for the task. Furthermore, an explanation on how results have to be interpreted must be given. This means, the user must have the information if an accurate calculation model or an estimation model is used, e.g. for offer estimation, or if clear system processes are applied e.g. change of a parameter value in a database. Requested workload which the system is occupied with is to be listed in a user-private menu including a presumable finish and residual time for processing.

A.2.3 System Feedback

Employees at AVL make use of the system if they significantly save time by using the system. Saving time includes short response times of the system as well as qualitatively convincing results delivered by the system. Regarding response times, three main time limits must be considered for web and web application performance [Nielsen, 1993]. These are determined by human perceptual abilities and are about the same since they were first stated by Miller and Card et al. [Miller, 1968] [Card et al., 1991].

- 0.1 second is about the limit for having the user feel that the system is reacting instantaneously, meaning that no special feedback is necessary except to display the result.
- 1.0 second is about the limit for the user's flow of thought to stay uninterrupted, even though the user will notice the delay. Normally, no special feedback is necessary during delays of more than 0.1 but less than 1.0 second. Nonetheless, the user loses the feeling of operating directly on the data.
- 10 seconds is about the limit for keeping the user's attention focused on the dialogue. For longer delays, users will want to perform other tasks while waiting for the computer to finish, so they should be given feedback indicating when the computer expects to be done. Feedback during the delay is especially important if the response time is likely to be highly variable, since users will then not know what to expect.

These bullet points must be considered for all user-interactive action of the MoBat-System, especially regarding processing information.

A.2.4 Internationality

AVL is an international company, so the company's common language was chosen to be English. For this reason and as cost may be avoided by a non-multilingual system, the MoBat-System user interface is to be implemented using English language (American English).
Regarding the system service and development, the following must be stated: Although the coding language does not imply a specific language to work, developers are urged to use English

variable naming, abbreviations and especially comments as far as it does not lead to confusing coding style. The premise shall always be that code can be interpreted as easy as possible by every person that has some experience as a programmer.

A.3 Performance Requirements

A good performance of the system is essential for being accepted and for supporting the battery-related processes effectively.

A.3.1 Session Handling

Session handling comes into play when actions are to be taken by one or more system users. The number of sessions indicates the amount of users accessing the system and as such is a measure of performance. This section introduces shortly how sessions are expected to work and Section A.3.2 gives explanations on capacity and utilization.
A user who accesses the system is assigned a unique session ID. Processing requests to the system are always related to a certain session ID and the user rights which are assigned to the user of the session.
System users who have not been authenticated are assigned a preliminary session ID. The core system's session handling module performs authentication of the user with the AVL authentication server if the user is not already registered as logged in to the system. After a successful authentication, the session is given a permanent ID, else the session is closed. An exception from authentication is made for a session initiated by the system in order to perform automated processing, e.g. if an e-mail must be processed.
Regarding session cancellation, the session handling keeps track on log-in times of a user and cancels the related session after a time limit of 10 hours. The time limit is chosen like this in order to not disturb system user with many log-in processes during a regular day of work. The session may also be canceled by request of the session user. If a cancellation is performed, the user and the corresponding session ID is deleted from active sessions. Thus a new log-in and authentication must be initiated by the user for further system usage.
The session handling is to support the distribution of core system computation time via scheduling and controlling the processes for each session. For the sake of completeness, it must be said that sessions may own an individual set of threads to run, but the consideration of system users sessions is a more detailed indicator regarding the expectations.

A.3.2 Utilization Rate

The MoBat-System must be built in a way that for the start, all Global Battery Competence Team members (around 40) may easily be handled. This number is expected to grow rapidly by means of enhancing it for further skill teams if the acceptance of the system means a breakthrough in the battery development. Thus, the session dimensioning of the system may not be fixed, but must be extendable to more sessions and utilized hardware. A session count of 500 is targeted for testing purposes.
A difficulty for the assessment of user session efforts may be constituted by the various functions offered by the system. Nonetheless, in the beginning it can be said that most of the system business will apply to requesting and data handling between different databases and tool systems. Features using machine learning approaches will be more consuming concerning CPU time of the system's servers and thus session capabilities must be adapted empirically.
Finally, the user must have the possibility to use the system without regard to the pending

requests. For the worst case, i.e. the system almost stalls as too many tasks are being handled, the user must be informed.

A.4 System Interfaces

The MoBat-System needs human-system interfacing (user interfaces) in order to provide the possibility of user interaction and system-system interfacing which provides communication between the MoBat-System and various other systems.

A.4.1 System-System Interfaces

As already introduced, the system-system interfacing shall be taken care of by means of the AVL tool-network. All interfaces shall be provided for bidirectional usage. The interfacing concerns connectivity to databases, tool systems and other servers. The details on interfacing issues must be discussed with the AVL employees that are engaged with AVL's *Integrated and Open Development Platform* (IODP) which addresses consistency, interoperability and communication challenges that enable a modern model-based development processes [Valnion, 2016].
Moreover, the MoBat-System-sided interface parts must be provided with the functionality to detect if an interface might be outdated and not be able to send or receive all data anymore (incompatibility detection). The mechanism shall be presented to AVL before implementation.

A.4.2 User Interface

The user human-system interfacing covers all kinds of user interfaces (UI) of the MoBat-System. The functional requirements of the system already contained necessary sub-functions. In order to keep costs low for the interface development, only a low number of requirements and constraints shall be introduced here and mostly free rein is granted to the system developer. The low cost factor mostly refers to the development contractor having experience in UI development. Proof of the contractor's experience is advised to be claimed.
Additionally, a presentation of utilizable content of a previously conducted Master thesis will be presented.
The last subsection to user interfaces explains a suggestion on the depiction of a task management menu. This suggestion was developed in a tête-à-tête meeting with a future system user and thus gives a qualitative guideline on user expectations.

General Requirements

First, the general requirements of the user interface go hand in hand with the *Usability Requirements* stated in Section A.2. Summarizing, a flat menu hierarchy and user guidance must be implemented. Nonetheless, the user interface content shall not be too crowded. The development of UIs shall be supported by an evaluation of mockups delivered by the contractor. An evaluation on the basis of mockups shall be performed two times. A final third evaluation on functionality must be performed when the system development is finished. For all evaluations, a collection of concrete change requests must be issued by AVL and these must have influence in the user interface change process.
Moreover, for AVL's software, a color coding scheme according to the corporate policy must be considered relating to certain functionality. As such, the elements of the user interfaces must be configurable in color assignment. Master palettes for these assignments must be additionally installable to the system.

Ideas on the User Interface

The idea of the user interface depiction was already developed in a Master thesis at AVL about visualization of information in product development with regards to MBSE [Müller, 2016]. The approach resulted in a mockup of a user interface of which some ideas may be well adaptable to the presented functional opportunities of the system. The following paragraphs sketch parts of these ideas plus own adaptions which would be useful for this system:

Home Page The home page of the user interface shall be accessible if a login to the system by a user was successful. The home page holds the major navigation menu on functionality on its left side (compare with Section A.2.1). Further content on the home page shall be the upcoming deadlines which a user is involved with, e.g. a deliverable of a certain project and its deadline. Furthermore, tiles shall be shown with regards to involvements, i.e. a tile for every project involvement of the user, or a bar in which system shortcuts may be remembered.

Project Overview Hyperbolic trees which model the dependencies between task, functions and components of a project shall give a quick orientation in a project. The single leaves (system objects) of the tree chart are clickable by which the respective system object is centered and information on it (object attributes) is presented in an information field next to the chart.
Furthermore, objectives, requirements and tasks with currently set values may be watched in a separate view. As described in the functional requirements, these attributes can be added and edited for the system.
Eventually, the progress of projects regarding tasks may be depicted by Gantt charts.

Task Menu

The sketch in Figure A.1 for the user interface of the general task management was created in discussions on the requirements of the MoBat-System. The task UI shall provide an overview on the task and its embedding between other tasks whilst managing to give an overview on the task itself and its surroundings.
The sketch shows the elements of the general task management menu. The elements are described including some functionality. The applicability of functionality thereby always depends on the access rights owned by the user.

1. There is the search functionality to retrieve a certain task. The search options shall be usable separately from each other. The sort order of search results is considered to sort chronologically regarding alphanumerics.

2. A selected task from the results box is displayed in the box right of the task search functionality. The selected task is shown centered. General knowledge and explanations to this task are presented in the task box. If the task search management within a project is used, the project-related information shall be shown here additionally (see Section A.1.1). The task attribute boxes must not show their overall contents if there is not enough space and full information must be retrievable by clicking a box.

3. Tasks that can be connected directly and are to be accomplished beforehand are shown left of the center task. These may deliver either fully or partly necessary information for the selected task. A task selection by double-click centers it and shows its information environment.

4. Subsequent task options that may need information directly from the center task are shown right of it. A task selection by double-click centers it and shows its information environment.

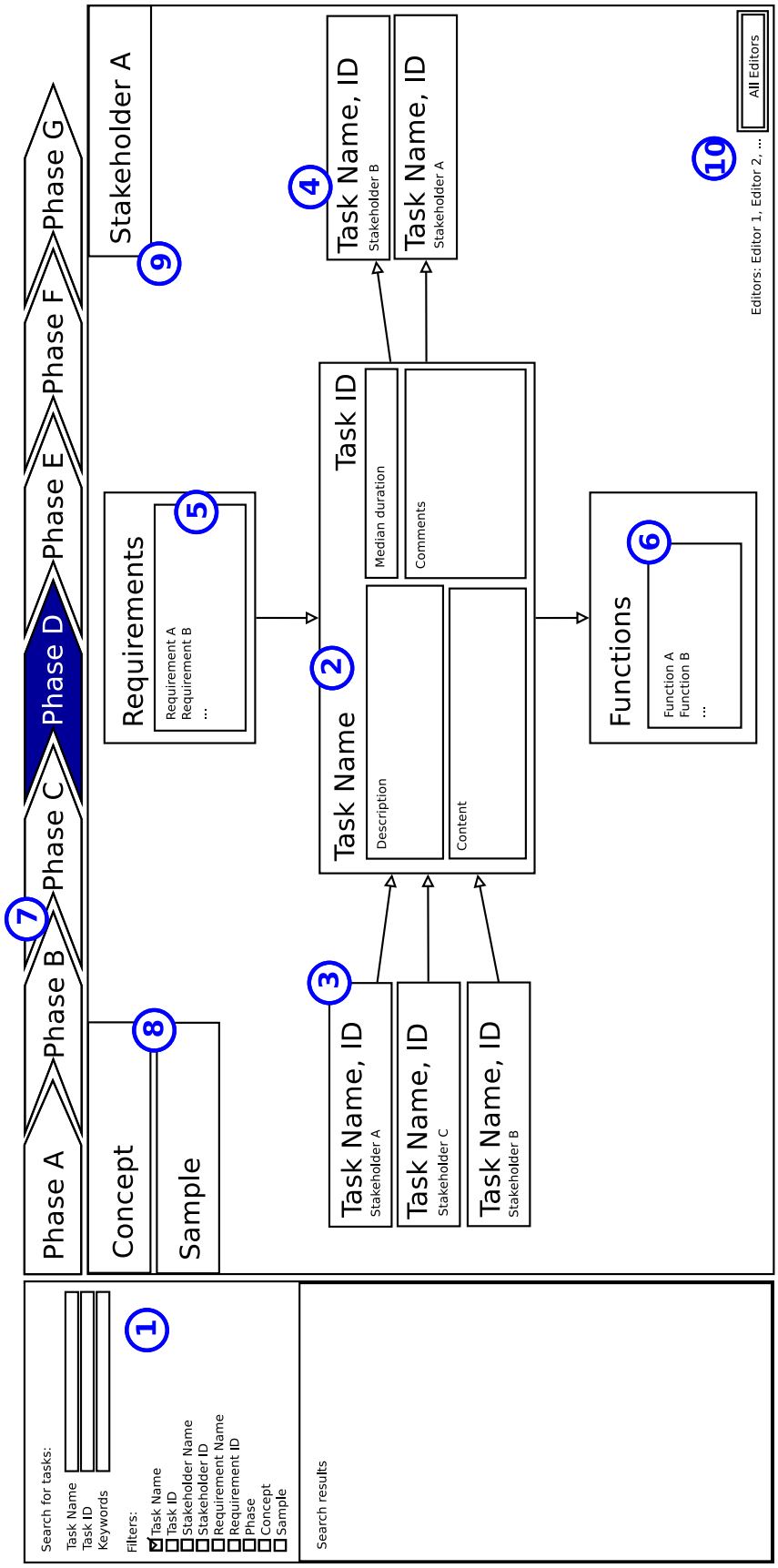


Figure A.1: Sketch of the task management user interface.

5. Above the center task, requirements from which this task may be derived are given. Not all potential requirements must be shown at once, but can be displayed by a selection of the requirements box. A requirement selection by double-click leads to the requirements management menu.

6. Functions that may stem from the center task are added below. Not all potential functions must be shown at once, but can be displayed by a selection of the functions box. A function selection by double-click leads to the function management menu.

7. The phase indicator of the task. Editable by double click.

8. The concept and the phase which the task is assigned to.

9. The stakeholder who is assigned to the task.

10. The editors of the task attributes. A short overview on last editors is given. The *All editors* button leads to all editors of the task including the changes they made to the task.

For task management within projects, all preceding tasks, subsequent tasks, requirements and functions are shown, but the ones that are involved in the project are shown at the top positions.

A.5 Information Management

In order to ensure proper functionality of the system, some criteria regarding information handling have to be met. These may be user-related or related to system processing.

A.5.1 User Attribution

Especially regarding content to be dealt with in the system, an attribution of general user involvement must be enabled. The used model describes four levels of possible involvement:

1. *R* - *Responsible* (needed)
   Owner of an activity and decision maker.

2. *A* - *Accountable* (needed)
   Person who must be informed by and who can veto the responsible.

3. *S* - *Support* (optional)
   Person who is informed about an activity and supports and contributes to it.

4. *I* - *Informed* (optional)
   Person who must be informed about an activity.

This model is derived from the RACI standard [Costello, 2012], where *C* - *Consulted* is replaced by the support participation. These attributions are on the one hand needed for distributing content rights to users of the system and on the other hand identify the involvement of users by AVL. Example: If a task is marked as responsible by an external person who is not AVL staff, the lead role for this task processing is occupied by the external company. This model is sometimes quoted as the RASI model within AVL. According to rights distribution, this work may state terms like *AS users* which indicates that users assigned the clearances accountable or support are meant.

A.5.2 Data Anonymization

All kinds of data that are imported into the system and not stored to a *Microsoft SharePoint* directory are stored within the system's databases or the databases which are connected with the system. Data which is stored to the MoBat DB also keeps track on user data regarding project involvement and system object changes. The access to user data must not be possible for any functionality with the following exceptions:

- Change history of system objects.
- Change history in project-related data within a not finished project.
- Automated project development tracking

Anonymization rules are also stated for customer assignments in the system. The customer regarding a project is visible to every system user, but access to other projects of the customer is not given generally. The access to a customer's project for usage of its data in other projects (knowledge sharing) may be denied by an opt-out decision, like stated in Section A.1.6. Knowledge sharing information shall not be relatable to certain customers if used for estimations or other proceedings in projects. An exception for this is data which stems from other projects of this customer.

A.5.3 Backups

Automated backups of the system - functionality as well as data - shall be contrivable on physically different servers than the ones which the productive and the testing system are installed on or productive data is stored on. Further explanations on backup functionality are given in Section A.8.

A.6 System Security

This section gives insight mainly on security requirements related to system usage and rights, but also on hardware-sided prerequisites for the running system.

A.6.1 System Site

The MoBat-System in software and hardware has to be stored in the headquarter facility of AVL in Graz, Austria[17]. An exception for this is the initial system development which ends with the first deployment of the testing system on AVL servers. All further developments are to be done in the testing system via an appropriately secured online access.
The system software and data must be provided failure safety by means of an appropriate RAID storage virtualization. The RAID level to be applied must be concluded in agreement with the AVL IT service.

A.6.2 General System Access

In general, system access is granted to all AVL employees who were authorized with the MepSec system. Access to the system must be controlled through authentication with AVL authentication servers which is initiated by users via a log-in procedure provided by the MoBat-System. User rights relate to the contents and functionality of the MoBat-System. These user rights are

[17] Address: AVL LIST GmbH, Hans-List-Platz 1, 8020 Graz, Austria.

assignable by users with respective appointment rights. The relevant user hierarchy is described in Section A.6.3.

For the mobile application, no distinct log-in procedure or other authentication is needed. The pass code to unlock the phone is considered sufficient. Access and content rights to the application functionality are granted if the application has been installed on a company's mobile device. An installation is only possible by the AVL IT service if the user was authorized in the MepSec system.

Securing the core system functionality, such that it is not used by unauthorized mail users must be considered. This must be ensured by usage of secure codes which are assigned to single system users. The secure code is to be delivered in the header of mails that are sent to the system. The system needs to check the secure code on its validity and user assignment before initiating further processing.

A.6.3 User Rights

The user rights distribution is essential with regards to preventing unauthorized insight in operational data and customer data. Examples are pointed out in the following lines for better understanding. Moreover, Figure A.2 gives an overview on the rights assignment options.

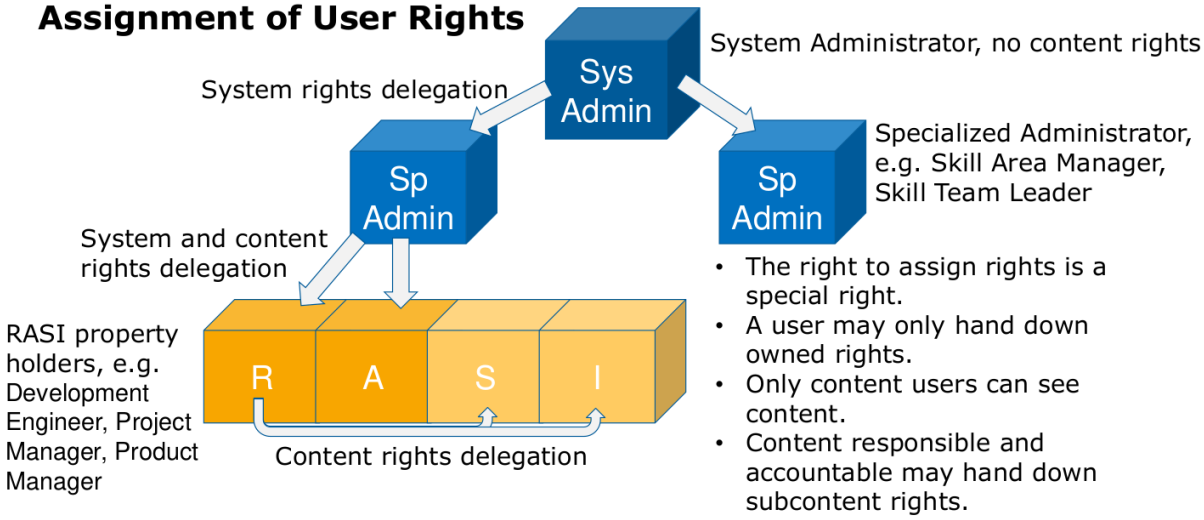


Figure A.2: Assignment of user rights.

Appointment Rights

The system must handle a number of different user groups, which own different access and administrative rights. Some users have the right to appoint rights to other users. The rights to appoint a user for a user group are given in Table A.9. The table only shows a general hierarchy of rights assignments. Users may only assign rights to users that they own themselves.

The user groups are divided into administrative users and content users. While administrative users only have system administrative rights, content users have the right to access data of the productive system.

The classification of content users follows the SARI model (see Section A.5.1) which provides a framework for an assignment of responsibilities to roles and functions within the system functionality.

User	can appoint					
	System Administrator	**Specialized Administrator**	**Content Responsible**	**Content Accountable**	**Content Support**	**Content Informed**
System Administrator	yes	yes	no	no	no	no
Specialized Administrator	no	yes	yes	yes	no	yes
Content Responsible	no	no	yes	yes	yes	yes
Content Accountable	no	no	no	yes	no	no
Content Support	no	no	no	no	no	no
Content Informed	no	no	no	no	no	no

Table A.9: User groups and corresponding appointment rights.

Administrative Rights

Administrative rights refer to enabling a user to access system functions and their sub-functions. System administrators are the general administrators of the system who are responsible for its functionality, but not the content which is created by the system's means. System administrators may grant the right to manage the access rights for system functions and sub-functions to specialized administrators. These rights relate to the essential functional capabilities of the system.
The term specialized administrator is applied, because the administrator shall be associated to a certain department or operational unit within AVL, for which the systems functionality must be distributed. Example: The lead of the department for powertrain development (PTE) is appointed as specialized administrator. It is in their duty to assign user rights to at least the leads of the department's skill teams. These may again be specialized administrators with nominator's rights and content rights, e.g. the right to be a content responsible person regarding battery projects.

Content Rights

Content rights on the one hand refer to system functionality and sub-functionality that may be accessed, e.g. task object management or project handling. On the other hand, these rights concern the content within a certain functionality, e.g. tasks in a project or the attributes of a task assigned to a project. The attribution of content rights according to Section A.5.1 may be interpreted like this:

1. *R - Responsible* (needed)
   The content owner, may see, edit or create any content regarding this system functionality or sub-functionality. Example: This user is enabled to be responsible for a certain project.

2. *A - Accountable* (needed)
   Person who is accountable and who must be informed by and can veto the responsible person. Example: The battery development skill team leader is accountable for a battery project.

3. *S - Support* (optional)
   Person who is informed about an activity and supports and contributes to it. Example: A developer within a project.

4. *I - Informed* (optional)
   Person who must be informed about an activity. Example: A department lead who wants to have insight in the progress of a certain project.

The rights to access the content can be granted by a specialized administrator who is enabled to use the corresponding functions and sub-functions or by the content responsible. A user can only be enabled to access content that is accessible to the access granting user.
Generally, the content responsible person and the content accountable person always have to approve of a modification of content. As this would be hindering in some situations, the content accountable person may delegate the approval option to the content responsible person. Furthermore, the content responsible person and the content accountable person may delegate modification rights to the content support personal.
Furthermore, a feature of content rights management shall be that a content responsible person may lock and unlock content from modification.
Eventually, regarding projects, a default scheme of modification rights must be considered in order to avoid setting user rights for every project from scratch. Table A.8 in Section A.1.6 shows this default setting.

A.6.4 Communications Encryption

Any communication in between servers or clients of the system as well as communication to external entities must be encrypted. The encryption level regarding functional characteristics must at least equate to the Transport layer Security (TLS) protocol.

A.6.5 Log History

In order to enable tracking of system usage regarding data alteration and transition as well as functions of the system that are used therefore, logging must be implemented for all system functions. The logs must contain information on session and respective user and their requests to the system and corresponding data which was transferred. All proceedings must be recorded and stored with regards to core system time.
A log must also be kept with respect to single system attribute and parameter changes of system objects. This shall not be a log to be evaluated in case of system error, but must be a history available for minor recovery by management and development users. Users thus must be enabled to see the last changes of and in system objects with respect to the date. A mapping between object changes and responsible persons must not be feasible.

A.6.6 Backups

Snapshots of the system shall be made. They are to include all states of functionality as well as the data involved in the MoBat-System. Some of the involved proprietary tool systems and database are already equipped with backup functionality, so only databases and functionality especially created for this system must be endowed with this possibility. Further information on system mode and operations necessities of a backup are given in Section A.7 and in Section A.8.

A.6.7 Session

A session comes into play when actions are to be taken by a system user. Only users that have been successfully authenticated by the system via the AVL authentication server are assigned the right to use the system functions according to their user rights. The communication process between authenticated or non-authenticated user and the system is called a session. A session must be uniquely identifiable and assignable to a user in order to ensure system protection and data integrity. System operations are always performed related to a session.

A.7 System Modes and States

The system generally exists in three modes in core and mobile system alike, the productive, the testing mode and the backup mode.

The two former ones may also be regarded as two different states as their systems shall be mostly separated from each other, i.e. different data and partly different functionality. The connection between these states is the possibility of porting program parts from the testing to the productive system. Although, most functions are equal in the states (except for the ones being tested), the systems must be separated like this in order to avoid the risk of producing and distributing wrong data in the productive mode. Generally, the productive mode only features system functions which were verified in the testing mode. Section A.8 gives insight about formal human criteria that must be given to initiate restoring the system with a backup.

The third mode is the backup mode, which shall particularly ensure the system safety. Snapshots of a system must be done on a regular basis and are a preventive measure in case of

unauthorized access or massive system failure and the associated chance of compromise of data and functionality.

A.7.1 Productive Mode

The productive mode works with all functionality which has been approved in testing mode. For the functionality to work properly, all productive data, that was produced by use of the system must be available. The data therefore must be maintained carefully by system users as this metaphorically states the fuel of the MoBat-System. Also, if defects in functionality were virtually ruled out, there may still be a chance of any unwanted behavior. This may be fixed by a partly backup, but the functionality may not just be shut down in order to prevent further problems.

A.7.2 Testing Mode

It was already described in Section A.1.12 that during the testing mode of the MoBat-System, special attention must be paid to the following issues:

- System security
- Flawless database processes
- Operationality of interfaces
- Massive number of users and sessions as well as parallel processes
- Process correctness

The testing mode has to provide means of documenting the verification of their functionality and to easily retrieve this information. Proper documentation helps testing staff to quickly port new software developments to productive mode. E.g. if a first review already was performed and handed back for rework, the testing status of some parts of the development already exists and must not be repeated.
The testing system moreover has to provide means to implement new functionality for the system. This may be achieved by a porting functionality or direct implementation in the test system.

A.7.3 Backup Mode

The backup mode can be divided in to sub-modes. First, the backup creation which is intended to make a snapshot of the system. This snapshot shall include all functionality as well as the data involved. The exact contents of snapshots (database, system functionality) and the execution data and time shall be configurable. Some of the involved proprietary tool systems and database are already equipped with backup functionality, as such databases and functionality especially created for this system must be endowed with this possibility.
The backup creation shall be executable regularly and the interval between backups as well as the time of day must be configurable. Also, backups must be created automatically or by system administrator request. A number of backups which is configurable must be held of the system, recognizable by timestamp and mode to be restored.
Moreover, created backups must be maintainable, i.e. loading or deletion of backups must be possible by an authorized system administrator. The deletion must be especially supported in order to simplify the selection of mostly redundant backups. It may be the case that two backups were taken in a short interval and potentially not the latter one is the better backup.

Thus, a differentiation support must be provided which delivers an overview[18] on two backups' differences in order to meet a decision.
In case of severe issues in productive or testing system, the system must provide the functionality to restore the system to a backup of a selectable state. The productive and the testing system must be restorable separately from each other as well as their databases. A backup must be applicable any time. Moreover a division in several steps of loading the backup to the system must be enabled in order to avoid a potentially high system activity during employees' core working hours. Section A.8 gives insight about formal human criteria that must be given to initiate restoring the system with a backup.
In order to avoid a bad design, the designated system maintenance group must be consulted for discussing the way of operating the system and ensuring functional compatibility with AVL standards.

A.8 System Operations

A number of system operations may be seen as crucial for long periods of unimpaired executability. These touch the human factor as well as maintainability and reliability of the system.

A.8.1 Human System Integration Requirements

The important partly human-interacted operations refer to the system modes and states which were introduced in Section A.7. The human proceeding for these must be carefully thought of as well as must be performed following a strict protocol. It is also highly recommended that these operations are checked in every step and are done in at least a four-eyes principle.

Testing and Productive Mode

The general proceeding already was similarly proposed regarding workflow management by Crnkovic et al. [Crnkovic et al., 2003]. Figure A.3 shows an adapted graph which exemplary states a workflow for change approval. All system parts that are worked on and tested are stored in the work in progress (WIP) vault. The designer is communicated work orders. The results of these work orders are reviewed when the work is assumed to be finished. The change control board (or change review board) takes care of comments by manufacturing engineering testers (here: various users of the system) and production and inventory control (system maintenance) and gives an order for rework if shortcomings were found. If the system part is approved and potential configurations have been done, it is stored in the release vault (productive system), which makes the new system functionality generally available.

Backup Mode

For backup creations, only a system administrator's configuration on backup intervals and the daytime shall be needed. Furthermore, the configuration shall feature the possibility of defining any location within the AVL network for storage of the system. The system shall perform backups with regards to these times fully automated. A backup handler shall keep records on backups regarding location and time stamps and deliver this information to the user if requested. The deletion of largely redundant backups must follow a more stringent protocol. The differentiation support between backups must be considered in order to compare if a backup on a specific

[18] A clear definition is not easy to grasp here, but a recommendation would be to use distinct comments of system functionality implementations.

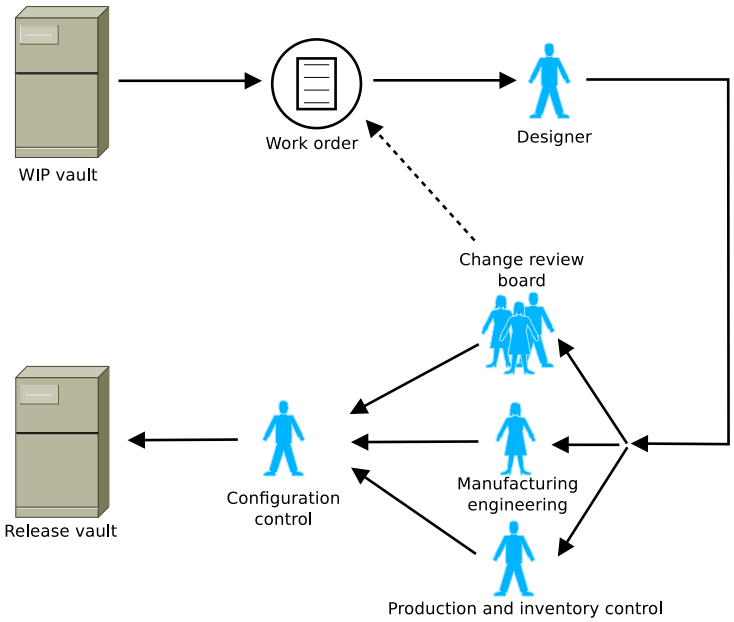


Figure A.3: Workflow for system change approval (adapted from [Crnkovic et al., 2003]).

part of the system is still needed or not. The suggestions on backup deletion must not only be accepted by a change review board of system administrators, but must also be confirmed by members of the board of AVL who are entrusted with this technological matter.

A.8.2 Reliability

The system is initially intended to drive the whole development process at the battery development department at AVL, but still serves the idea to be extendable to support all AVL development processes. This influences the necessary reliability parameters *Functional Service Safety* and *High Availability* of the system. These are partly influencing each other in an often not predictable way, thus the following considerations on these criteria shall be incorporated into the system design.

Functional Safety

The term relates to the system's faultless performance. It must be considered that hardware failures, e.g. of servers, networks, etc., or operator may errors occur which must be handled by the functional safety management of the system [Smith and Simpson, 2004].
For the failure of hardware, a safe shutdown of system functionality must be feasible. These shutdown operations must be performed in a way that functionality and data which is being processed at the moment of failure may be recovered. The time which AVL's uninterruptible power supply support (UPS) is granting must be included in the system design. Information regarding this matter must be obtained from AVL's IT service.
Regarding operator errors, the system must be implemented in a way that these may mostly be ruled out. In cases of unalterable, hardly recoverable or non-recoverable errors, backup operations must be initiated. As it is hard to grasp a qualitative number for describing this characteristic's success rate, it must be coupled to the high availability characteristics in the next paragraph, i.e. the qualitative measure of system availability.

High Availability

Regarding the high availability rate of the system, it must be considered that not all downtime is created equal. Outages which drive away users are more costly than outages which are a minor bother for them. Nonetheless outages that are minorly bothering are more costly to the company than outages which were not detected [Marcus and Stern, 2003]. Furthermore, the two large user groups of the system - productive system users and system maintenance users - have different views on the system availability. In order to satisfy their needs, the following considerations have to influence the MoBat-System design.

High availability rates for the productive user group must fulfill at least 99.5% regarding planned productive system uptime. The planned productive system uptime thereby implies that there are planned downtimes to be considered which reduce the maximum system uptime. These downtimes are maintenance periods in which productive system changes and enhancements are migrated from the testing system. Unplanned downtimes (0.5% of planned availability) thus relate to issues which relate to failures like described in the functional safety section or to unpostponable system changes, e.g. loading of backups as of system compromising or undetected severe deficiencies of functionality.
A planned downtime of 7 days could be considered operable for this system. This is reasoned from less development operations being performed from end of December (around Christmas) till the first week of January, which enables updates further system maintenance. According to pertinent high availability tables, this relates to an average yearly productive system availability of 98%. As AVL is an international company, this may not be the case in every country, but is considered as battery development is mainly done in the headquarters in Graz, Austria.

The testing system must offer less availability rates for development test users as it must be expected that changes are implemented continuously. Nonetheless, testing users should not be driven away by an constantly unavailable testing system. So these users should at least be able to access the system part for 95% of days per year. This leads to a rate around 5% which may be spent for testing system maintenance and migration operations.

All rates were determined by the tradeoff between keeping costs low - for higher levels of availability, costs increase very rapidly [Marcus and Stern, 2003] - and needed operability of system components. For initial versions, especially while no testing system exists, these rates may be regarded as guide values, but they shall come true within the first year after the first deployment for the MoBat-System.

A.9 System Life Cycle Sustainment

AVL is the single owner and proprietor of all MoBat-System components that the contractor is commissioned to develop. This relates to ideas and intentions stated for the system, the whole implementation and the data which must be provided for developing it. The next sub-section holds different information about what must be considered for the system's lifecycle.

A.9.1 Development Phases

The development of the system is divided into the three phases: design, implementation and review. These may be passed through repeatedly in case of development shortages detected in the latter phase.

Functional and Technical Design

Every detail of the system design must be put down to a design document. The design is dependent on the specification of system requirements which comes from AVL. This encompasses software and hardware design alike. Design documents that are edited regarding existing functionality must mark the document's sections which mean a change of functionality in comparison to the previous design. This pertains both, system components that are already implemented and ones that are not implemented, yet. Enhancements to the initial design are to be compiled in an extra document if no already related design document exists.
The design documents must be identifiable by system functionality, design version and the date proposed.

Implementation

The implementation of system parts may only be done after their definition in design documents. This requirement shall ensure an elaborate design and the traceability of system developments. The implementation phase ends with the delivery of the system parts by deployment in the testing system.
The only exception pertains for deployments while no testing system is existent. For these cases, the deployment is directly added to the productive system which must be interrupted from usage during deployment and the review phase. The testing system must be delivered as soon as possible, as the interruption of the productive system could pose a serious outage of battery development. In order to reduce the stress and effects of this case, a deployment during AVL's non-core working hours according to the central European time zone (CET) must be considered.

Review

The review phase starts when new system parts have been deployed to the MoBat-System. Tests on system functionality must be evaluated by the AVL change review board regarding testing comments by test users and system maintenance testers.
In case of deficiencies, the system producer is ordered to rework the current system parts design and the implementation. The deficiencies have to be collected and documented by AVL testers and the change review board and passed to the producer as a reclamation document.
The circle of development phases ends for a functionality development if the acceptance test was rated successful. Nonetheless, hidden functional errors that were not detected may be the case. The producer must be obliged to patch such errors without further charges as working functionality is a core requirement of contracts.

A.9.2 Versioning

The software system must be relatable to a release versioning. The versioning shall follow a composition based on the explanations on so called *Semantic Versioning 2.0.0* by Preston Werner [Preston-Werner, 2013] with only some minor changes. The here relevant excerpts for versioning can be found in Appendix C. The composition is to be divided into the parts MAJOR.MINOR.PATCH+METADATA. The general meaning of the first three positions of the versioning policy is:

1. MAJOR version if incompatible API changes are done,
2. MINOR version if functionality is added which is backwards-compatible
3. PATCH version if bug fixes which are backwards-compatible are made.

The METADATA part serves the purpose of being a version control system. This number must be able to identify sources of the compilation precisely. This way, sources can be retrieved in case of error [Westfall, 2009].

A.9.3 Operational Support

The software system must be migrated to AVL servers when having reached a deployable stage. The migration shall be performed by the AVL IT service in coordination with the system developer. Maintenance support must not only be given for the first system migration, but must generally be offered by the system developer especially with regards to potentially emerging problems.
Moreover, the system provider must inform AVL about the hardware dimensioning, dependencies which are needed for the system to be installed, maintained and executable as well as about constraints introduced by the system, e.g. inference with other systems or RAM capacities needed.

A.9.4 Training

Cadre training for system usage must considered by AVL as well as by the system producer. Producer-sided training thereby has to help to familiarize a initial cadre of users with the system functionality. The AVL-sided MoBat-project responsible persons are intended to assist this cadre training, because they should have extensively dealt with the functional opportunities of the system.
Furthermore, two kinds of cadres must be distinguished, the battery development cadre and the system maintenance cadre.
Cadres that have been trained, have to delete the information on system maintenance and usage regarding further workshops with AVL staff in order of the system knowledge to spread within the AVL departments.

A.9.5 Data Provision

Data needed for the system provider's development purposes are to be delivered by AVL. Also, data which is identified during the system production and specifically related to the system functionality is property of AVL. The data ownership stays at AVL at all time and data must not be passed on to any third party without written approval by AVL.

A.10 Policies, Regulations and Ethics

AVL and the workers' representation have high standards of not only work ethics, but ethics as a whole. Nonetheless, restricted budget is a necessary evil which must be served and thus not all ethical correct ideas my be implemented to the system from scratch. These and inevitable policies must be respected and are given in the next subsections.

A.10.1 Support of Visually Impaired People

A pressing issue is the barrier-free accessibility of the system, especially regarding blind and visually impaired people. There are Braille keyboards which make reading content possible, but still the system content must be prepared for better application. An extra budget for serving

this purpose is recommended, not only as of the actual need, but as of AVL taking up a position which it may be respected for.

A.10.2 Labor Policies and Personnel Information

The workers' representation pays high regard to employees' rights and the protection of their personal information. A borderline functionality, especially regarding these rights, was introduced in Section A.1.6 on Project Development Tracking. It is especially advised here to discuss this matter with the workers' representation in order to meet a mutual agreement before giving approval for its development. This agreement is essential as the sheer functionality would offer the possibility of directly tracking the working behavior of an employee which must be clearly ruled out by system abstractions, protections and regulations.
These concerns of course also apply to issues which may come up in the future and must be handled with the same care as the mentioned one. The employees' rights to be considered for global use must refer to the Austrian legal basis, which is to be taken as of its internationally high standards.

A.10.3 Multilingual Support

Language is an essential part for the applicability of a system. On the one hand, this refers to written and spoken language. Unfortunately, not every of these languages may be supported by the system, but as the system usage is mainly associated with highly educated developers and English is a standard language in globalized business, the necessity is only minor in the early stages of the system development. Nonetheless, there are tools especially for webservices which support the translation of web interfaces and as such must be included with the project advancing.

B

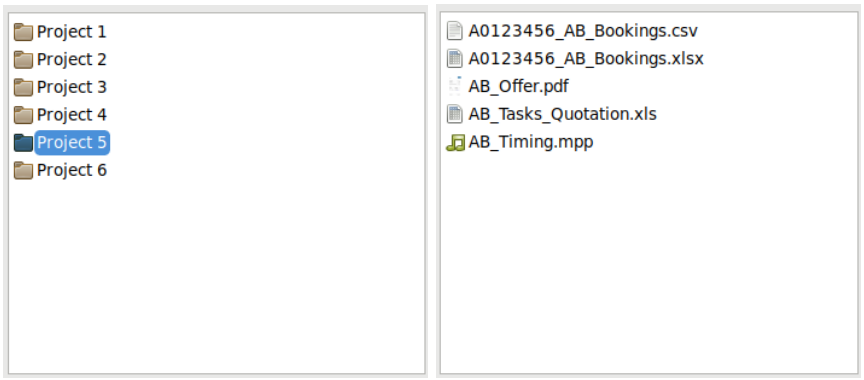
Offer Assistance Prototype Manual

B.1 Maintenance of Data

The application is in need of a certain structure of data directories right now. This also provides a clear filing structure for inputs and outputs of the application and for the results of major application levels and algorithms.

B.1.1 Directory Structure

A new directory needs to be created if a completely new estimation model shall be built. We call this directory the base directory. This directory only contains data folders which include data of finished projects, like depicted in Figure B.1(a). The folders can be named any way considered best.
Each of these folders is to contain files according to a single project. Of these files, only a single one is of utmost importance at the moment: the bookings *Excel* file with extension *xlsx*, which contains the single bookings of employees for the project. It must be the single file to contain the term *Bookings* (case sensitive) anywhere in its file name. It is also sensible to store other files related to the project in the respective folder, e.g. the offer, or the cost quotation (ProCalc file) or project timetable, in case of a possible enhancement of the program which makes further automation possible. An example of folder contents is depicted in Figure B.1(b).



(a) New application directory. (b) Project in application directory.

Figure B.1: Basic directory structure for offer assistance application.

After a successful run of the application, a further folder and several files appear in the base folder. The new folder should be named *Training-Results*. When entering this folder, a structure of numbered folders is to appear. Section B.1.2 holds information about all new files.

B.1.2 About the Data

A quick overview on the different files in the base directory and the training results directory is given. For explaining all files that may appear, an exemplary directory with contains a trained estimator is being used.

Base Folder

The sub-folders contained in here were already mentioned and the *Training-Results* folder is described in the next paragraph. The remaining files are briefly described here. An example of files and folders is shown in Figure B.2.

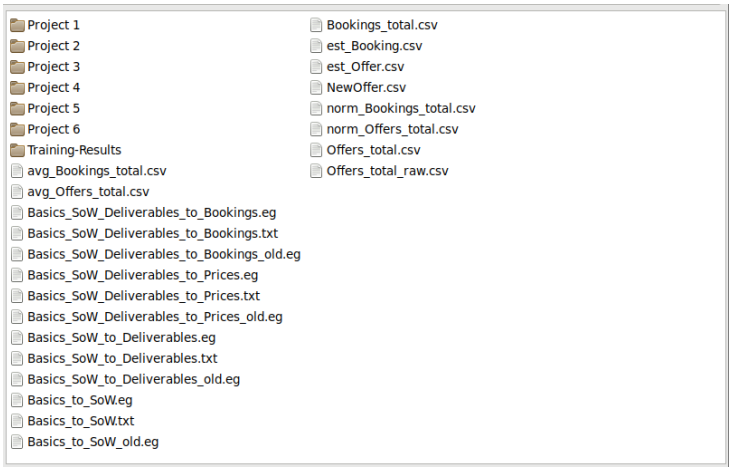


Figure B.2: Example for the base folder regarding an estimator model that has been used to estimate bookings for a future project.

The explanation is done in a chronological way, considering the usage and generation of files. The offer learning data is fed to the program. When finished, the *Offers_total_raw* file is built additionally with the *Offers_total* file which contains conversion of textual components of former file to numerals. The *Bookings_total* file also appears, containing accumulated cost center bookings to projects.
Before the actual training of different network models begins, normalization of the *total* data files is performed leading to files with the name prefix *norm_* as well as the calculation of an average of these stored in files with the name prefix *avg_*.
After the training of network models, *txt*-files comprising the mean square errors (MSE) of training validation regarding different trained models are created. There are several of these files as several computations from inputs to outputs are performed. The model scoring best by MSE is selected and stored in the equally named *eg*-file. For the reason that already a trained estimator already existed, the old *eg*-files are appended the suffix *_old*.
Finally, when using the estimator, data is entered to the program with the application interpretable format provided by the *NewOffer* file. After estimating the unknown parameters of the offer and corresponding bookings, files with the prefix *est_* are generated, including the estimated data.

Training Results Folder

The *Training-Results* folder contains numbered folders. Each of the numbered *txt*-files in a numbered folder keeps track of the learning results of different machine learning methods (see Figure B.3). These files also contain information on the full parameter settings which the networks were set up with. Equally named *eg*-files store the final state of respective networks.

The method names are given in the file names within respective folders. Those names are accompanied by numbers at the end of the file names, which indicate either the number of hidden layer neurons of a feedforward network or the number of RBF neurons used for the other two methods.
The first number after the term *Fold* relates to the k-fold distribution of datasets. With the low number of datasets, every dataset represents a fold. For every run of the application with different number of hidden layer neurons and a different machine learning method, one dataset is spared to verify the performance of the network settings and shown here. The subsequent number only exists for a more neatly overview in a folder sorting by name.
It is of best interest to not change these files. An exception may be the case when everything is retrained for obtaining estimation models. In this case, the results folders may be deleted.

Figure B.3: Example for training results folder.

B.2 Using the Offer Estimator

The offer estimator may be used if a training run has been performed in a base directory at least once. After starting the application, the window in Figure B.4 appears. The selection of the *Estimate offer* option sets the application mode. A base folder which contains training results has to be chosen via the section *Training set directory*.
A selection of booking data is not possible, as this is to be estimated by the program.
Finally only basic offer data has to be provided to the program. This can be done by importing a file or by manually providing the basic input information. Note that only the basic information is needed as the rest of the data is estimated step by step. A *NewOffer* file is created, which contains given information.
In the course of bookings and offer price estimation by the application, missing information is added. When the estimation is finished, files with prefix *est_* are created, containing the desired data.

B.3 Training the Offer Estimator

For the training of an offer estimator, a base folder structure like it was shown in Figure B.1 must be given with each folder containing at least the booking files described in Section B.1.1. When starting the application, the window depicted in Figure B.4 appears.

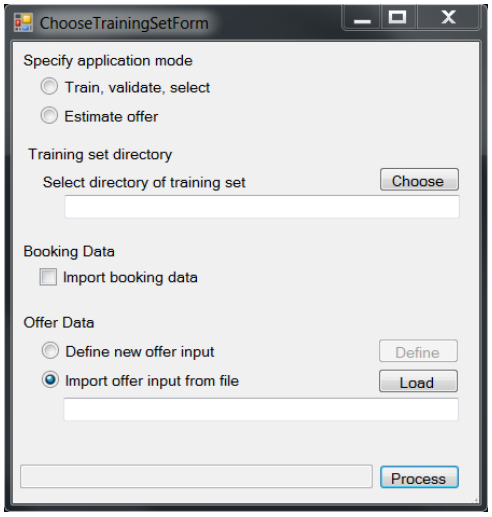
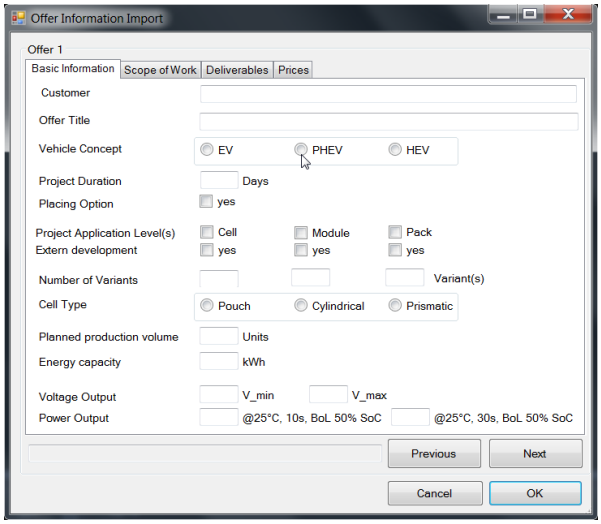


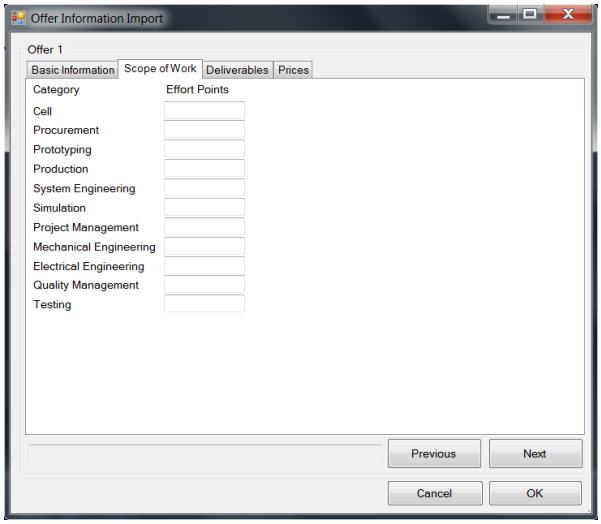
Figure B.4: Start form of the application.

Choose *Train, validate, select* as the application mode. Then specify the base directory for the training set directory.
Check the *Import booking data* checkbox if you have not imported respective booking data, yet. If you already have imported booking data, a *Bookings_total* file should be visible in your base folder. The booking data is imported chronologically according to folder names where they can be found.
Regarding the offer information, one out of two options is to choose. The easier way is to import an already existing offer information file, whereby an *Offer_total_raw* file should be visible in your base folder. Be aware of the fact that file entries must be equally ordered like the bookings information. Each line corresponds to a dataset. The costlier way of feeding the offer data to the program, is to manually enter it.
The form for entering the offer data appears when selecting *Define new offer input* and the then activated button *Define*. Figure B.5 shows the tabs of this form. The filling of given tabs follows the offer data preparation guidance given in Section 4.3. After filling the information of a single offer in each tab, the next button delivers the next empty form for the next offering. Be aware that the information is filed in the same order as the booking data information is read. When the last offering data was inserted, press the *OK* button to return to the application start form.

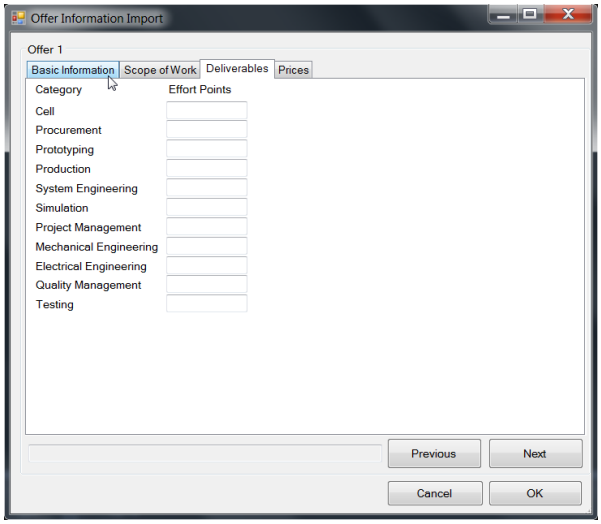
Clicking process in the application start form starts the main program and finishes when models have been selected in form of *eg* files for all used networks. In case of faulty inputs, error messages will appear in form of message boxes or in textboxes at the foot of each form.
The training mode may be run several times. For each run, more training results are generated. These results are added to further folders in the *Training-Results* directory. The final selection of best models takes all training results that were performed and are available into account, if none of the previously obtained has been deleted.



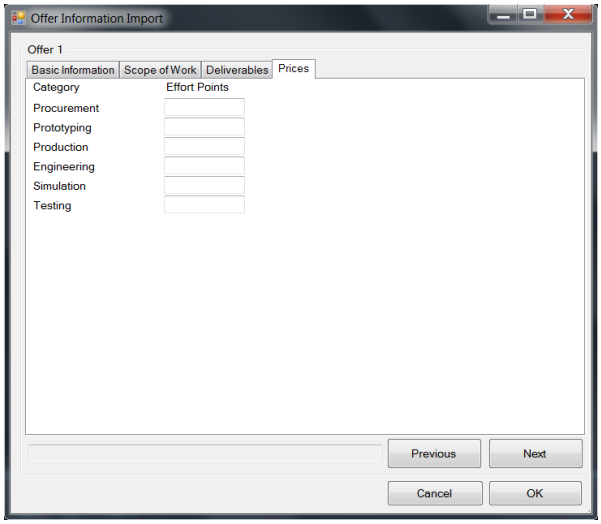
(a) Basics tab.



(b) Scope of work tab.



(c) Deliverables tab.



(d) Prices tab.

Figure B.5: Manual offer definition form of the application.

B.4 Developer Guide

In order to support further development of the prototype, several aspects of the current implementation are represented here.

B.4.1 Classes

Major Program Sequence Classes

From the several classes implemented, the ones depicted in Figure B.6 represent the major classes regarding handling of the program sequence. Chronologically described, they have the following higher-level functionality:

1. Windows Forms Classes: These forms represent user interfaces. They are responsible for collecting information on program mode, data locations and data itself.

2. HandleDataImport Class: This class handles reading information from previously specified files. This includes preprocessing information of data sources if needed, e.g. the accumulation of cost center entries in a booking file. Preprocessed information is stored in certain files like described in Sections B.3 and B.2.

3. HandleMachineLearning Class:
   a) Training mode: This class handles with which kind of methods machine learning is performed. This includes the settings for hidden neurons as well as the selection of the best model. Also normalization is done here.
   b) Estimation mode: The class also handles the preparation of information from which an estimation should be performed and feeding it to respective network models. Final estimations are stored to the selected base folder.

Importer Classes

The class responsible for importing data can draw on several basic importer classes which all return a DataSet object. The class attributes and methods are depicted in this application manual as they are designed for the following files' conversion of data to DataSet objects:

1. Booking file: Files that contain every single booking to cost centers during the process of a project.

2. MS Project file: Timetable of a project containing a list of tasks and respective start and end dates, milestones and quality gates.

3. ProCalc calculation file: Calculations that are performed when tasks are settled by the project planning team including start and end dates and respective effort and cost per task.

Not all of these classes are used in the current state of the application. This is due to the fact that the target of this prototype application changed in the course of this thesis and because of problems with the coherency of data. Nonetheless, this excess functionality may still be of use in further developments.

Data Preparation Classes

Two classes do the major work for preparing information for usage with neural networks. The *StructualLearningDataPrepration* class exists for converting offer information that was obtained by the application's user interface into DataTable objects.
The second one, the *Normalization* class, performs normalization on data before being forwarded to the neural network training. The *Encog* framework, introduced in Section B.4.2 in fact provides normalization routines, but their application did not result in desired results, which led to an own implementation. The chosen process of normalization is explained in Section 4.3.3.

Network Application Class

The NeuralNetworkApplication class is the class of core functionality for the application. It makes use of the *Encog* framework for machine learning applications [Heaton, 2015] and provides means of starting to different kinds of learning approaches regarding different methods, activation functions, training modes and further neural network relevant parameters. The class is designed for networks covering owning a single hidden layer. Settings which were applied for this application prototype are described in Section 5.2.

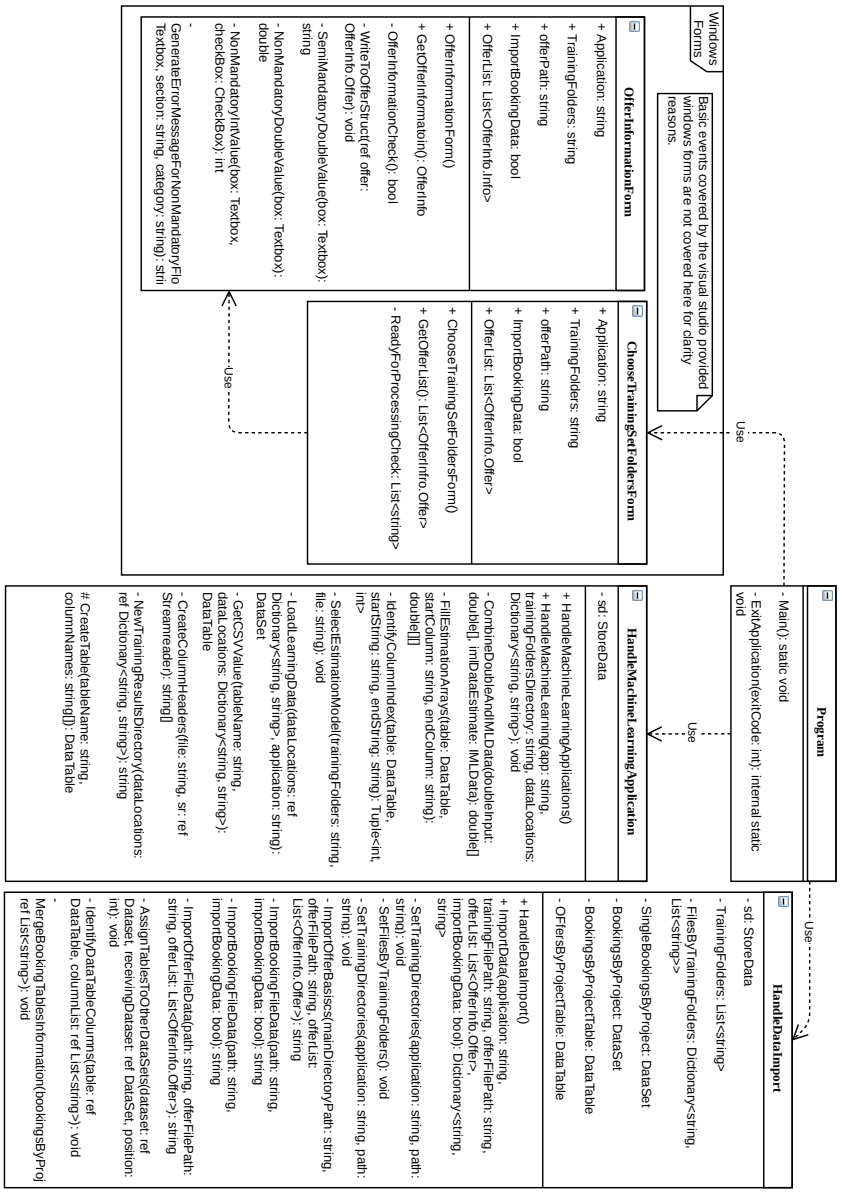


Figure B.6: Classes for major program sequence.

Auxiliary Classes

Several auxiliary classes were introduced to support the classes shown up to this point. Their content is mostly easy to grasp and can be looked up when stumbling over their usage. These classes are:

1. OfferInfo class: for handling information from offers
2. ImporterNode class: for handling hierarchical constructs of files
3. StoreData class: for saving different kind of data to different file types

B.4.2 Encog Machine Learning Framework

The *Encog Machine Learning Framework* was created by Jeff Heaton and makes various machine learning methods available for C#, .Net and Java and with its last update having been published in 2015 [Heaton, 2015]. A web service guiding through the Java Encog Core 3.3.0 API exists, but can be adopted for the C# application. A Wikipedia page holds a brief overview on the learning algorithms featured by the framework. The framework can be integrated to the *Visual Studio* solution with the installation of the *NuGet* package encog-dotnet-core.3.3.0.

B.4.3 NuGet Packages

Several *NuGet* packages were used for the application. These packages provide extended open-source functionalities which mostly handle the access to proprietary software without being in need of a proprietary interface. The *NuGet* package manager is designed for the *Microsoft development platform* and distributed as a *Visual Studio* extension. Multiple programming languages are supported by the .Net Framework and C++.

B.4.4 Implementation Prospect

Further implementation may be in need of rewriting some methods of this prototype. An example for this would be the hardcoded start column and end column names which are forwarded to the StartML method of the NeuralNetworkApplication class. Also this implementation uses many storing and reading processes which may slow down the application.
Furthermore, it originally was intended to estimate the ProCalc calculation for a project offer, which could be done if task names would be brought to a common thread and all data could be provided.
A big issue would be the automation of reading data like given in the offer documents, but would be in need of a fixed structure of these documents to ensure a swift, less error-prone implementation than it would be possible at the moment.
Eventually, a last suggestion concerns the visibility of the program progress for the user. A progress bar and a visual display would inform the user and shows when the program stopped working. The latter issue is currently only apparent from the files that were generated in the base folder. It may be the case that this issue would need a complete restructuring progress of the program with the used Windows Forms classes. The user of this application has to aware that it up till now is only a prototype for proving that machine learning is applicable for offer estimations.

Versioning

This versioning specification closely follows the ideas formulated in *Semantic Versioning 2.0.0* by Preston-Werner [Preston-Werner, 2013], inventor of Gravatars and cofounder of GitHub. *Semantic Versioning 2.0.0* is licensed under the Creative Commons Attribution 3.0 Unported License. To view a copy of this license, visit `http://creativecommons.org/licenses/by/3.0/` or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA. The versioning for the related system specification must be implemented as follows:

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

1. Software using Semantic Versioning MUST declare a public API. This API could be declared in the code itself or exist strictly in documentation. However it is done, it should be precise and comprehensive.

2. A normal version number MUST take the form $X.Y.Z$ where X, Y, and Z are non-negative integers, and MUST NOT contain leading zeroes. X is the major version, Y is the minor version, and Z is the patch version. Each element MUST increase numerically. For instance: $1.9.0 \rightarrow 1.10.0 \rightarrow 1.11.0$.

3. Once a versioned package has been released, the contents of that version MUST NOT be modified. Any modifications MUST be released as a new version.

4. Major version zero ($0.y.z$) is for initial development. Anything may change at any time. The public API should not be considered stable.

5. Version 1.0.0 defines the public API. The way in which the version number is incremented after this release is dependent on this public API and how it changes.

6. Patch version Z ($x.y.Z|x > 0$) MUST be incremented if only backwards compatible bug fixes are introduced. A bug fix is defined as an internal change that fixes incorrect behavior.

7. Minor version Y ($x.Y.z|x > 0$) MUST be incremented if new, backwards compatible functionality is introduced to the public API. It MUST be incremented if any public API functionality is marked as deprecated. It MAY be incremented if substantial new functionality or improvements are introduced within the private code. It MAY include patch level changes. Patch version MUST be reset to 0 when minor version is incremented.

8. Major version X ($X.y.z|X > 0$) MUST be incremented if any backwards incompatible changes are introduced to the public API. It MAY include minor and patch level changes. Patch and minor version MUST be reset to 0 when major version is incremented.

9. Build metadata MUST be denoted by appending a plus sign and a series of dot separated identifiers immediately following the patch version. Identifiers MUST comprise only ASCII alphanumerics and hyphen $[0-9A-Za-z-]$. Identifiers MUST NOT be empty. Build metadata SHOULD be ignored when determining version precedence. Thus two versions that differ only in the build metadata, have the same precedence. Examples: 1.0.0+001, 1.0.0+20130313144700, 1.0.0+$exp.sha.5114f85$.

10. Precedence refers to how versions are compared to each other when ordered. Precedence MUST be calculated by separating the version into major, minor and patch identifiers in that order (Build metadata does not figure into precedence). Precedence is determined by the first difference when comparing each of these identifiers from left to right as follows: Major, minor, and patch versions are always compared numerically. Example: $1.0.0 < 2.0.0 < 2.1.0 < 2.1.1$.

List of Figures

List of Tables

Bibliography

[Ahmad and Tresp, 1992] Ahmad, S. and Tresp, V. (1992). Some Solutions to the Missing Feature Problem in Vision. In *NIPS*, pages 393–400.

[Almeida and Rocha, 2011] Almeida, F. L. and Rocha, R. M. (2011). Introduction of a wiki in an enterprise: Motives and challenges. *Journal of Systems Integration*, (4):46–53.

[Alt, 2012] Alt, O. (2012). *Modellbasierte Systementwicklung mit SysML*. Hanser, München.

[Alt, 2014] Alt, O. (2014). Modellbasiertes Systems Engineering und seine Technologien als Schlüssel für Industrie 4.0. In Maurer, M. and Schulze, S.-O., editors, *Tag des Systems Engineering*, pages 3–9. Carl Hanser Verlag, München.

[Bergstra and Bengio, 2012] Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, 13(1):281–305.

[Bergstra et al., 2013] Bergstra, J., Yamins, D., and Cox, D. (2013). Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In Dasgupta, S. and McAllester, D., editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 115–123, Atlanta, Georgia, USA. PMLR.

[Bishop, 1995] Bishop, C. M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, Inc.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer.

[BMBF, 2017] BMBF (2017). Industrie 4.0 - Innovationen für die Produktion von morgen. https://www.bmbf.de/de/zukunftsprojekt-industrie-4-0-848.html. Bundesministerium für Bildung und Forschung (BMBF).

[Broomhead and Lowe, 1988] Broomhead, D. S. and Lowe, D. (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks. Technical report, Royal Signals and Radar Establishment Malvern (United Kingdom).

[Broy et al., 2010] Broy, M., Feilkas, M., Herrmannsdoerfer, M., Merenda, S., and Ratiu, D. (2010). Seamless Model-Based Development: From Isolated Tools to Integrated Model Engineering Environments. *Proceedings of the IEEE*, 98:526–545.

[Card et al., 1991] Card, S. K., Robertson, G. G., and Mackinlay, J. D. (1991). The information visualizer: An information workspace. *Proc. ACM CHI'91 Conference 1991*, pages 181–188.

[Cavanaugh, 2006] Cavanaugh, E. (2006). Web services: Benefits, challenges, and unique, visual development solution. White paper. `http://www.softgoo.com/download/1069/webservices.pdf`; most recent access on July 30th, 2017.

[Costello, 2012] Costello, T. (2012). RACI-getting projects "Unstuck". *IT Professional*, 14(2):63–64.

[Crnkovic et al., 2003] Crnkovic, I., Asklund, U., and Dahlqvist, A. P. (2003). *Implementing and Integrating Product Data Management and Software Configuration Management*. Artech House, Inc., Norwood, MA, USA.

[Eigner et al., 2014] Eigner, M., Roubanov, D., and Zafirov, R., editors (2014). *Modellbasierte Virtuelle Produktentwicklung*. Springer Vieweg, Berlin.

[Friedenthal et al., 2012] Friedenthal, S., Moore, A., and Steiner, R. (2012). *A practical guide to SysML: The systems modeling language*. The Morgan Kaufmann Omp Press, Waltham, MA.

[Fritz, 2013] Fritz, J. (2013). *Integriertes Vorgehensmodell zur Evaluierung von Systems-Engineering-IT-Werkzeugen im Umfeld automobiler Produktentwicklungsprozesse*. Master thesis, FH Campus 02, Graz.

[Gausemeier et al., 2013] Gausemeier, J., Gaukstern, T., and Tschirner, C. (2013). Systems engineering management based on a discipline-spanning system model. In *Conference on Systems Engineering Research (CSER 13)*. Elsevier B.V.

[Haberfellner et al., 2012] Haberfellner, R., de Weck, O. L., Fricke, E., and Vössner, S. (2012). *Systems Engineering: Grundlagen und Anwendung*. Orell Füssli, Zürich.

[Halawa, 2014] Halawa, K. (2014). *A New Multilayer Perceptron Initialisation Method with Selection of Weights on the Basis of the Function Variability*, pages 47–58. Springer International Publishing, Cham.

[Harley, 2014] Harley, A. (2014). Functional overlays and coach marks for mobile apps. `https://www.nngroup.com/articles/mobile-instructional-overlay`; most recent access on August 7th, 2017.

[Heaton, 2011] Heaton, J. (2011). Programming Neural Networks with Encog3 in C#. St. Louis, MO, USA. Heaton Research, Inc.

[Heaton, 2015] Heaton, J. (2015). Encog: Library of Interchangeable Machine Learning Models for Java and C#. *Journal of Machine Learning Research*, (16):1243–1247.

[Holt and Perry, 2013] Holt, J. and Perry, S. (2013). *SysML for Systems Engineering: A Model-based Approach*. Computing and Networks. Institution of Engineering and Technology.

[Horton and Kleinman, 2007] Horton, N. and Kleinman, K. P. (2007). Much Ado About Nothing: A Comparison of Missing Data Methods and Software to Fit Incomplete Data Regression Models. *The American Statistician*, 61:79–90.

[Huan et al., 2011] Huan, H. X., Hien, D. T. T., and Huynh, H. T. (2011). Efficient Algorithm for Training Interpolation RBF Networks With Equally Spaced Nodes. *IEEE Transactions on Neural Networks*, 22(6):982–988.

[Kriegel et al., 2012] Kriegel, H.-P., Kröger, P., and Zimek, A. (2012). Subspace clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 2(4):351–364.

[LeCun, 1988] LeCun, Y. (1988). A theoretical framework for back-propagation. In Touretzky, D., Hinton, G., and Sejnowski, T., editors, *Proceedings of the 1988 Connectionist Models Summer School*, pages 21–28, CMU, Pittsburgh, Pa. Morgan Kaufmann.

[LeCun et al., 1989] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. (1989). Backpropagation applied to handwritten zip code recognition. *Neural Comput.*, 1(4):541–551.

[Lugger, 2016] Lugger, H. (2016). *Operationalisierung und Evaluation einer Methode des Model Based Systems Engineerings im Batterieentwicklungsprozess für elektrifizierte Fahrzeuge*. Master thesis, Graz University of Technology, Graz.

[Marcus and Stern, 2003] Marcus, E. and Stern, H. (2003). *Blueprints for High Availability*. Wiley Publishing, 2nd edition.

[Miller, 1968] Miller, R. B. (1968). Response time in man-computer conversational transactions. *Proc. AFIPS Fall Joint Computer Conference*, (33):267–277.

[Minke, 2015] Minke, R. (2015). *Stakeholder perspectives model-based-systems-engineering in battery development*. Master thesis, FH Joanneum, Graz.

[Müller, 2016] Müller, M. (2016). *MBSE approach for a demand specific visualization of information in product development*. Master thesis, Karlsruher Institut für Technologie, Karlsruhe.

[Nguyen and Widrow, 1990] Nguyen, D. and Widrow, B. (1990). Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights.

[Nielsen, 1993] Nielsen, J. (1993). *Usability engineering*. Academic Press, Boston.

[Powell, 1987] Powell, M. J. D. (1987). Radial basis functions for multivariable interpolation: A review. In *Algorithms for Approximation*, pages 143–167, Oxford. Clarendon Press.

[Prechelt, 2012] Prechelt, L. (2012). Early stopping - but when? In *Neural Networks: Tricks of the Trade - Second Edition*, pages 53–67.

[Preston-Werner, 2013] Preston-Werner, T. (2013). Semantic Versioning. `http://semver.org/spec/v2.0.0.html`; most recent access on August 10th, 2017.

[Sarle, 2014] Sarle, W. S. (2014). How many hidden units should I use? `http://www.faqs.org/faqs/ai-faq/neural-nets/part3/section-10.html`; most recent access on August 25th, 2017.

[Schiffbänker, 2016] Schiffbänker, P. (2016). *Evaluation of model-based systems engineering in the traction battery product development process*. Master thesis, Graz University of Technology, Graz.

[Seni and Elder, 2010] Seni, G. and Elder, J. F. (2010). *Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions*. Morgan and Claypool Publishers.

[Smith and Simpson, 2004] Smith, D. J. and Simpson, K. G. (2004). *Functional Safety: A straightforward guide to applying IEC 61508 and related standards*. Routledge.

[Specht, 1991] Specht, D. F. (1991). A general regression neural network. *IEEE Transactions on Neural Networks*, 2(6):568–576.

[United Nations, 2015] United Nations (2015). Paris agreement.

[Valnion, 2016] Valnion, B. D. (2016). Combined Simulation and Testing takes development into a new era. *Economic Engineering*, (2).

[Weilkiens, 2008] Weilkiens, T. (2008). *Systems Engineering mit SysML UML: Modellierung, Analyse, Design*. dpunkt.

[Westfall, 2009] Westfall, L. (2009). *The Certified Software Quality Engineer Handbook*. American Society for Quality Press.

[Yu et al., 2011] Yu, H., Xie, T., Paszczynski, S., and Wilamowski, B. M. (2011). Advantages of radial basis function networks for dynamic system design. *IEEE Transactions on Industrial Electronics*, 58(12):5438–5450.

[Zell, 1997] Zell, A. (1997). *Simulation neuronaler Netze*. Oldenbourg.

[Zingel, 2013] Zingel, J. C. (2013). *Basisdefinition einer gemeinsamen Sprache der Produktentwicklung im Kontext der Modellbildung technischer Systeme und einer Modellierungstechnik für Zielsystem und Objektsystem technischer Systeme in SysML auf Grundlage des ZHO-Prinzips*. Phd thesis, Karlsruher Institut für Technologie, Karlsruhe.