Dipl.-Ing. Matthias Müller, BSc

Managing Open Ecosystems: An Agile Approach

Doctoral Thesis

to achieve the university degree of

Doktor der technischen Wissenschaften

Doctoral School: Informatik

submitted to

Graz University of Technology

Institute of Softwaretechnology

Supervisor/1st Reviewer

Univ.-Prof. Dipl-Ing. Dr.techn. Wolfgang Slany

2nd Reviewer:

Assoc.Prof. Kenji Tanaka

Graz, April 2020

This document is set in Palatino, compiled with pdfLaTeX2e and `Biber`.

The LaTeX template from Karl Voit is based on KOMA script and can be found online: https://github.com/novoid/LaTeX-KOMA-template

Affidavit

Abstract

Major parts of our current digital world and the services we regularly use, were only made possible by the existence of open source software projects. For over 20 years, volunteer developers, and increasingly also companies as well as large enterprises, freely publish their source code and executable software for different domains. This enabled not only firms to drive innovation and distribute services globally, but also individuals and small organizations to do so. The challenges of running such projects successfully with a community, are to understand and manage them in a way, that focuses on creating a benefit for all involved actors. Open source contributors follow, different than employees and companies, no direct financial return, but rather very individual motives, such as creating a personal benefit for using/developing other software, knowledge exchange with experts, or feeling good through becoming an active part of a community. These aspects are respectively hard to measure and are often not recognizable for outsiders. Consequently, such projects are situated in rather complex ecosystems that are hard to understand. Popular tools used in industry to analyze these ecosystems, are just applicable on a limited basis for open source organizations due to their open character. Intangible values, such as knowledge, providing support, or personal wellbeing, are key for their long term success and must get covered when analyzing them. These values can also be found in the research strands of sustainable innovation and New Business Models. Approaches and tools emerging in these domains are capable of analyzing and considering these intangible aspects in detail. Thus, this thesis combines these domains in a conceptual research approach. Theories and frameworks get analyzed, applied, and extended to enable a holistic picture of open ecosystems. As the analysis shows, this ultimately allows to gain deeper insights into open source communities, the complex system they are situated in, and the value-processes within them.

Such an open source software project is Catrobat. International growth and a variety of innovative connected services, made it necessary to analyze the project from various perspectives and adapt the internal processes to the project's needs. Catrobat is based on a diverse ecosystem of various involved actors, of which each has individual needs and expectations. For instance, the majority of developers is connected to Graz University of Technology, which goes along with educational aims and needs that are necessary to get considered. Further, the services attract several thousand active users, that express their functional and non-functional expectations and desires. But also, external stakeholders, such as companies or other communities, expect a certain (intangible) revenue for their involvement. But, common project-management frameworks from software industry, that are intended to cover those needs and expectations, cannot cope with such project's open characteristics. Processes need to get introduced, that ensure that the needs of the actors are balanced, but also can handle such project's unique circumstances. Permanent change of contributors, different forms of contribution, or a worldwide distributed community, are just some of the challenges that got identified in the conducted case study relying on surveys of the community and an analysis of the digital activities of the contributors. To overcome these hurdles, the central role of product owners, who balances the needs of all actors, and a transparent workflow to do so, got introduced. Aiming to create a net-benefit for all involved actors, this new process has been perceived positively by the contributors in a final survey, i.e., especially the resulting increased amount of communication and structure, and also shows promising results on the project's public development tracking system in regards of unhandled requests and tickets.

Kurzfassung

Die digitale Welt, in der wir uns heute bewegen, wurde in großen Teilen durch offene Software Projekte ermöglicht. Freiwillige, sowie auch Unternehmen und Konzerne, stellen der Öffentlichkeit Quellcode und Anwendungen zur Verfügung die in verschiedenen Bereichen Einsatz finden. Dadurch kann nicht nur von Firmen, sondern auch von Einzelnen und kleinen Organisationen Innovation weltweit vorangetrieben werden. Die Herausforderungen solcher offenen Projekte, um sie nachhaltig und erfolgreich zu betreiben, sind es jedoch, sie in einer Art und Weise zu verstehen und zu steuern, die es zum Ziel hat für alle Beteiligten einen Mehrwert zu erstellen. Die Mitwirkenden folgen, anders als in Unternehmen, keinen finanziellen, sondern sehr individuellen Motiven (z.B., einen Vorteil in der Nutzung, zu lernen, einer Gemeinschaft anzugehören, etc.). Diese Motivation ist schwer messbar und oft für Außenstehende nicht erkennbar. Entsprechend komplex stellt sich das Ökosystem hinter diesen Projekten dar. Gängige Methoden zur Analyse kommen durch den offenen Charakter an ihre Grenzen diese Systeme und Beziehungen darin darzustellen. Alternative Werte, wie Wissenstransfer, Unterstützung oder Wohlbefinden, sind für diese Organisationen maßgeblich. Diese Werte finden sich jedoch auch in einem anderen Forschungszweig wieder. Ansätze aus dem Bereich der nachhaltigen Innovation und der neuen Geschäftsmodelle zeigen sich vielversprechend um die aufgezählten Aspekte analysieren und darstellen zu können. In dieser Thesis werden diese Forschungszweige durch ein konzeptionelles Vorgehen vereint. Theorien und Werkzeuge werden evaluiert, angewendet und erweitert, sodass ein ganzheitliches Bild von Ökosystemen ermöglicht wird und offene Projekte holistisch betrachten werden können.

Ein solches offenes Software Projekt ist Catrobat. Wachstum und eine steigende Anzahl an entwickelten Diensten macht es notwendig das Projekt ganzheitlich zu analysieren und interne Vorgänge zu adaptieren. Catrobat basiert auf einem Ökosystem verschiedener Akteure, die alle individuelle Ansprüche und Erwartungen haben, sowie sich unterschiedlich einbringen. Zum Beispiel, hat der Großteil der Mitwirkenden eine Verbindung zur Technischen Universität Graz, was eine essentielle Bildungsperspektive mit sich bringt. Des Weiteren, nutzen zehntausende Benutzer die bereitgestellten Dienste, welche ihre Wünsche und Anforderungen an das Projekt heranbringen und maßgeblich an dessen Erfolg beteiligt sind. Schließlich bringen sich auch externe Akteure, wie Firmen oder andere Gemeinschaften, ein, die eine eigene Erwartungshaltung haben. Herangehensweisen aus der agilen Softwareentwicklung, die ein Steuern dieser Bedürfnisse ermöglichen sollen, sind jedoch nur bedingt einsetzbar. Adaptiere Prozesse sind notwendig um solch offene Projekte zu steuern. Diese sollten sicherstellen, dass auf die Bedürfnisse der Beteiligten eingegangen wird, aber auch die Rahmenbedingungen berücksichtigt werden. Ein ständiger Wechsel der Beteiligten, unterschiedlichste Formen der Mitarbeit oder die örtliche Verteilung der Organisation, sind nur einige der beeinflussenden Faktoren. Die Durchführung von Umfragen, sowie eine Analyse der Mitarbeit bestätigten diese Probleme im konkreten Fall. Durch das Einführen einer Rolle die den Überblick über die Organisation und Beteiligten behält, sowie einen definierten Prozess, der transparent alle Bedürfnisse balanciert, konnte in dem Fall von Catrobat ein entsprechender Ablauf geschaffen werden, welcher auf die Mitwirkenden ausgelegt ist. Dieser Ablauf folgt dem Ziel einen Mehrwert aus aktiver oder passiver Mitarbeit am Projekt zu kreieren. Dieser Schritt wurde abschließend durch weitere Evaluierungen bestätigt und zeigt positive Effekte auf das Projekt. Umfragen zeigen, dass sich die Kommunikation und Ziele für die Mitwirkenden subjektiv verbesserten, aber auch die Anzahl nicht bearbeiteter Tickets reduziert werden konnte.

Acknowledgement

This thesis would not have been possible without the continuous and generous support of certain people over the last years. To begin with, I would like to thank Professor Wolfgang Slany for his guidance and supervising me already on from my Bachelor's studies. His continuous personal support enabled me to conduct my research in depth and also to focus on a research-area of my very personal interest. Further, I want to express my appreciation to Professor Kenji Tanaka, who welcomed me to learn at his lab and agreed to accompany me as reviewer and examiner during the final phase of my studies.

I am also very thankful for the support I received from my colleagues at Graz University of Technology and the Catrobat project. I want to especially give thanks to Christian Schindler and Wolfgang Vorraber, who mentored me during my scientific career and actively supported me in conducting the research for this thesis. Working together was a pleasant time that I do not want to miss. But, also all staff and students related to Catrobat and the Institute of Software Technology, that I had the honor to work together with, have been an essential part of my work and enabled me to conduct my research. I am pleased that I had the honor to work together with each of you. It was a fun time for me that I enjoyed on from my first day as PhD student.

Most importantly, I am grateful for my family and friends, who backed me on my way to this thesis. Especially for the key people in my life: my parents, siblings, and my beloved partner Conny. Without their support and encouragement, I would not be where I am today. I owe you my deepest gratitude for all the big and small things you supported me and my journey with, and the deep understanding you showed to me.

Contents

List of Figures

1. Introduction

1.1. Motivation

During the last decades, the technology landscape has tremendously shaped the world we live in today. The world wide web, the digitalization of businesses, and most recently the triumph of mobile technologies changed the way we are living, interacting, and working. Technology and therewith related innovation is released faster than ever before, continuously challenging the status quo. We are facing constant change in terms of new products and services that come to market. Furthermore, we can see new tech-companies and businesses emerging that are dominating our economy and revolutionizing even long-living industries. This is not only changing the way of living and working, but also how products, services, and ultimately value get created. As a result, new forms of management and business models emerged, that are no longer only driven by firms, but increasingly by whole ecosystems and open communities.

As we can see today, this technological revolution would not have been possible without the contribution of thousands of community driven open source software (OSS) projects (a more detailed definition of OSS is provided in the following chapters), that made their code and knowledge available to the public, fostering innovation and change. Diverse communities of volunteers and professionals, often backed by non-profit organizations, official entities, and firms, are jointly developing software that is available for the public good. Whereas the origins of the OSS movement go back to research environments, it is today commonly used in and also developed by industry. Well known examples, such as Linux distributions as common operating systems for servers, Android as most commonly used mobile operating system[1], or databases built on open source software (e.g., MySQL or PostgreSQL) successfully competing to closed-source commercial vendors[2], have been a driver of our digital economy and also enabled various new possibilities for our society. Consequently, businesses and even whole industries are today dependent on OSS projects, building the basis for services they provide, products they manufacture, or even business models they realize.

This urges the need to better understand the phenomenon of these projects, but especially the communities behind them and how their value creation (that is by definition captured by business models) can be streamlined into a common direction. These communities are social constructs that evolve over time and are influenced by dynamic processes, as this thesis shows. The ecosystems of open source software projects, consisting of various stakeholders, constantly grow, reaching new industries and also inspiring other movements with their openness and dynamics. Whereas this clearly indicates the importance of the OSS movement, it also highlights the challenges for the future and points out some still open questions. As *Wired* Journalist Klint Finely in his article *"Open Source Won. So, Now What?"* [Finley, 2016] describes, the movement encountered all doubts of the past and now dominates the digital landscape, but consequently faces new challenges such as funding, dependency on companies, or sustainable development. Journalist Mitch Wagner from *Light Reading* even goes further, claiming that the revolutionaries (open source communities) stormed the castle, but now *"(They) have to figure out how to actually run things"* [Wagner, 2018]. These, from a scientific view rather populistic, articles represent the current state of open source software perceived in industry. But, as highlighted in this thesis, also research in the last years showed that OSS projects evolved and changed their way of contribution and interaction. This leads to various relevant topics for researchers, investigating how OSS communities changed, how they interact with stakeholders within their ecosystem, how they shape our technological as well as business landscape, and how the current challenges in management of such projects might get encountered. As discussed in this

[1]https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/, accessed on March 28th 2020

[2]https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/ , accessed on August 20th 2019

work, these challenges not only occur in open source projects, but also in other projects that are either of a non-profit community nature or driven by intangible values (e.g., non-monetary values) in open communities. Therefore, not only a general picture of the current state of OSS projects and their communities (i.e., the contributors to the communities and involved stakeholders) is presented here, but these projects get also investigated from a general business model and ecosystem perspective, outlining similarities to other industries and research strands. Based on these insights, OSS is also analyzed from a general software-industry context and how commonly used methods, frameworks, and processes can get adapted to these specific open environments.

This thesis aims to create a holistic picture of OSS projects, their communities, processes and underlying business-relations by outlining the current status quo of these projects from different perspectives. This work refrains from generalizing open source projects and trying to phrase a general applicable picture of such projects, which is not possible through the different circumstances they are situated in, but rather highlights potential challenges and chances for such projects on the example of the non-profit OSS project *Catrobat*. The used methodologies focus on shedding some light into practical real-world aspects, that in similar ways can also be found in other open community/ecosystem projects, but are intended to provide a working case on how OSS projects can be successfully managed through introducing processes that pay respect to the unique setting of such ecosystems. The Catrobat project at Graz University of Technology is thus used to investigate how open source projects face change and how methods from agile software development may help to handle this change. Whereas frameworks such as extreme programming (XP), Scrum, and Kanban are already practice in industry, they need to be considered with care in the sensible social and value context of open ecosystems of various actors, including volunteer contributors, profit-seeking companies, and charitable organizations. Within this thesis, the ecosystem and used processes of this project are analyzed and changes to it are performed based on beforehand conducted theoretical research insights. As a result, also implications for the future, relevant for this specific project but also for open community projects in general, are drawn and further discussed.

1.2. Aims and Methodology

The research for this work emerged from ongoing management and process changes in the OSS project *Catrobat*[3], situated at Graz University of Technology. As described later, the project consists of an international community of users and contributors, but also has a strong educational context in relation to Graz University of Technology and is connected to other external stakeholders such as companies or other OSS organizations. Thus, it represents an interesting case that has been in the need to adapt its processes to this environment and ensure a long-term and sustainable development of the provided software services.

As this thesis lines out, the domain of OSS today is broad and definitions, at least from the public perspective, vary. Thus, each project is situated in a unique setting and differs in various aspects. Consequently, this thesis is based on an adequate research strategy to represent the required realistic view on such a specific environment (i.e., the Catrobat project). Stol and Fitzgerlad [Stol and Fitzgerald, 2018] provide the ABC framework for Software Engineering (SE) research, based on Runkel and McGrath [Runkel and McGrath, 1972], which helps to situate research in this domain. This framework categorizes research strategies for SE by two dimensions: the authors control on the environment (*obtrusiveness*) and the *generalizability*, i.e., how applicable the results are as general valid theory. According to their work, each strategy and methodology follows a certain purpose and therefore has individual strengths and weaknesses that must be regarded [Stol and Fitzgerald, 2018].

The research for this thesis followed two phases: in the first stage to seek knowledge in the domain of OSS with the help of Catrobat and later on to develop solutions to identified problems within this case. The knowledge seeking part focused on analyzing the business and network aspects of OSS environments. Therefore, literature has been reviewed and conceptual frameworks (e.g., from the domain of New Business Models and value networks) got applied and also verified on Catrobat. This was intended to gain a better understanding of the value-dynamics within such an open

[3]https://www.catrobat.org

setting and gain deeper knowledge how this specific case is situated, also in comparison to other OSS and open projects.

To analyze the specific case of Catrobat, the research strategy for the *solution-seeking* part of this work (i.e., adapting the management and processes to the environment and needs of the project) is situated in realistic context-near strategies within the *Natural Setting Quadrant* of the ABC Framework. Since OSS projects, such as Catrobat, have unique settings and certain information is due to the informal character of these projects often only accessible for insiders (especially management and business aspects), large parts of the practical work are designed as single case study and are classified as a revelatory case (according to the definition of Yin [Yin, 2018]), since detailed information can get gained from within the Catrobat project. Whereas this strategy has been primarily used by the author in the first phase of his practical work to analyze the project and build and share a solid knowledge base of it (e.g., how the community is structured, the motivation of the contributors, etc.), he later on also applied an *Field Experiment*, that in contrast to a *Field Study* involves control over events within the researched case [Stol and Fitzgerald, 2018], by introducing and afterwards evaluating an agile workflow that was intended to improve processes within and management of Catrobat.

Since this thesis is a cumulative work, consisting of several peer-reviewed articles, also different methods got used to accomplish this holistic view. Also, this thesis follows the ABC approach - *Actors, Behavior, and Context* - as described in the framework by Stol and Fitzgerlad [Stol and Fitzgerald, 2018]. The chapters cover these three dimensions and will further describe those aspects, as subsequently described in detail in Section 1.3. As cumulative work, parts of the work (especially descriptions of the project), can be redundant. Each chapter and section describes in detail what methodology it is based on and what the main aim and research questions of this section/chapter are. To guide the reader through this work, the thesis can be described by three research objects with sub-topics that are covered throughout the thesis in the presented order and will get discussed in the last chapter:

- RO1: The economics of open communities
  - RO 1.1: Creation of intangible value by communities
  - RO 1.2: The business aspects of open source software projects
  - RO 1.3: The involved ecosystem of different stakeholders in open organizations
- RO2: How are Open Source Projects evolving over time?
  - RO 2.1: The structure of Open source communities
  - RO 2.2: The change of open source software communities over time
- RO3: How can an evolving Open Source Software project be managed to ensure sustainable development?
  - RO3.1: What are the benefits for contributors to (educational) open source projects?
  - RO3.2: What influences the outcomes of open source communities?
  - RO3.3: Which methods from the classic software industry can be adopted?
  - RO3.4: How can the needs of all actors in the ecosystem open source be balanced?

All those aspects got covered in various peer-reviewed and published articles (either available in international journals or conference proceedings) of the author, which are either fully or partially included in this work. They are put into a logical order, creating a holistic picture and underpinning the theoretical background, ultimately covering those proposed objectives and providing in-depth background information. The corresponding articles are cited in the footnotes of the sections and a detailed overview of the author's publications in regard to the research of this thesis is provided in Appendix A.

1.3. Organization of the Thesis and Outline

First, in Chapter 2, the economic aspects of open source software projects and how value creation takes place in them is discussed. To provide the necessary context for the research, open source software projects and their current state from a business viewpoint are reviewed in a detailed way from a New Business Model perspective. To advance this theoretical work on the business perspective, in Chapter 3 the relation of value networks and ecosystems to (New) Business Models (NBMs) is investigated and a connection to the defined *RO1* is therewith drafted.

In the following Chapter 4, the evolution of the open source movement, on from its informal start through *the Cathedral and the Bazaar*, to the current stage of the *Third Generation of Open Source* is lined out. Further knowledge is therefore gathered by an empirical analysis of open source software repositories, deepening the understanding of the communities' contributors and their change over time (*RO2*). This is intended to provide the readers the needed insights into the contributing-actors of open source projects.

The actual behavioral part of this research work is described on the case of Catrobat in Chapters 5ff, which is also the unit of analysis regarded in the conducted case study and field experiment. On Catrobat, the benefits, but also potential issues, of agile frameworks and methods are discussed and applied, providing a real-world example that gets analyzed within the scope of the proposed Research Objective 3.

The results of the thesis are discussed resumed in the last Chapter 8, further lining out occurring limitations and potential future work in the domain and the specific case.

1.4. Scientific and Practical Contribution

As this thesis shows, OSS projects have already been the subject of many scientific publications in the past. Especially empirical research has been enabled by the public availability of data from code-repositories, mailing lists, and other resources (e.g., bug ticket tracker, chat channels, etc.). In addition, many cases of OSS projects have been scientifically published as case studies, further helping to understand the phenomenon *Open Source*. Nevertheless, the socio technical system of the communities and the underlying processes can just barely be formally defined being valid for all projects. Applying research strategies aiming to foster the realistic context perspective therefore provides constantly new insights, helping to gain deeper knowledge about open communities in that value of different kind gets co-created. Especially the field study strategy may help to gain an in-depth understanding of organizations and communities [Stol and Fitzgerald, 2018]. Therefore, this thesis adds value by combining methods and insights from different research strands (i.e., OSS in relation to New Business Models, ecosystem management and agile software development), creating further knowledge in the domain of ecosystems involving open communities. The case of Catrobat further provides another potentially working case on that future research can be based on and practitioners can learn from.

2. The Business of Open Source Systems - Open Principles in New Business Models for Information Systems[1]

Business models have been a trending topic for academia and industry in recent years. Especially new business models (NBMs) are increasingly gaining importance not only for today's society, but also research. Driven, beyond others, by sustainability, sharing, and collaboration, they gather more and more attention by a manifold audience and got an emerging topic for economic research. These upcoming models are built upon diverse communities co-creating value and are enabling a new view on today's businesses. However, no general definition of what new business models are, is currently available, leaving space for interpretations. Regardless of the definition, technology has been a main enabler for business models and innovation in the last decades. One of the technologies that has often been therewith connected is the development of open source software. Focus of this connection has mostly been how open source enables business models and supports innovation. Already early the collaborative nature of users and other actors in open source projects has driven innovation that was able to compete with professional businesses [Von Hippel, 2001]. But, open communities also provide chances for these businesses. Early on, companies started to create value from open (source) projects and to gain benefits out of them in various business models [Chesbrough, 2006a]. But the social aspect of collaboration in and value creation of the underlying open communities has not been considered in or as a business model itself yet. Although we

[1]This Chapter is published as [Müller et al., 2019e] CC-BY. DOI: 10.3390/joitmc5010006

can see innovation originating from communities which is in a certain way commercialized by firms, these communities still innovate for free in the terms of money, but for gaining alternative rewards [West and Lakhani, 2008, von Hippel, 2017]. Failing to represent these communities, their value creation and distribution process by common business model notations points out a current gap in research.

Especially in times in which more and more companies get connected to communities that innovate and create open value it is essential that companies understand the communities, but also the communities get insights into their value creation and delivery process. Traditional business models are usually focusing on monetary values, but as we can observe, the current characterization of NBMs pays also respect to this alternative definition of value and its co-creation [Jonker, 2012]. By regarding communities through new business models further implications on how to improve the value, independently of its definition, for all involved actors may be made. This might help to support innovative and open communities in their value creation and increase the thereby created net-benefit. Due to the large amount of published work and cases we focus in this first work on open source projects and their communities, representing collaborative value creation and innovation. We target to investigate similarities between the emerging trends in NBMs and the already well investigated domain of open source software. We expect that a holistic view on these domains will provide further insights in both of them, enhancing the understanding of how (open source) communities create value in new business models that are not necessarily focusing on monetary values or business objectives.

Open source projects have already been subject of extensive research in the past, considering a variety of domains and different projects. As current trends in research and industry show, these open projects are evolving, strengthening the involvement of businesses in the ecosystem and collaboratively creating value for different involved parties. This review aims to enhance this research by fostering the understanding of how open source projects can be represented by the currently emerging definitions of new business models. Therefore, similarities in research on these two domains get pointed out by outlining the status quo in these fields and comparing their recent progress. We intend to build the foundation for further research in combining these research strands and to establish the base for following project-related practical and action-based scientific work by reviewing current literature covering these specific fields [Webster and Watson, 2002]. This work may give further insights into how collaborative value is created by communities, not necessarily following monetary profit, but also paying respect to social aspects and individual motivations for participation. Whereas the idea of NBM is relatively new, research on open source software emerged in the late 1990s, resulting in a large base of available resources for analysis. The objective of this work is to analyze if open source software communities match the current definition of NBMs. This would allow new perspectives for research on open source communities and ecosystems, but would also add additional cases for further studies on NBMs. For this analysis, current literature on business models and open source software gets reviewed and upcoming trends pointed out. Publications from both domains and a variety of sources get considered, trying to avoid a bias of focusing on publications from single geographical regions or only from top-publishers [Webster and Watson, 2002]. Furthermore, emerging ideas and new approaches in these fields are highlighted, that might also foster the potential connection between open source software projects and NBMs.

2.1. Aims

This article aims to show a gap in business model research when it comes to co-creative open communities. Communities, open to contribute for everyone, in the last decades intensively impacted innovation and have especially been an active driver for technology. However, whereas in many cases firms realized these innovations in monetary terms, also other values have been created for and captured by the communities and their individuals that have been involved in the innovation process. Especially in times in which business models need to evolve and firms more and more engage in open communities, the need to better understand these communities comes up. As an often-used example for these communities, the well-defined domain of open source communities, also directly connected to the principles of open innovation, has been used in this work.

First, we sum up the current state of research and highlight upcoming trends in the emerging work on business models. We therefore focus on frameworks that origin from reputable sources in this field and that have been applied in various published cases. We line out the progress of development in this field, especially regarding the representation and characteristics of business models as well as the increasing consideration of social aspects. As indexing databases show, further studies and ideas in the field of NBMs are constantly emerging in specially dedicated conferences and different international journals on production and sustainability issues. Furthermore, this topic has been discussed in a variety of business publications, that get considered and analyzed in our review.

Second, we review existing literature on open source software projects. The therefore considered open source literature primarily focuses on the underlying communities, the role of individual contributors in them, and the connection to industry and businesses. A variety of publications, primarily originating from international journals and conference-proceedings especially focusing on this domain, has been evaluated from common index databases, i.e. Scopus, IEEE Xplore and Springer, and taken into consideration to line out the current state of research on open source. These publications include peer-reviewed empirical research results, case studies and literature reviews focusing on open source software development and communities. Especially results from published and analyzed cases from community and industry driven open source projects within the last years provide detailed insights and have been used for this work.

Third, the connection between open communities, firms' innovation processes, and business models gets highlighted. Especially in the research strand of open innovation the community aspect gets pointed out in various publications. Furthermore, these publications also relate to the connection between business model (innovation) and open innovation. We show how this strand is connected to this review and potential implications for communities involved in open innovation approaches.

As main work, we combine these two fields by regarding open source software communities from a NBM perspective. We map the existing insights gained from open source projects to the emerging definitions and characteristics of NBMs. This approach of mapping is intended to make a statement if open communities may be adequately represented by frameworks and tools originating from research on NBMs. In the following discussion and conclusion we point out similarities as well as potential room for further action based research in combining both domains.

2.2. Background

2.2.1. New Business Models

Creating and capturing value can be a simple description of the functions of a business model [Chesbrough, 2006a]. However, today's literature provides a variety of definitions of what a business model is and how it can be described and represented (e.g., [Osterwalder and Pigneur, 2010], [Chesbrough and Rosenbloom, 2002], or [Johnson et al., 2008]). Depending on the field of application and targeted usage of the model, researchers contributed various definitions to this domain [Yun et al., 2016b]. They all differ in several aspects such as the number of elements, scope, and viewpoint, as the currently common representations of business models demonstrate:

- Early concepts to describe business models such as the '"Fribourg ICT-Management Framework" [Teufel et al., 2004] and the "STOF Model" [Bouwman et al., 2008] focus on basic building blocks consisting of technology, service, finance in an socio-economic context.
- The widespread "Business Model Canvas" (BMC) by Osterwalder [Osterwalder and Pigneur, 2010] and its derivations such as the "Business Model Starter Kit" by Breuer and Ketabdar [Breuer and Ketabdar, 2012] have a focus on the value proposition for customers while also including important concepts for value generation on customer and provider side.
- Frameworks such as the "BIZTEKON Business Model Skeleton" [Gjerde et al., 2007] and the "Service Innovation Triangle" [Furseth and Cuthbertson, 2013] provide more detailed views on selected aspects of business model elements.

Research on business models gained further popularity in the last years and the social aspect of business models is getting increasingly important. The concept of business models is now also used for the analysis of social and inclusive businesses [Michelini and Fiorentino, 2012]. This can be seen in new upcoming business model frameworks focusing on additional therewith connected aspects:

- The "Social Business Model Canvas" [Tandemic, 2017] for instance focuses on social innovation to create business models that also have a social impact in addition to commercial revenues.
- Similar, the "Triple Layered Business Model Canvas", enhances Osterwalder's notation with an environmental and social layer [Joyce and Paquin, 2016].
- The "Service Business Model Canvas", also based on the building blocks of the BMC, allows a holistic view on services that are based on co-creation of value [Zolnowski et al., 2014].
- A stand-alone approach not grounding on Osterwalder is provided by Bocken et al. [Bocken et al., 2013] who provide a framework and a process to innovate business models with a strong social and sustainable focus.

All in all, one can see that research on business models is increasingly paying respect to new components, such as social aspects, environmental issues, and an alternative definition of value/profit, not only seen from a monetary perspective. Constantly new frameworks emerge, adding additional perspectives to business models and enhancing our understanding of them.

Business models can be seen as a driver for management research, since these models may be regarded in different non-exclusive roles (e.g. scientific or descriptive as role-model) the same time [Baden-Fuller and Morgan, 2010]. Research in the last years on business models is further enhancing the understanding of the domain in general, but is also intensively paying respect to new trends in our society and economy. Business models are changing due to a new way of thinking [Jonker, 2012]. As an example, sustainable business models that create value and not primarily focus on profit generation got in the focus of research in recent years [Dentchev et al., 2016]. Evaluating this work highlights the potential of this emerging field for academics and practitioners [Lüdeke-Freund and Dembek, 2017]. But also the part of innovation and technology is changing today's business models [Timmers, 1998].

As already mentioned, there are multiple ways to define business models, all considering different viewpoints. Similar to business models, also these upcoming new business models can be formalized in different ways. One approach by Jonker [Jonker, 2012] preliminary identified seven features that shape new business models:

- Cooperative collaboration
- Creation of multiple value(s)
- Money not only mean of trade (e.g., to be earned or traded)
- Economy based on needs and uses
- Access over ownership
- Parties' long-term commitment
- Alternative forms of money (e.g., points)

This definition also fosters the aspect of (co-)creating multiple values for various actors within a business model. In comparison, current models often lack in managing this creation of multiple values that can especially be found in services [Jonker, 2012, Zolnowski et al., 2014]. The involvement of stakeholders in the value-creation process also comes along with certain challenges, that are also interesting for research [Dentchev et al., 2016]. Co-creative organizations of different actors can lead to complex ecosystems that are hard to manage and understand [Vorraber et al., 2019a]. But the fact that today's boundary between for-profit and non-profit organizations is blurring, proofs that creating shared value for the involved actors is possible [Porter and Kramer, 2011]. As an example, the investment in communities in inclusive business models also allows companies to increase their profit [Michelini and Fiorentino, 2012]. As a result of these manifold benefits, new viewpoints in this field are emerging rapidly, fostering a multi-value and actor centered understanding of business models. These models are also focusing more on a value perspective independently of the definition of value as monetary currency and taking into account current needs of the related actors, society and environment.

2.2.2. Open Source Software

Although open source software got in the focus of research and industry in the end of the 1990s with the definition of licenses and principles, its origins go as far in the back as in the 1960s, when it was common for programmers to share their code [Lerner and Tirole, 2002]. Over the last 20 years a variety of projects established using these widely accepted and well defined open licenses. Although they all refer to develop open source software, projects are following varying characteristics and licenses, slightly differing in certain aspects [Gacek and Arief, 2004]. Typically, open source projects get considered by the public and developers solely in a technical view, only peripherally connected to business objects [Krishnamurthy, 2005a]. Especially early publications focused primarily on the collaborative aspect of this development method. As best-known example, Raymond [Raymond, 1999] presents in "The Cathedral and the Bazaar" 19 lessons describing the process of producing open source software based on his experience in this open domain. The described collaborative process of open development, also including users and stakeholders, differs from conventional software development approaches in the dimensions of incentives, control, and coordination [Von Krogh et al., 2012]. Whereas conventional software development is often compared to building a closed cathedral, where just a few people are involved in the construction progress that is not public, Open Source Software (OSS) development can be seen similar to an open bazaar of different people co-creating and sharing value [Raymond, 1999]. This aspect is also in the focus of several of Raymond's principles:

- Every good work of software starts by scratching a developer's personal itch.
- Good programmers know what to write. Great ones know what to rewrite (and reuse).
- When you lose interest in a program, your last duty to it is to hand it off to a competent successor.
- Release early. Release often. And listen to your customers.
- If you treat your beta-testers as if they are your most valuable resource, they will respond by becoming your most valuable resource.
- The next best thing to having good ideas is recognizing good ideas from your users. Sometimes the latter is better.

- Any tool should be useful in the expected way but a truly great tool lends itself to uses you never expected.

Excerpt of Raymond's lessons in "The Cathedral and the Bazaar" [Raymond, 1999]

Besides Raymond's OSS movement, also the Free Software (FS) community, initiated by Richard Stallman, established. In his essays [Stallman et al., 2002], he outlines that software needs four freedoms:

- The freedom to run it
- The freedom to study and to change it
- The freedom to redistribute it
- The freedom to distribute modified copies

Whereas the latter definition of OSS is primarily focusing on collaborative and development aspects, the characterization of FS is more of a philosophical nature. This perspective and the ethical reasons for it are often unknown by users, who are just regarding open source as one single ideology [Stallman, 2009]. Today several FS licenses pay respect to this movement's philosophy and are widely used by developers. However, due to their similar nature, most open source software can also be considered as free software [Stallman, 2009]. Therefore, in this work we will refer to open source as the definition of Free/Libre Open Source Software (FLOSS), meaning that they comply to the principles of both communities.

Due to its character and availability for researchers, open source software has been the subject in a large number of publications since its emergence [Von Krogh and Von Hippel, 2006]. Not only computer scientists, but also researchers from other disciplines such as management, economics, or law considered open source for their work [Gacek and Arief, 2004]. Therefore, a broad research-basis and numerous published cases are available to investigate open source from various perspectives and combine it with other disciplines, such as in this case with research on NBMs new business models.

2.2.3. Communities for Open Innovation

Fast progress in technology, shorter product cycles and the therewith connected high costs of innovation challenged firms in the late 20th century, leading to the need of new and more efficient ways to innovate [Chesbrough, 2007]. As a result, the open innovation approach came up. This approach treads internal and external ideas and ways to market the same way [Chesbrough, 2006c]. Various movements, e.g. user innovation, crowdsourcing or open source innovation are based on this principle and are referred to open innovation [Yun et al., 2016a]. Whereas this phenomenon started in the high-tech sector and at large international firms, today the stream of open innovation got mainstream and is also common in smaller enterprises [Gassmann et al., 2010]. Furthermore, in recent times we can see an increasing interest of firms to collaborate with external communities as potential source of (open) innovation [West and Sims, 2018]. This idea of innovation by communities is not new and has already been subject of extensive research in the past. As pointed out by von Hippel [Von Hippel, 2001], user innovation communities can be found in a variety of domains and they can differ in their structure. One of the most referenced communities in this context is the one of Linux, originating from the open source movement. However, especially the fourth industrial revolution, coming along with various new technologies, will influence the economy and bring many potential new opportunities also for open innovation [Park, 2017]. Already now the outcomes of these innovations get visible and impact economic structures on different levels [Lee et al., 2018]. One famous example for these new innovative possibilities is the emergence of the sharing economy business model [Park, 2017], building on technology enabled collaboration and connection of internal and external actors.

Especially in the (direct or indirect) creation of technological innovation, communities play an important role [West and Sims, 2018]. Thus, especially when it is related to innovation, firms actively engage in such open communities, e.g. by sponsoring the projects or paying active contributors [West and Lakhani, 2008]. This direct involvement, often intended to ensure sustainable innovation, comes with certain risks for the viability of the communities, especially when the main contribution is done by employees [Chesbrough and Appleyard, 2007]. The connection between firms and communities is still raising questions and may be challenging, e.g. if firms ignore the communities' needs or if the created output is not meeting the firms' expectations [West and Lakhani, 2008]. But still, current models show that users, consumers, and communities play an important role in firms' (open) innovation process towards new business models [Yun et al., 2016b]. Therefore, a proper understanding of the related communities and individuals, their aims and needs is essential to successfully innovate in an open way and to respond to future economic challenges.

As already mentioned, open innovation also relates to business models and their innovation. When the term open innovation came up, it has been early actively connected to business models, referring to the fact that the economic value of (innovative) technology is created through commercialization [Chesbrough, 2006c]. Today, we can see that the connection between business models and technology is far from clearly defined, resulting in two main research movements within this domain [Baden-Fuller and Haefliger, 2013]. Especially the ongoing progress in this field, the different current movements in research, and practical application in various industries leads constantly to new insights and questions connected to open innovation and its relation to as well as influence on business models. But, since in current times more and more disruptive innovation is on its way, we can see that new combinations of technologies will also lead to new innovative business models in various sectors [Lee et al., 2018]. Business models will be further defined by value creation happening through the interplay of technologies and markets, in which open innovation will be essential [Yun et al., 2016b]. Therefore, we will see in future even more progress in this domain, outlining the important correlation between business models, technology and (internal as well as external) open innovation.

Open innovation, also originating by communities, firms' business models and technological progress are connected in various ways. As a result, communities got an important source of potential innovation and directly influence the business models of companies following this open innovation approach. The community construct in open innovation is still raising questions and has in previous work often just been considered peripherally, leaving room for future research[West and Lakhani, 2008]. This work, focusing on open source communities, may therefore also contribute to this field and give important insights into the community aspects of open innovation.

2.3. Open Source Business Models

Already early a connection between technology, such as (open source) software, and business models has been identified. As an example, in 1998 Timmers [Timmers, 1998] lined out that technology will either be complementary to traditional businesses or will provide totally new approaches to make business. Furthermore, the emergence of new technologies provided new ways of value creation, enabling innovative business models [Amit and Zott, 2001]. Today, one can see a complex two-way relationship between business models and technology [Baden-Fuller and Haefliger, 2013]. Already on from the early 2000's open source got associated in various ways to business models, providing a potential benefit by introducing it in industry or building business models around it (e.g.,[Fink, 2003], [Gacek and Arief, 2004], [Chesbrough, 2006a], [Osterwalder and Pigneur, 2010],[Andersen-Gott et al., 2012]). Since the projects' output is publicly available, there are often just limited direct benefits for companies involving in open source, but indirect benefits and advantages in businesses leads firms to invest in such projects [West and O'mahony, 2008]. In addition, open source software projects have established themselves as an important driver for innovation by communities, also relevant for and supported by businesses [West and Lakhani, 2008, Andersen-Gott et al., 2012]. Especially innovation can get fostered by collaboration that is enabled through new technologies and virtual markets [Amit and Zott, 2001]. Whereas in former times especially firms, e.g. manufacturers, have been the force of innovation, also individuals and communities with the pure intention on personal benefit and without financial resources started to become compatible innovators especially in open source communities [Von Hippel, 2001]. This innovation by open source projects is ironically also protected by the open character, supporting business models around it. As an example, companies started to donate their intellectual property to the public, therewith reducing the risks of lawsuits and also lowering the costs for the company itself and also related businesses [Chesbrough, 2006a]. Today, one can see that free innovation communities and companies are increasingly getting connected, in the best case creating benefits for both [von Hippel, 2017]. Especially in technology industries we can see many firms currently profiting from a mixed approach of combining closed internal and open community-driven innovation, e.g. by having an open source approach for software development and closed approach for the hardware on that the software is shipped [Yun et al., 2016a].

However, the traditional business models that have been considered for open source research in the past are in most cases not necessarily fulfilled by the projects that produce software itselves, but enable firms to base a business model on them, e.g. by acting as distributor, providing additional services, or offering hardware [Fink, 2003, Krishnamurthy, 2005a, Andersen-Gott et al., 2012]. In general, open source business models can be defined as clear and distinct business models built around the open source movement [Chesbrough, 2006a]. Famous examples therefore are RedHat, Suse, or MySQL who built business models on open source projects that are also strongly driven by volunteer communities. Which exact business model can be built around open source software is dependent on the license, since copy-left or other restrictions may influence them [Krishnamurthy, 2005a]. However, all licenses, even Free Software licenses, do not restrict selling software, as long as the defined freedoms are given [Stallman, 2009].

Many open source projects receive commercial support or find themselves in ecosystems with companies and other partners, helping to drive the projects forward by e.g. stimulating the communities, employing developers to contribute or to build strategic alliances [Krishnamurthy, 2005a, Andersen-Gott et al., 2012, Ehls, 2017]. In these settings businesses in certain cases also see a moral obligation to contribute to such projects and give value back to the communities [Andersen-Gott et al., 2012]. Besides that, also firms or governments established open source projects, so called sponsored open source communities, themselves where defined goals are controlled by the founding entity [West and O'mahony, 2008]. In general, the chosen business model connected to or built around an open source project might affect which individuals, or also corporations, are engaging in these organizations [Gacek and Arief, 2004]. But still, as one can see by browsing through the projects' websites is that many projects are non-profit and just have loose tights to commercial businesses. The emerging definition of NBMs might also allow to see these projects, not following commercial goals but focusing on the community and its needs, from a business model point of view. This view can consider not only value defined as monetary profit, but also alternative forms of value that are present in these open ecosystems.

2.4. Open Source from a NBM Perspective

Open source gets developed by a diverse community of contributors, adding different values to these projects. These communities are well defined and known, but in many cases not arranged in a clear structure [Gacek and Arief, 2004]. However, analyzing these communities shows the complex ecosystem built around such projects [Vorraber et al., 2019a]. Leadership in open source communities is often fulfilled by the project's initiator [Nakakoji et al., 2002], but already Raymond highlighted that these initiators, in case of their retirement from the project, have to take care to define successor, ensuring the sustainable and long-lasting development of these projects [Raymond, 1999]. The absence of this leader role can bring a whole project to halt, till someone fulfills this role again [Ye and Kishida, 2003]. Therefore, it is necessary that project leaders, who do not have any formal authority in the meaning of giving instructions to contributors, are present to the community and provide and communicate an initial shared vision and goal that this community is following [Lerner and Tirole, 2002]. Failing to pay respect to the community, including it's greater goal, is a common source of failure in open source projects [Ehls, 2017]. In vital projects, the structure and evolution over time are collaboratively driven to the individual needs of the communities and their members [Nakakoji et al., 2002]. As an example, shifting open source projects to commercial business models potentially leads to project collapse [Ehls, 2017]. Thus, we can clearly see that intentions of profit-generation, in the terms of money, is not a driver for value creation in community founded open source. Furthermore, social aspects are strongly leading to a personal motivation to contribute to open source projects. However, it is important to note that sponsored open source communities, that are backed by a corporate entity, behave different since conflicting goals, i.e. control of the entity and openness of the community, must be balanced [West and O'mahony, 2008].

Although many open source projects get backed by paid contributors, volunteers play an important role for such projects [Riehle et al., 2014]. Regardless if paid or voluntarily, individuals contribute to open and also innovative projects if there is an immediate or delayed net-benefit defined as individual benefit (e.g., reputation, knowledge, or experience) minus costs (e.g., time, effort, or innovation) of contribution [Von Hippel, 2001, Lerner and Tirole, 2002]. Such communities innovate even without direct monetary compensation [Chesbrough and Appleyard, 2007], making it necessary to understand their motivation that generates the mentioned benefit. Hence, various motivations to participate in open source communities have been identified in literature in the past and highlighted its complexity and impact on open source projects [Von Krogh et al., 2012]. Similar to this described contributors' net-benefit, the outcome of a value oriented win-win situation is also important for new business models [Jonker, 2012]. In the case of different stakeholders and also users, net benefit as described by [Delone and McLean, 2003], needs also to be ensured in the given context of value creation. As a result, it is necessary to know all actors involved in the open value creation and capturing process and their motivation that links to their individual received net benefit.

Each community has a unique structure of different formal or informal roles that are either well defined or just loosely connected [Nakakoji et al., 2002, Ye and Kishida, 2003, Gacek and Arief, 2004]. In general contributors can be characterized by their contribution. Each community has a core that is responsible for most of the code and has the biggest influence on the community and the future direction of the project, whereas peripheral contributors and users play a less important, but still essential, role [Crowston et al., 2006]. Especially regarding these peripheral developers, one can see that in such communities there is a large number of contributors that just make one commit (code contribution) to the project and then disappear [Lerner and Tirole, 2002, Ye and Kishida, 2003]. Contributors don't get formally assigned to a role, they rather decide on their own how much effort and work they want to contribute, enabling them to evolve to more responsible roles over time, but also to leave the community at any point [Nakakoji et al., 2002, Ye and Kishida, 2003, Gacek and Arief, 2004, Von Krogh et al., 2012]. Furthermore, as also the described open source software based business models have shown, such projects are connected to a variety of external stakeholders, resulting in this complex ecosystem of different relations and multiple (intangible) values exchanged [Vorraber et al., 2019a].

Bringing together contributors, users and other stakeholders, e.g., involved businesses supporting a project, got essential for the success open source software. Open source organizations, driving innovation, not only consist of a community of different individuals, but for various reasons also firms actively engage, either directly or indirectly, in these ecosystems [West and Lakhani, 2008]. Companies directly stimulate communities to innovate, but also give something back to them [Andersen-Gott et al., 2012]. The same co-creation process and transfer of various values by various actors seems to take place in NBMs. Different interests need to be balanced in a way to ensure socio, economic and ecological goals of a business. Therefore, holistic stakeholder analysis and understanding got vital to create new business models. Tools such as the Power Versus Interests Grid [Ackermann and Eden, 2011] as described in [Bryson, 2004], or the Stake Model of a Firm [Fassin, 2009] may support stakeholder identification and analysis. Analyzing stakeholders and actors of open source ecosystems unveils the complex relationships and highlights the co-creation of value [Vorraber et al., 2019a]. Such manifold relationships in ecosystems, collaboratively creating value, got common in businesses and foster innovation especially in the current digital age. Therefore, current research on business models is increasingly focusing on the collaborative multi-value creation by various interconnected stakeholders.

Especially since value is co-created, no direct ownership exists in new business models. Instead of ownership, these models focus on the aspect of access [Jonker, 2012]. This aligns with the before described principles and movement of free software. The freedoms of software ensure this access and even extend it by the rights to modify and redistribute it. As Chesbrough puts it [Chesbrough, 2006a], no one owns the right to exclude others from using open source technology. This goes along with the principle of open source software to not necessarily write code on your own, but also reuse code of others [Raymond, 1999]. This leads to the conclusion that the defined characteristics of access over ownership has already been present in open source communities within the last years. This aspect also already required firms to adopt their processes when they introduced open innovation, making it necessary to change their business models to the new open character [Chesbrough and Appleyard, 2007].

The co-creation of value by different contributors in a community, the exchange of intangible values in the ecosystem, following a vision that is not founded on monetary profit or business objectives, and the principle of openness over ownership are strong characteristics shared by both practical oriented research strands. By outlining the characteristics of these domains we can see similarities between open communities and new business models that are heavily influential for both domains.

2.5. Discussion

Current trends in business model research show that there is high potential for open source ecosystems of contributors, users, and stakeholders who are collaboratively creating value to be considered as a new business model. Existing and widely used frameworks to represent general business models lack in regarding the before mentioned alternative definition of value that gets co-created and distributed by communities of individuals that participate for a variety of different reasons in this process. Although research on NBMs is still in its beginning, the current state indicates that this might impact the understanding of how open communities create value not only from a profit, respectively commercial, viewpoint, but also from a social or individual viewpoint of the involved actors. This leaves room for further research in combining these domains, allowing to see open communities not only to enable firms to build business models on them, but as (new) business models themselves.

The created intangible net-benefit in open source communities can be analyzed with these upcoming models and help practitioners to gain a better understanding of their ecosystems, as well as firms to understand the communities they are working with in their innovation process. Not only monetary value is generated by the developed software or innovation, but also an individual net-benefit for each actor involved in this co-creative value generation and distribution process. The co-creative approach potentially involves a variety of actors, e.g. individuals, firms, or governments, that work jointly towards a shared, as well as different individual goals. This collaborative approach can be captured by new representations of business models that might help to also explain the goals of a community, motivation of the individual contributors, and co-creation of mostly intangible value, such as innovation.

Till now, NBMs often focused on sustainability issues and stakeholder relations. But as this first review outlines, doing research on new business models is also a potential chance for researchers in the area of collaborative open source organizations and open innovation approaches. These open communities are more and more situated in complex ecosystems of individuals, firms, users and other partners, all creating, but also expecting, value. This co-creative nature, based on needs and individual motivations, can be a role-model for other communities not necessarily only connected to ICT, but due to their social nature of sharing and collaboration also for cases in the emerging field of NBMs. The state of art in these domains shows many similarities when it comes to the research strands' main characteristics. Although, the sustainability character outlined in most work about NBMs is not explicitly related to open source software development, the social aspect strongly is. This aspect leaves room for further research in combining the fields in explicit cases, targeting to gain further insights into the social character and the intangible needs originating from the individuals' motivations in communities, that results in the collaborative creation of various values for all entities in the connected ecosystem. Not only open source projects can benefit from these insights, but also other open communities that drive free innovation or create alternative kinds of value. Open innovation got common in different industries, facing different challenges that still need to be solved. The outlined application of the proposed tools for NBMs in a variety of such open communities is expected to give further implications how researchers and practitioners can understand their value creation and delivery process that lead to innovation and their sustainable long-term existence.

2.6. Conclusion

As we showed, current economic research is going towards a new understanding of business models and the creation of value. Involvement of various actors, creating multiple values and replacing money as main currency of value can get identified as key points of emerging new business models. This still developing field has the potential to also provide new insights into information systems, especially related to the collaborative development of open source software. Communities driving the movement of open source have been mainly seen in literature as enablers of business models that generate profit for firms in a project's ecosystem. But in recent years also businesses started to sponsor and support these innovative communities themselves, making open communities an active part of their ecosystems. New trends in business model research allow to see these open communities through a new perspective, co-creating and realizing different values for several involved actors. We showed that both research strands, open source communities and NBMs, are following trends of collaboration in heterogeneous ecosystems that heavily depend on intangible values often also connected to social and individual needs. Representations and further definitions of NBMs have the potential to foster the understanding of the therewith connected mainly non-monetary values transferred, that is essential for the viable management of open ecosystems. The growing involvement of firms in these ecosystems, actively driving innovation, makes it necessary to gain a better understanding of the processes that cannot be represented with previous business model notations, not considering social and other non-economic aspects.

Therefore, based on current literature, this work is pointing out the chances that are provided by combining research on (open) information systems and new business models in practice, to enhance the understanding of both research strands. Tools and frameworks that come up with research on NBMs can be applied to open communities and ecosystems, helping to see them from a holistic point of view and various relevant perspectives for the communities. Although the focus of this work has been on open source communities, it indicates to be also valid for the related field of open innovation communities. This review is intended to build the theoretical basis for action based research, initiating cases that take into account the outlined similarities between the two domains of NBMs and value creating open communities.

3. Gaining the Networked Perspective of (New) Business Models

3.1. Analyzing and Managing Complex Software Ecosystems[1]

We can see that especially software projects, commercial as well as free open source ones, became complex ecosystems of various contributing parties following heterogeneous benefits and expectations. Understanding and managing those projects, which is key for their success and viability, consequently becomes harder and requires special approaches and framework to analyze and align them adequately. A variety of different actors, representing managers, developers, users, or stakeholders, are actively involved in these ecosystems for individual reasons and needs, resulting in dynamic relations and exchanges of values. Especially in open source projects, each contributing actor has its unique role and motivation to contribute [Hars and Ou, 2002, Oreg and Nov, 2008, Von Krogh et al., 2012], ultimately expecting a certain (individual) net-benefit for their contribution [Lerner and Tirole, 2002]. This also aligns with more general research in this field, pointing out that the degree of motivation to contribute to a (value) network is dependent on if the needs of an involved actor are met or not [Vroom, 1964, Porter and Lawler, 1968]. Thus, to run a software project in a successful and sustainable way, it is necessary to introduce frameworks to understand this value-process of creation and capturing, not only from a just tangible,

[1]This section is based on [Vorraber et al., 2019a] ©2019 IEEE. DOI: 10.1109/MS.2018.290100810

but also intangible perspective for every actor and relation. To achieve this, visualization tools can help to communicate and analyze the underlying value networks. These value networks not only cover the complex exchange of tangible values (e.g., products or money) between groups of people or organizations, but also pay respect to intangible values (e.g., knowledge or information) [Allee, 2009]. Therefore, they represent an adequate approach to gain a deeper understanding of the influence and exchanges of values within ecosystems.

Through applying an already existing notation, the V^2-notation (as introduced in [Vorraber and Vössner, 2011]), we can analyze this complexity and help to visualize, communicate, and ultimately understand ecosystems to base further management actions on it. V^2 is derived from Biem and Caswell's notation (see [Biem and Caswell, 2008]) and extends it in various ways. Most importantly, this extended notation regards the intrinsic motivation of actors, but also external influences that might have an impact on value within the system. This helps to identify potential actors that can negatively impact whole networks, so called *bottlenecks*, but also *value engines*, representing reinforcing-loops of value creation and exchange, that might positively impact ecosystems. These aspects are also related to possible network effects, that may emerge through the dynamic exchanges in the ecosystem. Managers therefore need a holistic understanding for their strategic decisions, taking the biggest benefit out of the ecosystem.

The V^2 framework is primarily designed to provide a graphical representation, making it easier to create this needed holistic understanding. In this representation, actors are represented as circles, each having a unique name, capabilities, and assets. Actors are connected through directed links, that either represent a provision (solid line with an arrow of the direction) or a revenue (dotted-line with an arrow of the direction) relation. These links get further specified through the particular values that are exchanged between these connected actors. These values might be tangible (e.g., a product or money), or intangible (i.e., information or coordination), and are predefined trough colored triangles representing certain possible clusters of value (e.g., product or coordination). An extension to this framework, also further layers, is described in the following Section 3.2, also including this described visual notation as *Value exchange and resources layer* in Figure 3.2.

The goal of this framework is to provide a way to analyze complex and highly dynamic ecosystems (current and future ones), identify/anticipate bottlenecks and value engines therein, create measurements to improve ecosystems based on network analysis, and align strategies to meet the needs of the involved actors. Hence, the V^2-framework is especially for practitioners a handy tool to visualize, analyze, communicate and improve ecosystems.

To apply this approach in practice, six steps have been defined that can frame the basis for strategic management decisions:

1. **List Actors:** Listing all actors that are involved in the value creation and capturing process.
2. **Identify Actors' Needs:** Since each actor participates in the ecosystem for a specific reason, these reasons, representing individual needs, must be understood and captured.
3. **Specify Value Exchanges:** Values are transferred within the network. Therefore, linking the actors, representing these value-exchanges, and furthermore indicating the direction of the connection, allows to draft the value network as graphical representation in the means of the V^2-notation.
4. **Check Net-Benefits:** Managers need to analyze if all actors are satisfied, consequently creating a net-benefit for their contribution to the ecosystem.
5. **Identify Bottleneck Situations. & Winner Relations:** Highlighting potential flaws in the network (e.g., unmet needs or too low created benefit for individual actors) that might impact the ecosystem negatively (*Bottlenecks*). But also create awareness of positive dynamics that occur within the ecosystem (*Winners*).
6. **Seat measures to improve the ecosystem network**: Elaborating measures to improve identified bottlenecks and to reinforce existing winner effects.

The possible outcome (i.e., the visualized value network and derived possible measures) of this approach for the Catrobat project in practice is illustrated and further analyzed in Chapter 6.

3.2. A Networked approach for New Business Models[2]

A focus on the networked aspect of business models (i.e., within ecosystems) is also emerging in the field of sustainable business models, including the aforementioned *New Business Models* (NBMs). Conventional and common graphical business model frameworks, such as the Business Model Canvas [Osterwalder and Pigneur, 2010], serve well to formulate and easily communicate an ego-centric view of business models [Breuer and Lüdeke-Freund, 2014], but they are situated on a rather abstract level, resulting in their weakness of not being able to present an in-depth understanding of dynamics and network externalities, that can be found in sustainable business models [Massa and Tucci, 2013]. Also, more recent research in the field of business models [Massa et al., 2018] shows that many businesses are today situated in complex systems that increasingly include social components, such as human interactions, organizations, or the society. Common business model notations often do not cover these components, further challenging them to represent such systems from all needed dimensions. Whereas the general aspect of the increasing importance to cover networks in complex systems with a focus on innovation is described in more detail in Section 3.3, we show here an enhanced approach suited especially for systems situated in NBMs or similar domains (e.g., OSS projects or community driven businesses), that require additional perspectives (e.g., the mentioned social one) to fully understand them.

Reviewing current frameworks in the research strand of sustainable business models unveils that almost all of them include either implicitly or explicitly a networked perspective as part of their business model notation. This present perspective highlights the aforementioned importance of networked effects for sustainable business models. Nevertheless, as our analysis has shown (for the detailed analysis and comparison see [Vorraber and Müller, 2019]), these notations and models only partially provide a visual and easy way to communicate and analyze the underlying networks and their dynamics. We differentiated between an implicit representation (a networked perspective

[2]This section is based on [Vorraber and Müller, 2019] CC-BY. DOI: 10.3390/su11216018

being included as a concept or theoretical underpinning) and explicit representation, that also includes it in a visual way. Only one (the *Triple Layered Business Model Canvas* [Joyce and Paquin, 2016]) out of eight reviewed NBM innovation frameworks does not include a networked perspective at all. In contrast, four frameworks (*Strongly Sustainable Business Model Canvas* [Jones and Upward, 2014], *Sustainable Business Canvas*[Tiemann and Fichter, 2016], *BMC extended for infrastructure* [Foxon et al., 2015], and *Value triangle and VT BMC* [Biloslavo et al., 2018]) include this perspective implicitly and further three frameworks (the *Value Mapping Tool* [Bocken et al., 2013], *Values-based Innovation* [Breuer and Lüdeke-Freund, 2017, Breuer and Lüdeke-Freund, 2017], and *Visual Coding Scheme for Sustainable Business Models* [Brehmer et al., 2018]) also explicitly. We further evaluated those frameworks by the type of visual representation as categorized by Täuscher and Abdelkafi [Täuscher and Abdelkafi, 2017] (i.e., *Component Based*, *Transaction Based*, and *Causality Based*) and the aspect of stakeholder needs (i.e., explicating needs and explicating satisfiers). This evaluation showed that, except the *Visual Coding Scheme for Sustainable Business Models* (which is transaction based and does not explicate the stakeholders' needs), all representations are component based (i.e., the notations provides a basic static structure to be filled out) and are explicating stakeholders' needs and satisfiers. This brings to light the need for a further framework, complementing the existing ones, that allows an in-depth analysis of all those aspects, also regarding the networked aspects of value exchanges between actors (instead of a solely static component based view). Rather than defining such a new framework from scratch, we introduced a further layer-based enhancement to the V^2-notation, that has been described in the previous section.

As illustrated in Figure 3.1, additional layers are added to the notation described in Section 3.1. In the need for a holistic picture of sustainable business models, further perspectives complement the aforementioned basic elements of the V^2-notation (referred to as *Value exchange and resources layer*). After modeling the system with this notation, depending on the purpose of analysis, different or all layers can be regarded to gain further insights into the represented network, also supporting an easy communication by only illustrating parts relevant for specific stakeholders if needed. It is important to note, that this framework shall not replace, but built upon and complement existing frameworks (e.g., component-based business model

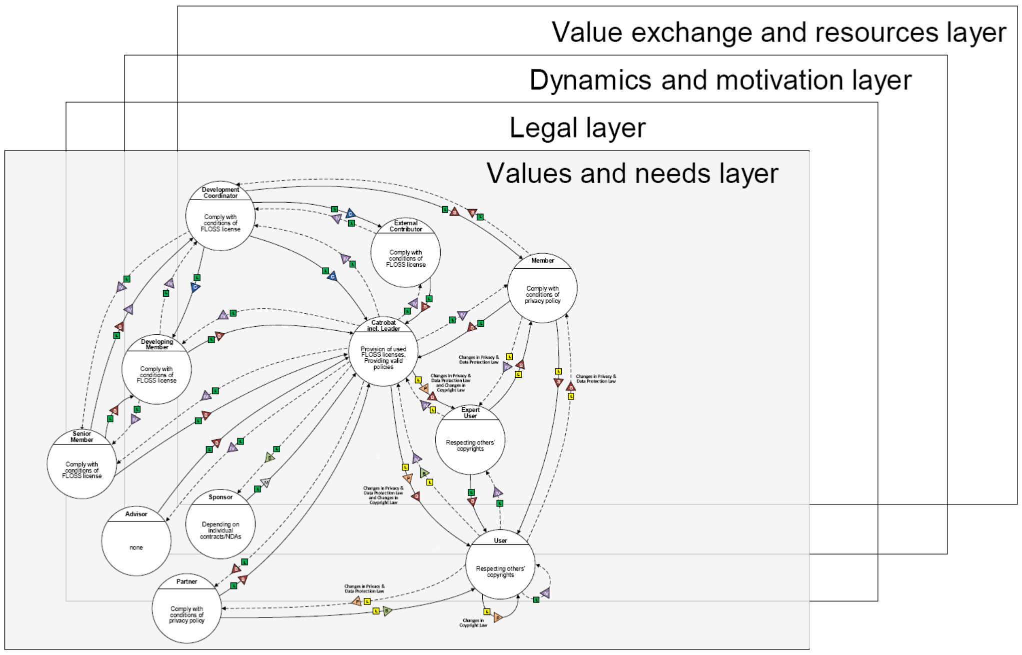

Value exchange and resources layer
Dynamics and motivation layer
Legal layer
Values and needs layer
Development Coordinator
Comply with conditions of FLOSS license
External Contributor
Comply with conditions of FLOSS license
Member
Comply with conditions of privacy policy
Catrobat incl. Leader
Provision of used FLOSS licenses, Providing valid policies
Developing Member
Comply with conditions of FLOSS license
Senior Member
Comply with conditions of FLOSS license
Expert User
Respecting others' copyrights
Advisor
none
Sponsor
Depending on individual contracts/NDAs
Partner
Comply with conditions of privacy policy
User
Respecting others' copyrights
Changes in Privacy & Data Protection Law
Changes in Copyright Law

Figure 3.1.: The layer structure of the enhanced V^2-notation illustrated on the case of Catrobat (this case will get described in detail in Section 6.2 (Figure as published in [Vorraber and Müller, 2019] CC-BY)

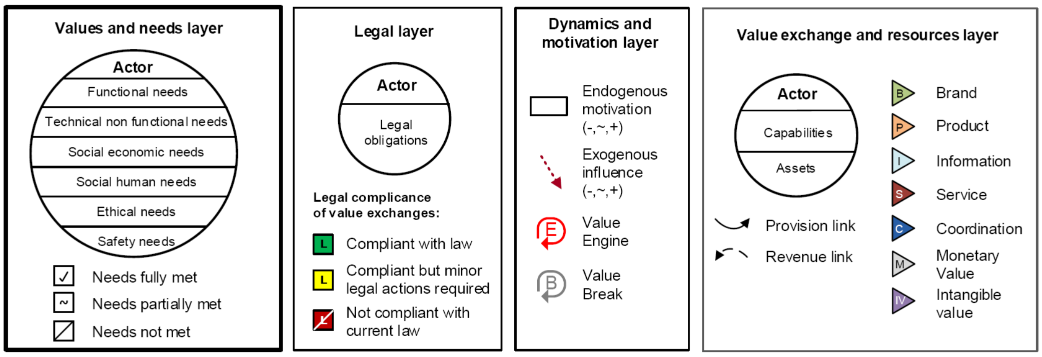


Figure 3.2.: The enhanced V^2-notation described for all four layers (Figure as published in [Vorraber and Müller, 2019] CC-BY)

notations) by providing a further possibility to analyze and represent the networked perspective of an ecosystem in one final big picture.

Whereas the visualization of circles representing actors and arcs representing relations is the same for all layers, they differ in how they are structured. In Figure 3.2, the layers are described in detail. Although parts of this framework have already been published previously, the holistic picture that is created through combining the layers is new and provides additional possibilities for practice. Despite that creating these layers in a proposed iterative expert-workshop setting is time consuming, we believe and have seen on the applied cases that creating such a big-picture of a value network has several benefits for analyzing and managing the related ecosystems. The provided details help to gain a deeper knowledge about how value processes work in the analyzed ecosystem and what aspects must get considered therefore. Hence, the layers provide following insights:

- **Value exchange and resources layer:**
  This layer basically represents the aforementioned V^2-notation that enhances Biem's and Caswell's notation [Biem and Caswell, 2008] and has been described in Section 3.1. It covers the involved actors, their assets, capabilities and their relations within the network. This allows to understand what resources are present and what exact values are exchanged between the involved actors. It is important to note that his not only includes tangible values, such as products or money, that get exchanged, but also intangible values such as information or knowledge.

- **Value and needs layer:**
  This layer is intended to provide a deep understanding of the needs of the actors within the represented system. Therefore, these needs are further classified in categories that are derived from existing literature and may be used for further analysis.
  - *Functional needs (FN):* The functional aspects of a system, i.e., what should be possible to get accomplished [Partsch, 2010]. As an example, a specific functionality or process that is needed to get a job of the actor done.

  - *Non-functional needs:* The *human side* of needs, that can get divided into:
    * *Technical non-functional needs (TNFN):* The non-functional aspects a system/service must provide to the actor (e.g., handling, quality, or design). (based on [Rupp, 2009])
    * *Social economic needs (SEN):* How an actor wants to be perceived by others in economic terms (e.g., feeling better than others). (based on the concept of social jobs by [Osterwalder et al., 2014])
    * *Social human needs (SHN):* The need of doing something good for others, i.e., focusing on an actor's external environment in a societal, economic and ecological sense. (based on [United Nations World Commission on Environment and Development, 1987, Pavie et al., 2014])
    * *Ethical needs (EN):* Complying with an actor's ethics theory [Pavie et al., 2014]. In contrast to the SHN, this is focused on the actor's personal ethical theory (e.g., what personal information of the actor are communicated to others).
    * *Safety needs (SN):* Covering the actor's need for safety when using services of the system (e.g., work or consumer safety).

- **Legal Layer:**
  Identifying the legal status of relations and values exchanged can be crucial for innovation processes and business models. Compliance to law (e.g., GDPR, environmental law, etc.), must therefore be incorporated early. As illustrated in Figure 3.2, the legal layer covers every actor with its legal obligations and furthermore states the legal compliance of every relation between actors. These relations are either fully compliant to law (visualized as green *L*), may require minor legal actions (yellow *L*), or are currently not compliant with law (red *L* in the notation) [Vorraber et al., 2016]. This assessment is intended to be made by experts in the field, who also consider measures that might get necessary in the future. The thereout created visual representation can then also be communicated easily to other actors without deep domain-knowledge, helping all involved stakeholders to understand the legal status of an ecosystem.

- **Dynamics and motivation layer:**
  Personal and organizational motivation can be seen influenced by internal and external forces [Vroom, 1964, Porter and Lawler, 1968] and must therefore be regarded in such a complex system. This perspective adds to the framework the *endogenous motivation* (i.e., the personal outcome that an actor receives for its participation) of and the *exogenous influence* (i.e., external forces influencing the actor) on certain actors. As depicted in Figure 3.2, these aspects are differentiated whether they are defensive (-), neutral ($\sim$), or active (+). Analyzing these motives can help to understand potential *value engines* and *value breaks* that may either positively or negatively impact the exchanges and dynamics within the value network. [Vorraber et al., 2019a]

Drafting these layers allows to also consider social und sustainable aspects in ecosystems, ultimately helping to gain a holistic picture of them. This provides the possibility to illustrate, communicate, and apply a deep understanding of all value processes happening in a networked setting within one picture, that gets build in workshops (with experts and stakeholders) or partially derived from the outcomes of previously applied tools from the domain. This is in particular of interest, when it comes to management decisions or future strategies/processes of a business (or organization), that not only affects one entity, but a whole complex system of actors that are dynamically connected in different ways (e.g. economic, social, sustainable, etc.) and impacted by potential network effects.

3.3. Getting the Networked Perspective of Business Models: The Road to Innovation?[3]

3.3.1. Introduction

Current developments in information systems show that today's businesses are getting more and more connected, are practically depending on platforms or complex ecosystems of various partners to generate and capture value. The underlying (system) complexity, that increased in importance the recent years, also effects business models, since more aspects must be considered to ultimately understand how they work [Massa et al., 2018]. Innovation, especially related to technology and business models are connected in various ways and interact regularly [Baden-Fuller and Haefliger, 2013]. Various examples from today's industries outline that innovative technologies and business models, especially related to digital services, challenge the classic value chain logic and are situated in value networks instead [Peppard and Rylander, 2006]. This also underpins the findings of Wirtz et al. [Wirtz et al., 2016], who identified that *Network Models* currently have the highest relevance for researchers in the domain of business model research. Hence, we can see that innovation also requires a new, broader and more complex, dynamic view on business models.

Consequently, to succeed in today's economy, managers and decision makers must not only consider a chain, but a whole complex ecosystem, represented as network of different actors that create and capture value within their business model. These networks consist of various types of stakeholders, including users and external actors, who are either directly or indirectly involved in it. Furthermore, the created value and the connections are increasingly based on a sustainable definition of value (e.g.,covering environmental or societal aspects), making them harder to measure and understand. However, managing such a network the right way, may be the key to success as today's innovative businesses show.

[3]This section is working paper, "Getting the Networked Perspective of Business Models: The Road to Innovation?", by Matthias Müller and Wolfgang Vorraber, submitted to the *International Journal of Innovation Studies* by *KeAi* in December 2019

Due to the new possibilities of connectivity, we can see that innovative business models that solely depend on the relation between enterprises and customers became rare. Ecosystems in that value for customers is created jointly by coordinating the actions of various actors, not only businesses but potentially also customers, became state of the art. Famous examples are Amazon's Marketplace, Uber, Airbnb, or also open communities such as in open source projects. Therefore, the need of understanding the value creation process, that is illustrated by business models, can often just barely be represented by currently common notations. Graphical frameworks or archetype, that are common in practice because of their simplicity, lack in covering dynamics, resulting in the need for meta-models and activity-systems of lower abstraction to represent more complex business models [Massa and Tucci, 2013]. But still, to quickly get an understanding of the overall value process, which is essential for management decisions or the planning of new business models, also these aspects of relations, dependencies, and dynamics must be easy to communicate and analyze. From our experience, creating such an understanding within one easy tool is challenging and comes at the cost of representing all needed aspects deeply enough. Therefore, we describe in this work how commonly used business model notations may easily be complemented by a more holistic network view (i.e., through value network notations) that may help to identify opportunities and challenges in the investigated value creation process within ecosystems. We review aspects of existing notations, their implications and relations, ultimately showing new directions for research in innovation by combining existing tools of different abstraction levels in practice.

The implications of this approach can quickly be illustrated through currently trending cases from practice. The emergence of digital marketplaces for products or services in the world wide web made it easier than ever before to connect customers and businesses globally. This emergence of new ways of distribution also led to the emergence of innovative multi-sided and platform business models. Famous and successful examples that are often noted are the rise of Amazon, the emergence of Google's and Apple's app marketplaces, or whole platform business models that emerged on digital advertising. In general, these multi-sided business ecosystems can be categorized into 'social network' (e.g. Facebook), 'merchant' (e.g. Alibaba), 'service platform' (e.g. Airbnb) and 'application platform' (e.g. Play-Store) business models [Schreieck et al., 2018]. Although many of these approaches are not new [Baden-Fuller and Haefliger, 2013], they reached a dimension through the Internet that has not been existing before. This progress was also fostered by the tendency of not only relying on internal innovations, but also opening organizations, co-creating value, and a new way to interact with customers.

Business model notations got a common tool to illustrate and analyze these new possibilities of value creation and ultimately to investigate how businesses can innovate towards now still unknown possibilities. Drafting a business model by configuring a unique setting of resources and processes may be the key to succeed against competitors [Johnson et al., 2008]. Therefore, applying frameworks such as the Business Model Canvas [Osterwalder and Pigneur, 2010] got an important step in practice to drive change and innovation. However, especially when it comes to technological innovation a broader view on the connected ecosystem is needed, in which value must be created for and captured by all involved parties in the business model [Baden-Fuller and Haefliger, 2013]. Through three examples from common and well known domains in Section 3.3.4, we line out how existing notations of business models and value networks can be combined in practice, to get a holistic understanding of the broader perspective of a venture. By purpose, we refrain from proposing the usage of specific tools or notations, but instead show a general approach that can easily be adapted to be used to extend existing processes in management, allowing to get an additional ecosystem point of view in business models. This decision is based on the high number of available and already used tools, that can be categorized in different views that are potentially complementary [Täuscher and Abdelkafi, 2017]. Consequently, our aim is to demonstrate how abstract representations of business models can be easily lead over to more dynamic network-based representations, helping to understand value in innovative ecosystems by investigating several levels of abstraction. Whereas the theoretical frame for research of this conceptual approach is still under investigation, its practical application already brought up interesting insights and results in the investigated cases, that will be briefly discussed in this work.

3.3.2. The Networked Perspective in Business Model Notations

In recent years business models (BMs) and their connection to innovation not only came into the focus of researchers, but increasingly also of practitioners through easy to use tools and frameworks [Zott et al., 2011, Baden-Fuller and Haefliger, 2013]. Therefore, a large variety of definitions, models, and examples can today be found in literature. For this work, we follow the simple definition by Chesbrough [Chesbrough, 2006a] that a business model describes the two functions of how value is created and captured. Nevertheless, it needs to be mentioned that a large variety of articles on business models exists and no common and broadly used language for this domain is used by scholars [Zott et al., 2011]. Already in early definitions, BMs included the aspect of co-creation within the different definitions and notations. Analyzing the network-component of early business model concepts (between 2000 and 2010), brought up a moderate intensity of use for this aspect, however, it was also highlighted as most important strategic component for future work [Wirtz et al., 2016]. We reviewed some of the current available and commonly used definitions and frameworks, to highlight this networked perspective in BMs. This listing is far from complete, but should represent the different approaches that can be used to include a network representation in BMs. But still, although all these definitions cover this perspective, they vary in the depth of representation, created understanding and visualization. The majority of available visual frameworks represents an element-based view rather than a transactional- or causal-based view [Täuscher and Abdelkafi, 2017]. This can be seen in relation to the level of abstraction of BM representations, leading to those different approaches. More abstract models (i.e., archetypes and graphical frameworks) are popular because of their simplicity to get drafted and communicated, however, more complex and sophisticated (meta-)models may be needed to understand the whole system[Massa and Tucci, 2013]. How such different models can be connected (independent of the exact notation used), will be drafted in the cases illustrated in this work.

Whereas Amit and Zott [Amit and Zott, 2001] consider the whole Transaction-Structure (including the participating parties and how they are linked), Gordjin and Akkermans [Gordijn et al., 2000], include the network perspec-

Year	**Author(s)**	**Role of Ecosystem**
2000	Gordijn and Akkermans [Gordijn et al., 2000]	Actors as individual economic entities doing value activities
2001	Amit and Zott [Amit and Zott, 2001]	Transaction Structure (involved parties and how they are linked)
2006	Chesbrough [Chesbrough, 2006a]	Value Network as a function of a Business Model
2008	Johnson, Christensen and Kagerman [Johnson et al., 2008]	Partners and Alliances
2010	Osterwalder and Pigneur [Osterwalder and Pigneur, 2010]	Listing Partners in value creation
2014	Zolnowski et al [Zolnowski et al., 2014]	Covering all dimensions from different perspectives (i.e., Partner, Company, and Customer)

Table 3.1.: Examples how a network perspective for the ecosystem can be covered in a business model definition

tive through considering (several) actors as individual economic entities involved in the value activities. Chesbrough [Chesbrough, 2006a] highlighted ecosystems, or respectively value networks, as a function of a business model that links suppliers, customers, potential complementors and competitors with the business. Also Johnson, Christensen, and Kagermann [Johnson et al., 2008] included partnerships and alliances as a key resource within their business model notation. The probably currently most known and used graphical framework in the domain of drafting BMs, the Business Model Canvas (BMC) [Osterwalder and Pigneur, 2010], defines key partnerships as a building block, lining further out that partnerships in business models may emerge for a variety of different reasons and motivations. This partner perspective becomes particularly important in the current times of change, in which business models shall create value for all participants [Amit et al., 2010]. Furthermore, as previously noted, value is not anymore created in a value chain, but in whole networks of different actors [Peppard and Rylander, 2006]. Specifically, the relation between business models and new technologies enables new possibilities for value creation, such as the two-sided platform implemented by Google Adwords [Baden-Fuller and Haefliger, 2013]. Such multi-sided platforms are not new, emerged early in different markets, e.g. also credit cards and shopping, and are dependent on initially getting the needed partners of the platform on board, but also on keeping them, which both is connected to certain network effects [Evans, 2003].

As such examples highlight, partners and ecosystems got an essential element of value creation and require a holistic perspective, that also allows to create new (innovative) business models. But, this holistic perspective, paying respect of the interconnected network structure, is often missing in existing models and notations in terms of visualization and ability to get easily communicated. Thus, the emergence of new service oriented business models which are dependent on a platform, or respectively network of different actors, challenges common (abstract) notations for business models. As regarded in the *Service Business Model* approach of Zolnowski et al. [Zolnowski et al., 2014], especially in services frequently used notations become indistinct or just partially applicable, since the business models' value creation and capturing requires to get considered from each actors' perspective. To illustrate such cases, also Osterwalder and Pigneur [Osterwalder and Pigneur, 2010] introduced for the BMC the pattern of multi-sided platforms that can be used for business models such as advertising, open source dual licensing, or news-platforms. However, this pattern includes various customer segments with different value propositions and other key elements within one single graphical canvas in which the different segments are differentiated by colors. Although, the BMC is well suited to give a quick overview if these segments are rather similar or just a limited number of segments is considered, in practice its application is limited when it comes to scaling these business models or to include further essential active partners in the value creation process as well as how they are linked. First, the canvas gets quickly confusing when using this color scheme within one representation, second, network effects cannot be illustrated and analyzed. Same applies to other frameworks in this domain, that primarily focus on an egocentric view of business models. Thus, further tools are needed in practice to regard these network aspects and put them into relation to business models. Value networks became increasingly beneficial as concept and tool with the emergence of mobile and connected services for which value chains are not capable to describe the co-creative nature of value anymore [Peppard and Rylander, 2006]. Currently various frameworks and notations such as e3 [Gordijn et al., 2000], Biem and Caswell [Biem and Caswell, 2008], V^2 [Vorraber and Vössner, 2011, Vorraber et al., 2019a], Allee's value network notation [Allee, 2009], or Becker et al. [Becker et al., 2011], each having a different level of details and focusing on different aspects, for value networks exist. Value networks are built on autonomous units represented as nodes, that might be suppliers, partners, allies, or customers, that are connected through links and operate together [Peppard and Rylander, 2006]. These notations and the therewith connected conceptual frameworks are powerful tools for managers to align actors and visually communicate a common understanding of an ecosystem [Vorraber et al., 2019a]. In particular the dynamics in ecosystems, the fact that each relation may affect the others, is enhancing the value chain perspective considered in the past [Peppard and Rylander, 2006] and therefore of importance for current (service) business models. In this work, we introduce a practical approach that still takes use of commonly used notations and models, but puts them into a more sophisticated context within co-creative value networks as complementary and less abstract construct to illustrate business models within ecosystems. The focus of this approach is to still keep the ease of use and benefits of clarity and communicability of more abstract frameworks, which is essential for their practical application, but include it into a bigger picture of the ecosystem that potentially helps to gain new chances and identify challenges for existing or proposed business models.

3.3.3. Related Domains

To underpin the importance of this interplay of business models, innovation and ecosystems, we want to highlight two related research strands, that both are still gaining importance. *Sustainable Business Models* and *Open or User-centered Innovation* are research strands that are strongly related to service innovation, but also challenge common abstract and egocentric business model notations. Both domains have recently been in the focus of scholars and show highly relevant directions for further research.

Sustainable Business Models

The new business model (NBM) community, which focuses on strongly sustainable business models, is a research strand, that is largely based on a networked perspective on business models (c.f. [Vorraber and Müller, 2019]). The term strongly sustainable denotes business models that also emphasize social and ecological aspects of business models in addition to economic aspects. Hence, Upward & Jones [Upward and Jones, 2016, p. 103] define a firm with a strongly sustainable business model as "[...] an organization that only enabled strongly sustainable outcomes as one that creates positive environmental, social and economic value throughout its value network, thereby sustaining the possibility that human and other life can flourish on this planet forever (Ehrenfeld [Ehrenfeld, 2000]; Willard et al. [Willard et al., 2014])." In a larger context, this community strives to facilitate the development of a sustainable society as outlined in a report of the United Nations, commonly known as the Brundtland Report [United Nations World Commission on Environment and Development, 1987]. Various frameworks to foster the creation of sustainable business models such as the Strongly Sustainable Business Model Canvas [Jones and Upward, 2014], the Value Mapping Tool [Bocken et al., 2013], the Values-based Innovation [Breuer and Lüdeke-Freund, 2017, Breuer and Lüdeke-Freund, 2017] framework, or the Triple Layered Business Model Canvas [Joyce and Paquin, 2016] exist [Breuer et al., 2018]. The analysis of selected frameworks in [Vorraber and Müller, 2019] showed that the majority of these frameworks are based on a network-centric perspective of the business ecosystem, but mainly use a component based visual representation rather than a representation also explicating transactions or causal relations between ecosystem entities. This may lead to similar limitations in communicating aspects such as value exchange dynamics in ecosystems as outlines in section 3.3.2 for component based frameworks such as the BMC.

Open and User-Driven Innovation

As described by von Hippel and Jin [von Hippel and Jin, 2009], recent years have shown that in terms of innovation we moved from a producer model, in which producers develop innovation that are protected by policies and laws, towards a user-centered approach, in which innovation originates from (lead-) users, which requires new policies and processes. Although it seems to be irrational that not producers innovate new products, various examples show that user-innovations, that happen when users expect their benefit of innovation exceed their costs, can compete with industry-products, even if those innovations get openly and freely shared for the public [Von Hippel, 2001]. This opening of innovation processes ultimately leads to open innovation, covering concepts such as user innovation, crowdsourcing, or open source innovation, crossing existing boundaries, what enables firms to increasingly use external knowledge and shorten product life-cycles [Yun et al., 2016a]. This is especially true in today's technological economy. Firms must figure out how internal and external innovation can be integrated into a system, which can be done through a business model [Chesbrough, 2006c]. A famous example therefore from out the technology-domain, which is also often investigated by scholars (e.g., [Von Hippel, 2001] or [Chesbrough and Appleyard, 2007]) and will also be used as one case for our work, is open source software. Such open value-creating communities challenge conventional BM representations, since the motivation and values transferred are often intangible in a highly complex network of different actors, also following social and alternative definitions of value [Müller et al., 2019e].

This ambiguity of the value definition, i.e., requiring multiple viewpoints also including alternative aspects and the perspectives of the individual actors, leads to different challenges in managing such open ecosystems, requiring a holistic understanding and adapted processes [Müller et al., 2019d]. Therefore, regarding such open systems from different viewpoints of the ecosystem, instead of just taking an egocentric perspective, can be the key to understand how they work and survive in the long term.

3.3.4. Combining the Networked View of Business Models

The aforementioned partner perspective of business models already underpins the need to regard further actors in the value creation process. In reverse, a business also gets part of one's partners' business model, as partner itself. Consequently, several business models, potentially by a multitude of economic entities, are situated within one ecosystem. This results in an inherent connection and possible dependency between these firms and their business models, which must be considered by management to minimize risk and facilitate possible chances through cooperation. As illustrated in Figure 3.3, to pay respect to this fact, we see that this relation between business models happens in the means of a value network (referred to networked representation of an ecosystem) on the architectural level of a business, as defined by Osterwalder and Pigneur [Osterwalder and Pigneur, 2002] in their Business Logic Triangle. The value network represents the horizontal layer that puts the business models' value creation processes (including value creation and capturing), realized by the different partner businesses, into connection. Through horizontally linking the involved partners' business models, dependencies between them become more apparent and highlight their importance.

Whereas the theoretical foundation for and implications of this approach are still in their beginning, the practical relevance is already evident and showing promising outcomes. Instead of having one business model representation including all these partners in detail, e.g. in multi-sided business models represented with the BMC, we see multiple benefits of combining various tools that enable different viewpoints instead. In detail, we propose an holistic value view of the ecosystem through value networks, but also

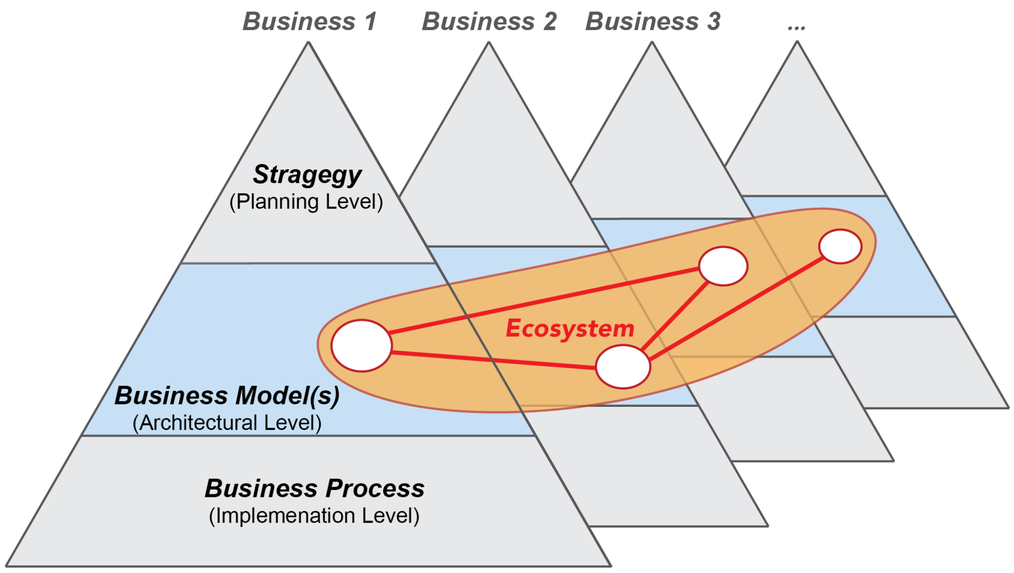


Figure 3.3.: Value Network horizontally linking the business model layer of Osterwalder and Pigneur's [Osterwalder and Pigneur, 2002] Business Logic Triangle

an egocentric representation of specific relations through more abstract frameworks. Since every relation in a value network represents an exchange of value [Gordijn et al., 2000], each relation potentially can be represented through an egocentric BM tool itself, further describing this value creation and capturing process. This links the holistic overall picture gained by value network frameworks to the detailed view of graphical BM frameworks that helps to strengthen this relation and ensure the biggest outcome for all involved actors. Combining those views, provides different viewpoints to practitioners and scholars that are easy to communicate and analyze, helping to understand innovative ecosystems.

This linkage is independent of the used notation of the value network and graphical business model, enabling its application in different existing settings in practice. To illustrate this concept and how it is applied, we introduce three different cases from the ICT-sector depending on ecosystems of different actors. We show how this approach can be applied with different notations and how it can be adapted to the different needs of the organizations. This is of special relevance for practitioners who are already using such tools and may extend it with just minimal effort.

Digital Advertising - A Platform Approach

Within their book "Business Model Generation", Osterwalder and Pigneur [Osterwalder and Pigneur, 2010] introduce multi-sided business models and how they can be modeled with the presented BMC. A simple and abstract example they also use is the one of advertising platforms, which became a picture-book case for platform based business models. On the one hand, a platform is providing a certain service or product to customers. The platform displays within its service/product advertisements from an external partner, the advertiser, and earns fees from this partner for the displayed advertisements. In an ideal case, the user follows this advertisement and becomes a customer of the advertiser itself. This results in an interconnected network between the three involved actors, who all are needed to realize the therewith connected business models of the service-provider and also the advertiser, since the ads are a channel to reach customers. As illustrated in Figure 3.4, the BMC (highlighted in grey) is supposed to represent this value creation process within one single canvas. Although the benefits of the BMC are manifold, already in this trivial example it shows weaknesses when it comes to illustrating dependencies and dynamics of value creation. Our approach instead, as highlighted in blue, includes the BMC for every relation in Allee's value network notation. Besides the clearer representation of the BMC for each relation from the platform provider, also the dependency on the external business model of the advertiser to the customer becomes obvious. This bigger picture can be used to improve the value creation process to ultimately enable every actor to capture the most value possible out of the ecosystem.

Whereas this simple example of multi-sided business models is well suited to explain the suggested approach, it is in reality already outdated, illustrating how quickly ecosystems and their dynamics evolve. Today, the full digital advertising example also includes an advertisement platform that connects service providers with advertisers through a bidding process and further provides the technical and financial framework to show the ads on the platforms and the afterwards billing. Furthermore, through personalized advertisements, also the advertisement platform is connected to the users, leading to a highly interconnected ecosystem of various dependencies, network effects, and values exchanged. Thus, this business model would

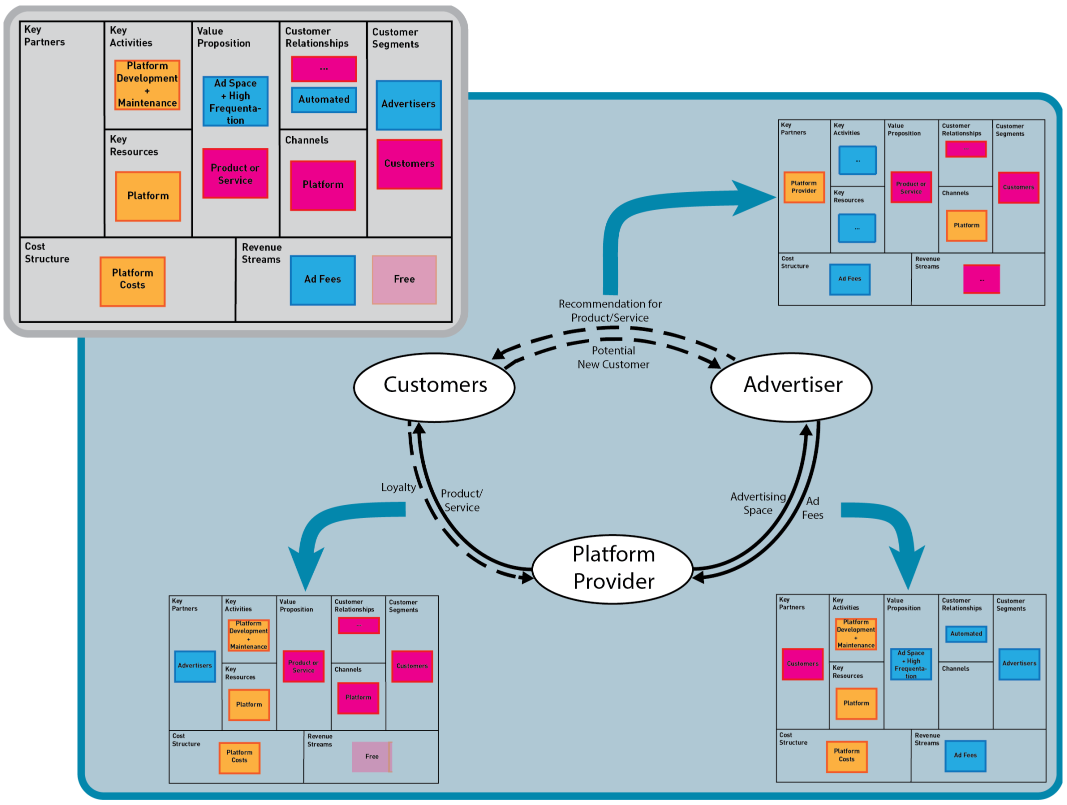
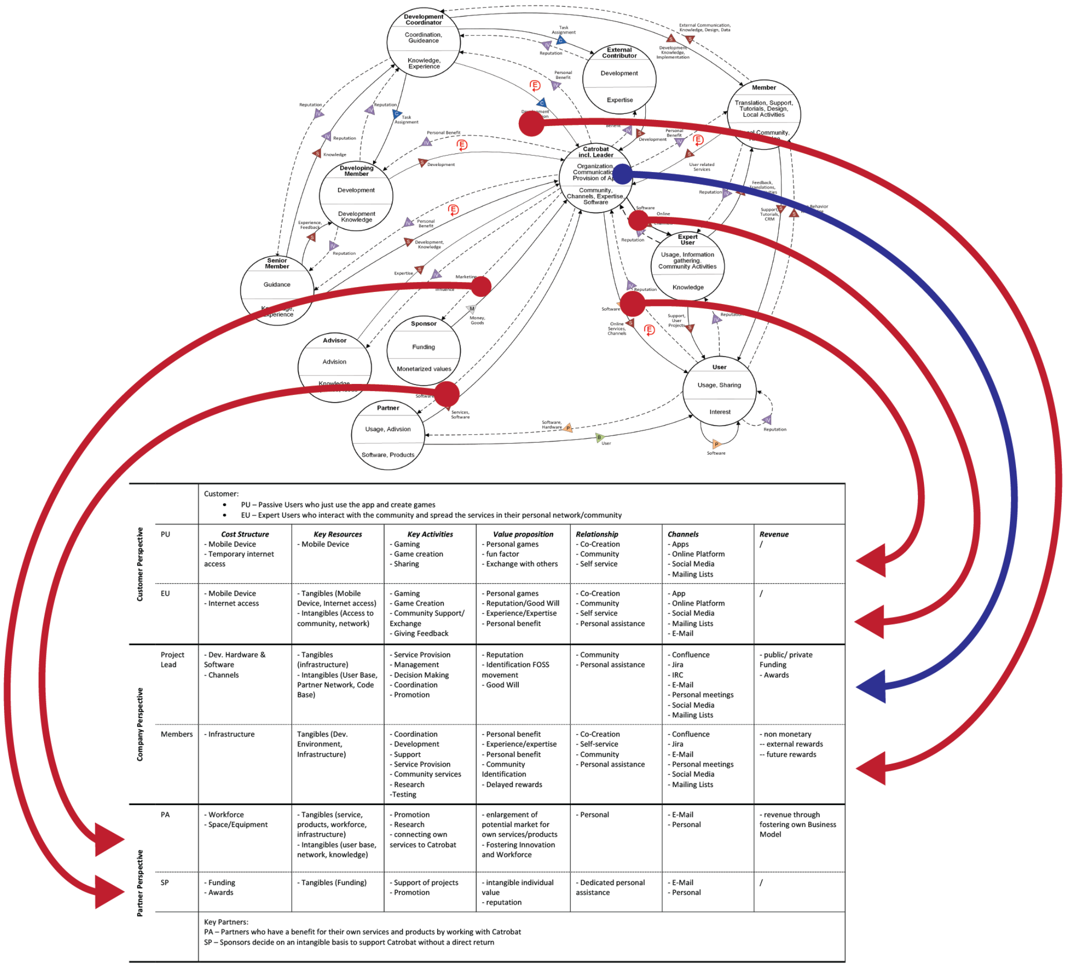


Figure 3.4.: BMC representation of the advertiser-platform business model (gray) lead over to the proposed ecosystem perspective of business models (blue)

require more than just two customer segments or partners, making it hard to be illustrated within one BMC, furthermore missing to consider the dependencies and network effects. The proposed value network approach is capable of doing so by still keeping the business model view available, as the following more sophisticated examples show.

Catrobat - The Case of Collaborative Communities

Over the last years, communities got an important source for external (open) innovation, especially in the area of open source software. External developers are collaboratively contributing to different projects that are often



Customer:
- PU – Passive Users who just use the app and create games
- EU – Expert Users who interact with the community and spread the services in their personal network/community

		Cost Structure	***Key Resources***	***Key Activities***	***Value proposition***	***Relationship***	***Channels***	***Revenue***
Customer Perspective	PU	- Mobile Device - Temporary internet access	- Mobile Device	- Gaming - Game creation - Sharing	- Personal games - fun factor - Exchange with others	- Co-Creation - Community - Self service	- Apps - Online Platform - Social Media - Mailing Lists	/
	EU	- Mobile Device - Internet access	- Tangibles (Mobile Device, Internet access) - Intangibles (Access to community, network)	- Gaming - Game Creation - Community Support/ Exchange - Giving Feedback	- Personal games - Reputation/Good Will - Experience/Expertise - Personal benefit	- Co-Creation - Community - Self service - Personal assistance	- App - Online Platform - Social Media - Mailing Lists - E-Mail	/
Company Perspective	Project Lead	- Dev. Hardware & Software - Channels	- Tangibles (infrastructure) - Intangibles (User Base, Partner Network, Code Base)	- Service Provision - Management - Decision Making - Coordination - Promotion	- Reputation - Identification FOSS movement - Good Will	- Community - Personal assistance	- Confluence - Jira - IRC - E-Mail - Personal meetings - Social Media - Mailing Lists	- public/ private Funding - Awards
	Members	- Infrastructure	Tangibles (Dev. Environment, Infrastructure)	- Coordination - Development - Support - Service Provision - Community services - Research -Testing	- Personal benefit - Experience/expertise - Personal benefit - Community Identification - Delayed rewards	- Co-Creation - Self-service - Community - Personal assistance	- Confluence - Jira - E-Mail - Personal meetings - Social Media - Mailing Lists	- non monetary -- external rewards -- future rewards
Partner Perspective	PA	- Workforce - Space/Equipment	- Tangibles (service, products, workforce, infrastructure) - Intangibles (user base, network, knowledge)	- Promotion - Research - connecting own services to Catrobat	- enlargement of potential market for own services/products - Fostering Innovation and Workforce	- Personal	- E-Mail - Personal	- revenue through fostering own Business Model
	SP	- Funding - Awards	- Tangibles (Funding)	- Support of projects - Promotion	- intangible individual value - reputation	- Dedicated personal assistance	- E-Mail - Personal	/

Key Partners:
PA – Partners who have a benefit for their own services and products by working with Catrobat
SP – Sponsors decide on an intangible basis to support Catrobat without a direct return

Figure 3.5.: Value Network of the open source Catrobat project (adapted from [Vorraber et al., 2019a], © 2019 IEEE. Reprinted, with permission, from Analyzing and Managing Complex Software Ecosystems: A Framework to Understand Value in Information Systems by W. Vorraber, M. Müller, S. Voessner and W. Slany in IEEE Software, May 2019) lead over to the Service Business Model Canvas (SBMC) representation (see [Zolnowski et al., 2014]).

directly connected or at least intensively used by firms. This results in complex ecosystems of open source organizations, different contributors, users, and ultimately other entities, such as firms, who are co-creating value and potentially realizing different business models. The free open source project Catrobat's ecosystem has already been analyzed through a value network analysis, as illustrated in Figure 3. Potential chances and challenges for managing this organization have been identified [Vorraber et al., 2019a]. Most management, especially in aligning the (primarily volunteer) actors into a shared direction is done by the project's leaders. This is in particular challenging since the ecosystem is rather diverse, including actors who contribute to the project for very different motivations. To gain a better understanding of the ecosystem and support future strategic decisions, a value network in V^2-notation has already been drafted in the past. The V^2-framework especially met the need of considering intangible values transferred and the motivation of the individual actors for their participation in the system. Based on this existing value network, all relations to the project's leading actor (highlighted in blue in Figure 3.5) have been mapped within the Service Business Model Canvas (SBMC)(see [Zolnowski et al., 2014]) to draw an egocentric picture out from a holistic viewpoint of the organization. The SBMC has reconfigured the elements of the BMC to especially meet the challenges that arise with service oriented business models [Zolnowski et al., 2014]. In Figure 3.5, we illustrate this process with Catrobat's value network. With this step, further examination on the relation from the project to the different individual actors can be performed, intended to further improve current management processes. Due to the project's unique setting and its demands for this analysis, a centralized view from the project, neglecting potential network effects emerging from the relations between the actors captured in the value network, has been in the focus of this representation. In this case, the existing V^2-value network helped to quickly identify the perspectives and segments of the SBMC and link their value propositions, since already the value network notation by Biem and Caswell [Biem and Caswell, 2008] (which got enhanced by the V^2 framework [Vorraber and Vössner, 2011]) outlines the involved actors and transferred values. Whereas the existing value network perspective in this case provided the holistic picture of the ecosystem and its possible network effects, the connected business model perspective gives more detailed insights into the individual key elements of the relations, i.e., revenues, costs, channels or relationships. The introduced approach of connecting value networks with business models also relates to the previously outlined [Zolnowski et al., 2014] shortcoming of the used SBMC when it comes to the proper representation of relationships and provides a solution to it.

MaaS - A Networked Service

Figure 3.6 provides an overview on the key actors and the value exchanges of a generic Mobility as a Service (MaaS) business model. The value network was generated based on the actor and business concept definitions provided by Kamargianni and Matyas [Kamargianni and Matyas, 2017] and includes only actors of the core business and selected actors of the extended business ecosystem. Typical examples of MaaS companies are Uber, Lyft or Grab, which provide a multi-sided technology and business platform to connect users and transportation service providers in a one-stop-shop manner. As indicated in Figure 4 each value exchange can also be zoomed in and represented as a detailed business model in form of a BMC. For the sake of clarity, this aspect is highlighted solely for the value exchange relation between 'MaaS provider' and 'Customers/Users'. However, in practice lining out each of these relations with a BMC becomes a key to understand complex ecosystems of different providers, as the example of MaaS illustrates.

3.3.5. Conclusion and Takeaways

This work reviews the connection between business models and ecosystems, represented as value networks, enabling innovation. In practice, as the outlined examples demonstrate, combining tools of different abstraction levels is beneficial when business models involve various partners or customer segments and therefore commonly used notations come to its limits. The presented cases give a general perception on how a holistic view of an innovative ecosystem including the business models of the connected partners can be drawn and communicated, but also how more abstract tools may be beneficial to just investigate and communicate specific value activities between two actors. By not introducing new tools or notations, the

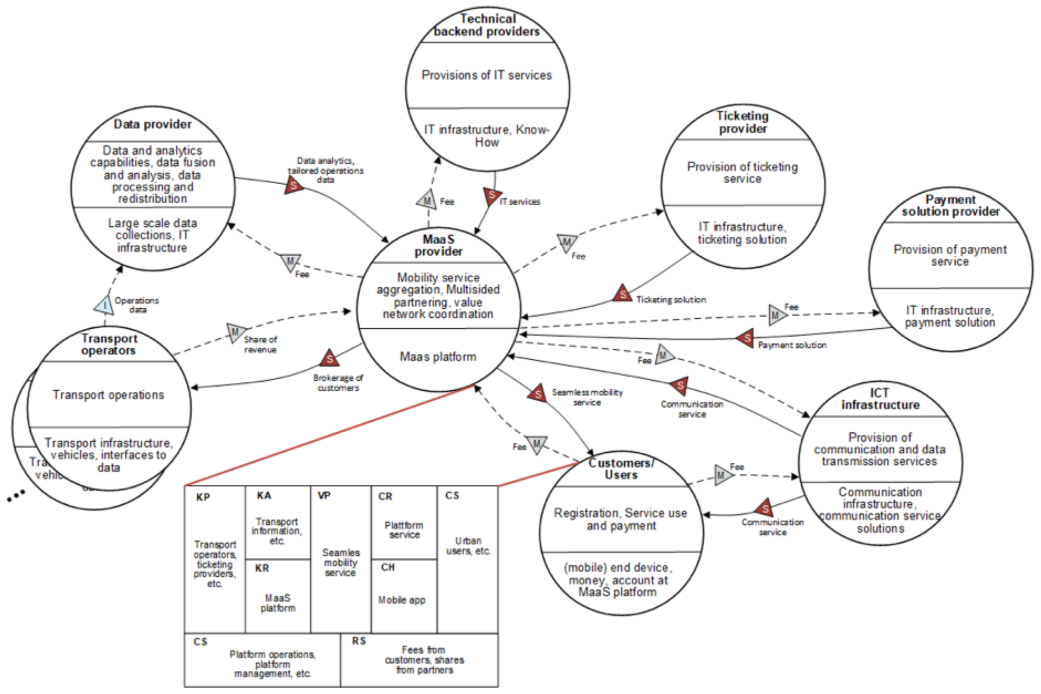

Technical backend providers
Provisions of IT services
IT infrastructure, Know-How
Data provider
Data and analytics capabilities, data fusion and analysis, data processing and redistribution
Large scale data collections, IT infrastructure
Ticketing provider
Provision of ticketing service
IT infrastructure, ticketing solution
Payment solution provider
Provision of payment service
IT infrastructure, payment solution
MaaS provider
Mobility service aggregation, Multisided partnering, value network coordination
Maas platform
Transport operators
Transport operations
Transport infrastructure, vehicles, interfaces to data
ICT infrastructure
Provision of communication and data transmission services
Communication infrastructure, communication service solutions
Customers/ Users
Registration, Service use and payment
(mobile) end device, money, account at MaaS platform
Data analytics, tailored operations data
Fee
IT services
Fee
Ticketing solution
Fee
Payment solution
Fee
Operations data
Share of revenue
Brokerage of customers
Seamless mobility service
Communication service
Fee
Fee
Communication service
KP
Transport operators, ticketing providers, etc.
KA
Transport information, etc.
KR
MaaS platform
VP
Seamles mobility service
CR
Plattform service
CH
Mobile app
CS
Urban users, etc.
CS
Platform operations, platform management, etc
RS
Fees from customers, shares from partners

Figure 3.6.: A Value Network of a Mobility as a Service business model based on Kamargianni and Matyas [Kamargianni and Matyas, 2017] illustrating that each value exchange relation can be zoomed in and represented as a single business model for example with a BMC [Osterwalder and Pigneur, 2010].

authors' approach can easily be established in organizations and supplements processes that are already commonly used. The takeaway for practice is that multi sided business models require a holistic value network view in addition to the commonly used egocentric perspective. The dynamics of ecosystems influence business models and their innovation. Therefore, building and understanding business models goes along with understanding the network, or ecosystem, it is realized in. Business models and in particular the Business Model Canvas are already often used to quickly scan, evaluate, and communicate business ideas from an egocentric viewpoint. Nevertheless, to succeed in the service industry where value is co-created often also in open communities with intangible values involved (i.e., in New Business Models or open innovation, a further holistic perspective is needed. If an idea seems reasonable to get realized from an egocentric point of view, this point of view needs to be transferred into the bigger picture including all partners and segments. Based on often already existing graphical business model representations of organizations, this holistic perspective can be reached by enhancing the business modeling process with the presented value network approach. Each relation within this value network represents a potential business model that can affect all other business models within the ecosystem. To innovate, compete, and gain new chances, understanding these business models can become an essential element when it comes to succeed with new or adapted existing business models.

Whereas this connection, its application and benefits for practitioners are highlighted in this paper, the further theoretical implications are still under evaluation and part of ongoing research. The concept of value networks is neither new, nor it is its connection to business models and innovation. Nevertheless, the proposed approach of combining tools of different abstraction levels brings a new viewpoint especially relevant for practitioners. Whereas Peppard and Rylander [Peppard and Rylander, 2006] already contributed to the theory and dynamics of value networks, they focused on its theoretical background with an emphasis on the strategical layer. New networked-notations, such as V^2, put this theory into practice on an architectural level and incorporate aspects such as additional relevant layers (e.g., social or legal aspects [Vorraber and Müller, 2019]), endogenous motivation and exogenous influences for each actor. Furthermore, they help to identify network effects and pay respect to new definitions of value, especially relevant for more sustainable businesses and ecosystems. Ultimately, combining these notations can in the future not only be used for drafting and communicating ecosystems, but also to simulate and thereby analyze promising ideas in more detail, leading to innovation. Further, the introduced approach also poses questions arising from its connection to business models. In the past, value networks or alliances have been described as component or resource of a business model (e.g. Chesbrough [Chesbrough, 2006a] or Johnson et al. [Johnson et al., 2008]). In this work, business models are used to describe relations within value networks, consequently seeing value networks as ecosystems in which business models get realized. The therewith arising questions are open for further research and are expected to contribute a new perspective to the still rising field of innovation in business models.

4. Quo Vadis Open Source?[1]

In 1999 Eric Raymond [Raymond, 1999] described open source projects as an open bazaar of different approaches and methods, as the opposite of a cathedral that is built by just a few dedicated individuals in isolation. Within the past 20 years since then, a variety of such open source projects established following the principles described by Raymond. By March 2019, Open Hub, an online directory for Free/Libre Open Source Software (FLOSS) projects by Black Duck Software Inc., listed more than 475,000 projects, of which 21.6% are active[2]. It can be assumed that the total number of existing open source projects is even much higher. This large number of projects and the rapid development of FLOSS also encouraged researchers to investigate this phenomenon from various viewpoints. Nevertheless, empirical research on this topic is in certain perspectives in an early stage and has a huge potential for further investigations [Crowston et al., 2008]. But, besides researchers also commercial entities got interested in the open source movement. The emergence of sponsored open source projects, i.e. projects that get established by such entities, already got mentioned early [West and O'Mahony, 2005], leading to the so called third generation of FLOSS projects that also follow commercial interests [Yamakami, 2011]. Consequently, open source communities are today in many ways connected to businesses and are complex ecosystems of different actors that are hard to manage and understand.

To analyze this evolution and to derive practical implications for projects that are situated in such ecosystems, the author wants to foster the understanding with empirical data of who is contributing to such communities and how

[1]This chapter is published as [Müller, 2019] (minor adaptions got made within this thesis to the original publication) ©2019 Copyright held by the owner/author(s). Publication rights licensed to ACM. DOI: 10.1145/3338906.3341461

[2]https://www.openhub.net/explore/projects accessed March 27th 2019

this evolves over time. Previous research indicates that most development in FLOSS projects is made by just a small number of highly motivated contributors [Krishnamurthy, 2005b, Ye and Kishida, 2003, Koch, 2004]. As this work is intended to show with a time based empirical analysis of 100 projects, this potentially influences a community's dynamics and how it may be managed to ensure a project's long-term development and success.

4.1. Background

The initial idea for this work came with an intended analysis of a local open source project (Catrobat) and its comparison with various other similar situated projects. In previous work from the author the case of the Catrobat project has been investigated from an ecosystem and business model point of view [Vorraber et al., 2019a, Müller et al., 2019e], from within the community at the project's unique environment [Müller et al., 2019a], and how such a specific project can be managed in an agile way [Müller, 2018, Müller et al., 2019b]. In a next proposed step, a comparison of Catrobat with other projects on common characteristics was planned to better understand the community and how it potentially can be lead into a shared direction that combines the goals of volunteer contributors and external objectives. However, this comparison brought up the need of a better understanding of open source communities and especially their dynamics over time as well as dependency on individual contributors, which directly influence the manageability of these projects and the risk for involved stakeholders.

Today, a large variety of FLOSS projects exists, each varying in several aspects such as size, domain, or license [Gacek and Arief, 2004]. The motivations for developers to contribute to such projects are manifold and have been subject of research in the past [Hars and Ou, 2002, Ye and Kishida, 2003, Fogel, 2005]. Developers change frequently, joining and leaving the community based on their motivation and needs, leading to different workforce being available for a project [Ye and Kishida, 2003, Koch, 2004, Terceiro et al., 2010]. This indicates that also the community is constantly changing, leading to an evolvement of such projects. Nevertheless, at specific time-points contributors can be characterized by different roles, e.g., as core, peripheral, or one-time developer [Ye and Kishida, 2003, Crowston et al., 2006, Terceiro et al., 2010, Lee and Carver, 2017]. Further, the degree and time of contribution varies between these groups for several reasons [Terceiro et al., 2010, Lee and Carver, 2017].

A conducted literature review backs the assumption that the majority of contributions to FLOSS projects is made by just a small number of people. However, much of this literature only regarded single cases (e.g., [Kagdi et al., 2008], [Ye and Kishida, 2003]), peripherally regarded this specific issue (e.g., [Koch, 2004], [Lee and Carver, 2017], [Weber, 2004]), or has been published several years ago and is questionable to be still today (e.g., [Krishnamurthy, 2005b]). Recent research on a larger sample of projects showed that the Pareto-Principle can not be applied for core-contributors, i.e. the distribution of the top 20% is higher than 80% [Yamashita et al., 2015]. Nevertheless, these studies neglect how and if these numbers are changing over time and its impact on the projects. This leaves room for further work, helping to better understand open source communities and their contributors, which particularly gets important when these communities must be managed or aligned into a common direction, e.g. in sponsored open source projects or projects with commercial goals [West and O'Mahony, 2005, Yamakami, 2011].

4.2. Aims

The author expects that FLOSS projects, their contributors, and co-creative approach evolved within the last years. Empirical data is intended to point out this assumed progression of such projects. Especially in the current commercially influenced generation of open source projects [Yamakami, 2011], knowledge about the involved communities of professional and volunteer contributors becomes essential to manage them, but also to reduce risks for the ecosystem's related entities (e.g., businesses) by ensuring their sustainable existence and healthy structure. Thus, based on the outcomes of an empirical analysis it is proposed to outline implications for practice. This might help to manage and guide such projects in a way that balances the needs and goals of the contributors (i.e., community) and connected stakeholders. This includes identifying key developers that are required to ensure a long-term development and on how to measure the current health of an open source community in general. This becomes in particular of interest when firms start using software by rather young or small communities.

As basis for this proposed work on open source communities in this regard, following research objectives get considered:

- Evaluating the dynamics of open source communities over time (a common yearly timespan, but also in relation to a project's individual lifespan, i.e., young vs. mature projects).
- Identifying key contributors, their impact and change.
- Finding measurements to characterize open source projects and communities (e.g., influence of the project size, similarities when firms engage in projects, etc.).

4.3. Methodology

An analysis of 100 different active FLOSS projects which use git as version control system and vary in the number of contributors has been performed for the years 2016 - 2018. The reason to limit the analysis to git repositories was that it is the currently primary used version control system by open source projects[3] and provides all necessary data needed for this work. The used sample includes FLOSS projects with between 36 and of 2,288 contributors during the projects' lifetime, representing a broad amount of different communities. Every project's main repository got checked out and its default branch analyzed with *gitstats* for each contributor. With the term "contribution" the changed lines of code (LoC), i.e., the sum of additions and deletions, is meant. The reason for this measurement is further lined out in Section 4.5. However, in contrast to previous works, this analysis regards a time aspect, i.e., calendar years, since it can be assumed that especially the first year of a project or years in which entities, such as businesses or universities, got engaged behave differently than "usual" years. This shall

[3]https://www.openhub.net/repositories/compare

help to line out the evolution of a project and further assess dependencies on developers.

4.4. Results

Before analyzing the sample of 100 repositories in detail the author expected to be able to confirm the assumption made in previous published work that just a few developers are responsible for the majority of contribution to FLOSS projects. The results from the taken sample indicate that the contribution of the top contributors is even higher than expected in the projects' lifetime, but also in individual years. As illustrated in Table 4.1, in 2018 the top 20% contributors are on average responsible for 93.98% of the LoC changed with a standard deviation of 0.088, underpinning that this high degree of contribution of the top 20% is true in the majority of projects. But, especially surprising is the high impact of the top most single contributor. The author expected the impact of this single developer to be high in small projects and to see a relation between this percentage and the number of contributors. But, as illustrated in Figure 4.1, there are certain projects with a comparatively high number of contributors, in that the most active contributor is still contributing more than 30% of LoC changed. But, with a standard derivation of 0.259, no general assumption can be made from this sample, outlining the need for further investigation. As stated by Weber [Weber, 2004], it has also to be kept in mind that this measurement represents just a quantitative aspect and cannot directly be connected to the importance and effort of contributors with smaller development contributions, leaving further room for combining these quantitative with additional qualitative methods and therewith also investigating the importance of peripheral or one-time contributors to a project's dynamic and survivability.

From a collaborative software engineering view, especially the highlighted importance of the single top contributor of the projects may lead to a heavy dependency and is potentially critical for the long-term existence of these projects. This gets further amplified through the slow change of the top contributor, i.e., 20 of the 100 projects had the same single top contributor in all three analyzed years from 2016 to 2018 and 41 of the projects in 2017 and 2018. The survivability of FLOSS describes that such projects shall be able to survive without being dependent on any individuals [Fogel, 2005]. This is in particular essential for the risk assessment of companies that are using open source software or contributing to it. As these results show, further investigation is needed regarding the survivability and dependency on these top contributors. In addition, the influence of companies in such projects leaves space for research. Already on from the early years of FLOSS there have been employed contributors to such projects, being paid for their development efforts by companies that generate profit out of it or by

	2016	2017	2018	2016-2018	lifetime
Top 20%	94.76%	94.85%	93.98%	96.78%	97.79%
Top 10%	87.25%	88.17%	87.03%	91.95%	94.08%
Top One	47.64%	48.35%	46.90%	43.32%	36.63%

Table 4.1.: The top contributors proportion on the total contribution in a timespan measured in LoC changed

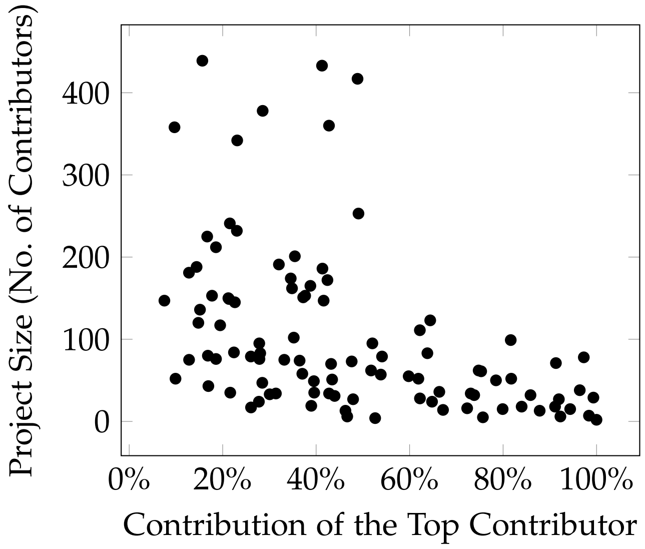


Figure 4.1.: The contribution (LoC changed) of the top contributor in relation to the total contribution in 2018

organizations that received funding to do so [Hars and Ou, 2002, Fogel, 2005]. Also in the small sample used for this work, there is a known number of projects that are actively maintained by companies but still open for everyone to contribute. This involvement of firms in open source projects may influence the results and needs to be evaluated. An interesting aspect for future research would be if there are significant differences between sponsored open source projects, i.e., projects that have been made open source by companies, existing community projects with active firms' contribution, and projects that are still primarily maintained by a purely volunteer community. Furthermore, expanding the analysis to inactive projects can help to identify patterns of failure in managing such communities that rely on just a small number of highly active contributors. However, in particular gaining this data from FLOSS projects about the involvement of external entities might be challenging, since these contributions are often informally and not documented.

4.5. Limitations

Whereas the results align with previous publications and provide further numbers to it, there are still issues that must be pointed out. The collaborative nature of FLOSS projects comes along with various challenges that need to be noted when it comes to such an empirical analysis. When taking a closer look on the results in this rather small data-set, in several projects' repositories potential problems have been spotted that influence the gained results.

4.5.1. Timeframe

This first analysis evaluated the communities solely by specific years and not in relation to their lifetime, i.e., if it is a young or mature project. In particular projects with a growing community are interesting to get examined and how they change in short time spans. In addition to that, only active projects got analyzed. The sample only includes projects that had commits in 2018. A comparison between active and inactive projects, also including projects with very few contributions, may be worth investigating. This might also provide insights into how these communities and their dynamics changed just before they went inactive, i.e., if certain specific contributors disappeared, forks emerged, or other events influenced the projects. This might help to evaluate the impact of the Truck Factor on such projects, potentially identifying it as common cause for the collapse of open source projects.

4.5.2. Measurement Methods

Already previous researchers highlighted the differences between various measurement methods for the contribution [Yamashita et al., 2015]. The measurement of LoC in this work is limited since it also counts copying or reformatting code and including additional resources. Nevertheless, it had been preferred over the usage of commits as measurement methodology. Especially the introduced time perspective limits the benefits of commits since certain projects just have a small amount of commits with large changes within the considered timespan. Commits in general not necessarily represent the contribution, i.e., a commit for a feature may be much more involving than a bug fix consisting of a single line of code. In addition to that, squashing commits falsify the results since previous data on the contribution gets lost. In this respect it would be interesting to add additional factors of analysis such as contribution-frequency or code quality [Behnamghader et al., 2017]. This is also beneficial to analyze the dependency on contributors for a project's further development, i.e., to not only measure contribution by LoC or commits, but combined with a qualitative statement on all contributions to the project, e.g., also activities on mailing lists or community support, which would need to get defined and quantified for comparison.

4.5.3. Contributor Mapping

The problems with contributor identification are threefold. First, many users are using the same username but different email addresses. Second, a smaller number of contributors is using different usernames with the same email address. Third, users got identified with very similar but different usernames, which could be considered to be the same person. Another issue is the usage of generic identifiers making it difficult to find out if one individual contributor is using it or several are working together. Therefore, the analysis has taken the email address of a commit as key for this sample, which from the author's point of view is sufficient in regard to the set aims. However, to gain an exact picture, more sophisticated user mapping measures would be needed to get defined, paying respect to different keys that might also change over time (i.e., one contributor using different usernames or e-mailadresses during the years active).

4.5.4. Comparison on Common Characteristics

In this study, the projects have not been further differentiated regarding characteristics such as age, size, domain, or business involvement. Based on this information, projects might get clustered or predictions be made when a project changes, e.g. when it grows or businesses get involved. Depending on the intended insight, such characteristics might help to better understand the communities behind FLOSS projects and how they behave in certain situations.

4.6. Conclusion

These results from a rather small sample confirm the assumption that most code in open source projects is created by just a small number of active contributors. Raymond states that a difference between open source and traditional software development is that traditional software projects consist of just a small number of developers having all the knowledge, whereas FLOSS projects are open for a larger number of contributors [Raymond, 1999]. Although open source projects are open for contributions and follow the open bazaar principles, this work outlines that also they heavily rely on a small group of active developers creating the majority of the code. In certain cases the majority of work is even done by only one contributor.

Potentially, this can be crucial for the survivability of a FLOSS project. Although projects accept contributions from the public, these active minorities can be seen as wizards crafting large parts of the code and having the overall knowledge. Therefore, in some circumstances, they might be building an open cathedral instead of an open bazaar as defined by Raymond. This also impacts the growing interaction of companies with FLOSS projects. Further investigation is needed in the proposed following research work, lining out the therewith connected risks for businesses. Further, it can be assumed that this aspect influences how FLOSS projects shall get managed today. These observations suggest therewith connected interesting questions, for instance how dependent these projects are on the top contributors to exist in the long term, how sustained development into a certain direction can be ensured, and how diverse communities can be led to a shared goal. Analyzing the evolution of FLOSS projects and their dynamics can be a key factor to efficiently manage open source projects in the future and align the goals of businesses and (employed and volunteer) contributors.

5. The Case of Catrobat

5.1. Engaging Students in Open Source: Establishing FOSS Development at a University[1]

The digital transformation has changed education tremendously in the last years. E-learning and open education are nowadays commonly used at schools and universities, providing new possibilities and chances for educators and learners [Lakhan and Jhunjhunwala, 2008]. Open source software, such as Moodle or Wikimedia, are today accepted services and used in classrooms all over the world. Since the beginning of open source, universities, e.g., the MIT or Berkeley, have played an important role in shaping the idea of sharing code [Lakhan and Jhunjhunwala, 2008, Lerner and Tirole, 2002]. Open source provides various opportunities for universities and especially its students [Ye and Kishida, 2003]. Awareness of open source software, especially learning management systems, e.g., Moodle, and adapting it is on its rise in higher educational institutions [van Rooij, 2011]. Although open source software use and development seems to be mainstream at universities all over the world, there seems not to be much literature about how to foster its development in the curricula and research of universities. There is a small number of practical examples and considerations for doing so, pointing out potential gains but also challenges [Dionisio et al., 2007, Ellis et al., 2007, Casson and Hawthorn, 2011]. Participating in open source software development is beneficial for students since they gain access to expert code, development tools, and are connected with

[1]This section is published as [Müller et al., 2019a] CC-BY-NC-ND. DOI: 10.24251/HICSS.2019.930

a community of skilled programmers [Ye and Kishida, 2003, Ellis et al., 2007, Seely Brown and Adler, 2008, Pinto et al., 2017, Beecham et al., 2017]. Moreover, companies nowadays are widely using open source solutions and gain benefits in hiring students who already have experience using this software during their studies [Lerner and Tirole, 2002, DeKoenigsberg, 2008]. Thus, fostering contributions of students to open source software at a university can have a positive impact on their learning success and later career [Casson and Hawthorn, 2011]. Bringing open source to classes also has benefits for educators, since it is a cost-effective, scalable, and practical approach [Beecham et al., 2017]. Nevertheless, it is also challenging in several aspects, such as community bonding or grading [Ellis et al., 2007, Pinto et al., 2017, DeKoenigsberg, 2008]. We present how open source development can be introduced at universities and how students can learn from it. We use the open source project Catrobat as a case study in our work. This project has its roots at Graz University of Technology (Austria), and the main proponents of the project are either employees or students of it. We introduce how this project is organized and how it is used for university courses. The results of a survey give insights into the background of the participating students.

For this chapter we consider the following questions:

- *RQ1:* How can open source projects be organized at universities?
- *RQ2:* How to respond to challenges in engaging students in open source during their studies?
- *RQ3:* How do students experience contributing to open source software projects during their studies?

The aim of our work is to provide a positive example how open source development can be brought to and driven forward by universities. We want to encourage more educational institutions to motivate students to be engaged in open source software development and foster an open and innovative mindset. We also highlight challenges that need to be considered.

Several definitions of what is meant by open source can be found in literature, especially connected to the movements of Free (Libre) Software (FS), founded by Richard Stallman [Stallman et al., 2002], and Open Source

Software (OSS), described by Eric Raymond [Raymond, 1999]. The studied Catrobat project is a classic free/libre software project that uses the GNU Affero General Public License version 3 and the Creative Commons Attribution-ShareAlike 4.0 International Public License. Since most of our findings apply to both FS and OSS projects, and since there is a lot of overlap, we use the term FOSS (Free and Open Source Software) in the following, with the understanding that the findings will also apply to some situations that do not strictly involve free/libre projects.

5.1.1. Used methodology and structure

Since our work represents a personal experience, we use qualitative methods to describe a case study, supported by the quantitative results of a survey. The focus of our work is "how" and "why" active involvement in FOSS projects can be beneficial for universities. Thus, the methodology of case studies is well suited to answer our research questions [Yin, 2009, Baxter and Jack, 2008]. We use a single case study for our work to analyze the benefits and challenges described in existing literature, and to present the unique project-setting [Yin, 2009]. To create a more holistic understanding of the presented case study and underpin the results, we also discuss quantitative data from two surveys [Baxter and Jack, 2008]. This mixed method of surveys within a case study helps us to enrich the evidence and gives further insights into the proposed research questions [Yin, 2009].

First, we present the case study of Catrobat: how it was established and how it is organized as an international FOSS organization. To answer RQ1 in detail, we describe the project's setting that has been established in 2010 and been adapted since then to the needs of contributing students and educators. On the one hand, certain of these settings are provided from the university and/or externally from the organization, on the other hand there are several aspects that resulted from the nature of the community.

Second, we describe the insights from an educator's point of view about bringing open source development to universities. To answer RQ2, we point out in detail how students can participate, how certain challenges are handled, e.g., grading or community bonding, and how research projects in the field of open source are conducted.

Third, we answer RQ3. In spring 2018 we conducted two anonymous online surveys for active and former students that have contributed to Catrobat during their studies at Graz University of Technology. 58 of 103 current students we asked have answered the survey (56% response rate) and provided detailed insights into their background, their motivation, and their contribution to the project. We also asked 98 former students to take part in a smaller survey, giving insights into how they see their past contribution to the project and whether it helped their career. 31 of these alumni provided feedback, resulting in a response rate of 32%. Although we are aware that these response rates may lead to a non-response bias, we can identify certain arguments that can be used to answer the presented research questions. We provide a holistic perspective on this unique setting and how it has been driven forward in an innovative way, paying respect to the students, educators, stakeholders, and also the software's users. Last, we discuss the obtained results and their implications, leaving room for further research.

5.1.2. The Catrobat project

In 2010 project founder Wolfgang Slany, professor at the Institute of Software Technology at Graz University of Technology, came up with the idea of a mobile programming framework for smartphones similar to the well known Scratch framework developed at the MIT Media Lab. Since no mobile solution existed, he kicked off the Catrobat project, aiming to develop an easy-to-use mobile app allowing to create programs with simple visual bricks. The project's vision is to provide tools that foster computational thinking skills among teenagers, independently of traditional PCs and in an environment that they are nowadays used to: mobile devices. Mobile devices have become part of our everyday life, are widely used by teenagers all over the world, and provide a cheap alternative for computer science classes that often still rely on traditional PCs. Catrobat aims at enabling young smartphone users to express themselves creatively throughout their digital mobile life instead of remaining mere consumers of the underlying technology. Starting with a few interested students, first ideas were implemented, and over the years it gradually became an international project. Several hundred people from all over the world already contributed to the project and delivered code, translations, educational resources, or other support under FOSS licenses, helping to realize the project's vision. In 2014 the first version of Catrobat's free coding app Pocket Code[2] was released on Google Play [Slany, 2014a], attracting more than 500,000 users as of June 2018. An additional drawing app, Pocket Paint, got released at the same time, allowing users to create and design their own graphics for their games and apps. Several extensions for various hardware (e.g., Arduino boards or Lego Mindstorms robots) were added to Pocket Code over the time and further features implemented to provide more possibilities relevant for the young target group. Whereas this Android app is already available to the public, a version for iOS is currently in an alpha testing phase, and a beta version of an HTML5 player for desktop and mobile browsers is available on Catrobat's sharing website[3]. On this sharing site, users can publish their projects created with Pocket Code. Besides the development of the described services, further value has been added by contributors through creating different educational resources, maintaining the community of users, or translating the services in more than 50 languages. Although the project nowadays is based on an international community of contributors, the majority of its developers is connected to Graz University of Technology. The main reason for this is that students are actively recruited, motivated, and supported to contribute to the project during their studies.

5.1.3. Project structure

As mentioned, the project got kicked off by a professor and a couple of interested computer science students working on the main Android app (called Catroid in the project's beginning) and its sharing site. Over the years, a community of international contributors was established to bring the project forward. Although many contributors are connected to the university, "Catrobat" itself exists independently as a FOSS project. This ensures the

[2]https://catrob.at/pc
[3]https://share.catrob.at

independence of the project and also allows external contributors to easily contribute to the project. These external contributors led to a distributed network of project members who are used to work together from all over the world as well as with new developers, which supports teaching global software engineering in an educational context [Beecham et al., 2017]. The closeness of the contributing community to the university is beneficial, since it makes it easier for students to get into it, something that has been identified as a potential challenge for bringing FOSS development to universities [Pinto et al., 2017]. Open source projects and communities usually don't rely on a strict hierarchical structure, yet contributors are organized by their roles and influence within the community [Ye and Kishida, 2003, Nakakoji et al., 2002]. This is also the case for Catrobat. Contributors are characterized by their role and team that focuses on a certain aspect of the project. Although there is no strict hierarchical structure, open source projects usually have a smaller, shifting leadership group that, while not being able to give strict instructions like in companies, instead can give recommendations and an overall direction to the volunteer contributors [Lerner and Tirole, 2002, Nakakoji et al., 2002, Fogel, 2005]. At Catrobat, the overall direction of the project is provided by a committee that is also motivating the community to contribute.

5.1.4. Project roles and teams

Open source communities are composed of a set of individuals, each having a unique background and personality. Not only developers are part of these communities, but also other types of contributors and users, who collaboratively drive such projects forward [Nakakoji et al., 2002]. Since the source code is freely available anyone can contribute to the code base [Raymond, 1999, Ye and Kishida, 2003, Fogel, 2005]. Each contributor can take over one or several roles according to personal interests [Nakakoji et al., 2002]. These roles are not assigned and can change, depending on the contributor's commitment to the project [Nakakoji et al., 2002, Torres et al., 2011]. Thus, the structure of the organization of a FOSS project varies according to the project's nature and its members [Ye and Kishida, 2003]. Also governing these communities of contributors usually depends on the project's needs and its structure. Although the structures in general tend to be flat, some leadership (providing a vision, giving recommendations, or keeping the community together) based on trust from the community is needed [Lerner and Tirole, 2002]. This trust is essential for contributors and the overall project, since mistrust may lead the entire project to collapse [Ehls, 2017]. The general development and management approach of Catrobat is based on agile principles, e.g., eXtreme Programming [Harzl, 2017]. Following these principles also fosters teamwork and guides the community towards a shared goal. It is important to mention that management as well as other responsibilities are usually shared among contributors, reducing complexity and dependencies on particular persons [Fogel, 2005]. Although several common roles and structures of open source projects have been identified [Ye and Kishida, 2003, Torres et al., 2011], there are differences between the individual projects. In the Catrobat project, we differentiate between the following roles:

- Users: There are various types of users that are part of the community. However, they are mostly not contributing, or just providing value to other users. Thus, we will not discuss their role in this work.
- *Peripheral Contributor*: Contributors that do not interact with the community and contribute in a narrow or irregular way (e.g., one bug fix commit or a few translations).
- *Active Contributor*: Contributors that are regularly contributing (e.g., code or resources) to the project and are an active part of the community.
- *Senior Contributor*: Experienced contributors that advise newer contributors, take the responsibility to review pull-requests, and accept code.
- *Coordinator*: Coordinating a certain part of the project (team) and guiding the involved contributors. Communication and coordination are the main tasks.
- *Product Owners*: Committee of highly involved contributors (including the project's founder) providing the overall vision and direction of the project.

Students that contribute are treated independently of their status and are therefore not treated differently than other members of the project's community. This aligns with other open source communities, where the role is independent of any attributes (e.g., age) and is earned through contribution [Nakakoji et al., 2002]. Contributors take one or more of these roles during their contribution, depending on their own preferences and situation within the project.

Besides roles, contributors can also be characterized by their team within the project. To keep such projects manageable and successful it is needed to divide the contributors in teams that can work on well defined tasks and almost independently from other teams [Lerner and Tirole, 2002]. As outlined, the Catrobat project with its many services, features, and other aspects such as education or design, provides many different possibilities for contributions. Contributors typically work in small teams that have a fixed scope and that can work more or less on their own. As illustrated in Figure 5.1, these teams have a special focus. This enables contributors, including students, to work in a domain they are personally interested in. As mentioned, these teams are guided by a "Coordinator", an experienced and highly engaged contributor. From an educational point of view, this structure gives students the choice to work on a field they like and to develop various skills depending on their interests.

Communication

A major aspect for open source communities is internal communication. Failing to interact with other members can upset contributors and slow down the whole project by hindering collaboration [Ehls, 2017]. Providing communication, documentation, and guidelines for developers and also users is important for FOSS projects right from their start [Fogel, 2005]. In the context of Catrobat, services and processes needed to be introduced to allow communication between distributed team-members and also ensure the flow of project relevant information [Fellhofer et al., 2015]. Catrobat provides a variety of services to tackle these potential issues. As an example, the project itself offers public instances of Jira and Confluence to track the development process and document the overall project structure. While the project in early times used IRC (Internet Relay Chat) for communication [Fellhofer et al., 2015], Slack channels are now the main place of

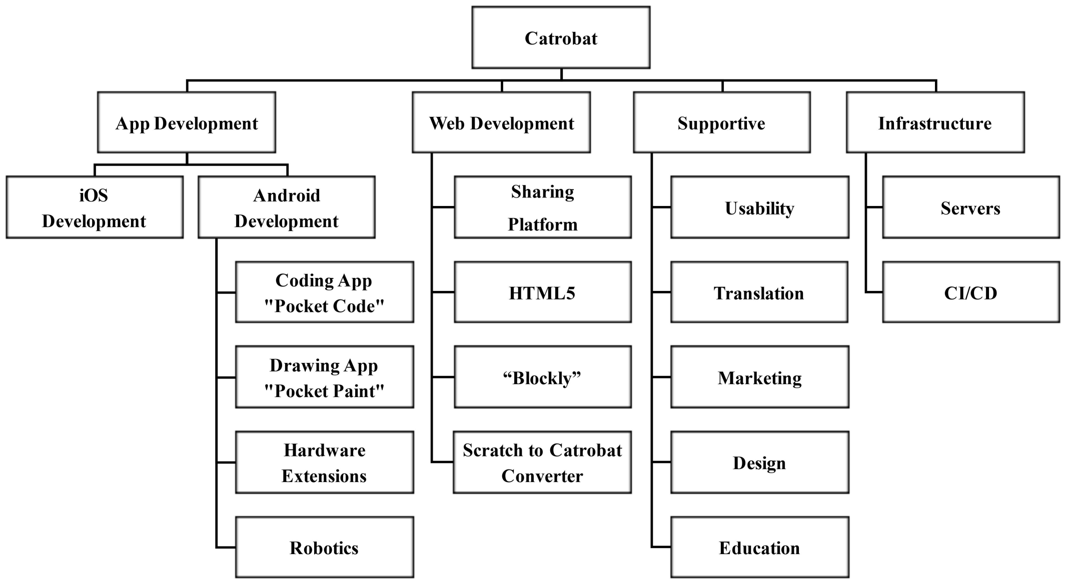


Figure 5.1.: Team structure of Catrobat as of June 2018 (Figure as published in [Müller et al., 2019a] CC-BY-NC-ND)

communication between contributors. These tools are commonly used in industry, fostering the technical skills of the contributing students [Pinto et al., 2017]. Local students are encouraged to use these tools but also meet with other contributing students in person. Although working remotely with external contributors is common, there are also regular meetings between students and the university staff. These meetings aim to enhance communication and also to identify potential problems early. Furthermore, students may use a dedicated room at the institute to meet and work on the project. This room, which has space for approximately 20 students, is the primary work space for more than a quarter (28%) of the currently participating students (over 100).

Infrastructure and setting

For a FOSS project, various infrastructure has to be maintained to keep the project running, e.g., hard- and software for development as well as internal and end user related services. Projects have to provide various tools, e.g., for communication, information management, or version control [Fogel, 2005]. We give an overview over the infrastructure maintained for the Catrobat project and briefly describe it. Some of this infrastructure is provided by the university (especially testing infrastructure), others (e.g., general services and software) by the project itself. We distinguish between hardware, software, and provided services to keep the project running.

Hardware
Dedicated Servers: We are running 12 dedicated machines ranging from desktop-class (32GB Ram, quad-core) to mid-range server-class (256GB Ram, 14 cores/28 threads) used mainly to host virtual machines, except an Apple XCode server and Jenkins instances, to flexibly provide services.
Mobile Devices: We provide over 120 mobile devices (phones and tablets) for the testing and development of the project's iOS and Android applications.
Equipment: A dedicated project room at the Institute of Software Technology, open 24/7 for the students, is equipped with 20 LCD screens for the developers and testers, 4 white boards, a color laser printer, and a 50 inch flat screen TV for meetings, presentations, and Jenkins monitoring. Two mobile beamers for presentations and meetings are also available, as well as a coffee machine and air conditioning (both not standard for student projects).
Miscellaneous: This category consists of single-board computers, such as Raspberry Pi and Arduino, electronic and robotic assembly kits, tinkering material, micro/little bit kits, various robots, drones, as well as robot arms for hardware testing on the CI server.

Software
Jira: Atlassian Jira for issue tracking and management of the developers' backlog.
Confluence: Atlassian Confluence is used as a knowledge base for every sub team in the project.
Slack: The daily short term communication is facilitated by various slack channels for every sub team in the project. Most channels are open and can be subscribed if needed.
GitHub: The project's source code is hosted on GitHub.
Crowdin: All project's translations are handled with Crowdin, an online localization management platform.

Yourls: A short link service to resolve short URLs.

Services for internal use
Jenkins: Jenkins is used as a continuous integration platform. The complete tests are run on a regular basis and when GitHub pull requests are issued.
Backups: Additionally to the individual local backups of the different systems, a centralized backup solution with redundant storage is maintained.
LDAP: An LDAP instance is used to log-on to the different services (JIRA, Confluence, Share, Jenkins,...).
Workspace maintenance: The project's open workspace is equipped for convenience reasons with a coffee machine, a fridge, air condition, and office material. Coffee and snacks are refilled regularly.

Services for end users
Catrobat Share: The central point of service for the end user is the project's sharing website https://share.catrob.at/. On this site users can download projects and, if registered, they are able to upload projects.
APK generator: To foster sharing, projects can be downloaded as Android Application Packages (APKs). They are install- and executable on any compatible Android device without the need of the installed Pocket Code app.
Recommender System: The sharing site is backed by a recommender system to suggests similar projects.
Converter: A Scratch-to-Catrobat converter allows to run Scratch programs with Pocket Code.

5.1.5. Students' involvement

Students, as other contributors, participate at Catrobat on a volunteer basis only, and clearly understand the rights and limitations entailed by the project's FOSS nature. There is no compulsion for students to contribute, as there are ample alternatives for them to choose from among many other projects and it is not required for their studies. Nevertheless, we provide an attractive environment to get engaged in Catrobat as part of their curriculum if they want to. Various possibilities are available for students to participate in Catrobat, depending on their degree program, the field they are interested in, and current research projects connected to the project.

5.1.6. Organization at the university level

Besides the described general structure of Catrobat, there is also a university related framework behind the project. Wolfgang Slany and his team of currently three assistants manage the students and take care of them. This team has several research backgrounds, covering fields such as software development, economics, usability, or sociology. This also allows to supervise the students while they are working in different teams of the project, focusing on a variety of domains. Cooperations with other academic institutes and companies provide access to further expertise in specific fields. In the setting of open source projects, instructors act more as a guide, fostering the students' understanding built from experience [Ellis et al., 2007]. Thus, the involved staff offers regular consulting hours, joins the local meetings of students, and actively participates on the project's communication channels. Furthermore, individual meetings with the currently over 100 actively involved students happen on a regular basis, also helping to ensure a positive learning outcome for them. This approach is time-consuming but apparently beneficial, since students get used to work individually in teams, learn from each other, but are still supervised in a way that ensures a relevant outcome.

5.1.7. Participation model

The project is situated at Graz University of Technology, which currently offers four degree programs in the domain of computer science. Although the scope of these studies varies ("Computer Science", "Software Development and Business Management", "Information and Computer Engineering", and "Teaching Subject Computer Science"), the presented structure enables students of all these fields to contribute to the project within their curriculum. Professor Slany and his team of assistants are offering several practical courses (e.g., "Mobile Applications" and "Software Technology") that are directly part of these curricula. Students have the choice whether they want to work on a traditional task offered in these courses, or whether they want to work on tasks that are directly related to the “Catrobat” project, but still following the scope of the course. The majority of students still take the “classical” courses, with no relation to Catrobat. However, several dozen students every year (e.g., over 40 in the first five months of 2018) choose this participation model. Although the majority of students just stays with the project for a short period of time (usually less than a year), there are students that get committed to the project for a longer period. As shown in Figure 5.2, by spring 2018 there are students in the project who started 6 years ago with their first contribution and are still an active contributor to it. The introduced setting allows students to stay longer than just a single course, getting committed to the project and gain deep knowledge in different fields related to their studies. But there are also students and even university alumni who voluntarily stay longer with the project for various reasons and do not earn any further credits.

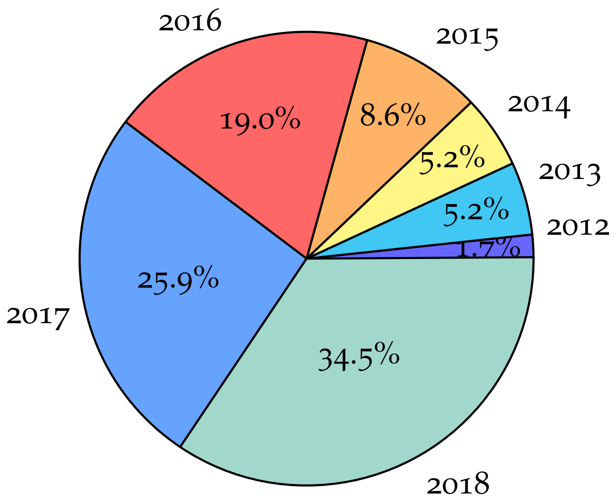


Figure 5.2.: Year active students started contributing to Catrobat as of June 2018 (Figure as published in [Müller et al., 2019a] CC-BY-NC-ND)

Besides the course-related contribution, students also have the opportunity to write their Bachelor's, Master's, or PhD thesis on a topic related to their contribution to Catrobat. This enables students to stay with the project for a longer time than just a certain course, as described above. The scope of the project provides many domains, e.g., software quality, usability, computer education, or project management, from which students can choose more or less freely for their thesis, according to their interests, the relevance to their studies, the level required by the thesis, and the requirements of the project. This allows them to write about a topic they are personally interested in. Furthermore, their practical work gets directly applied to a worthwhile, relevant, and publicly available project, impacting a real user-base that provides fast feedback and data for their work. Till today, over 190 bachelor's projects, more than 30 master's theses, and two PhD theses are related to Catrobat. In addition to that, university staff uses the project for their research and in courses, e.g., to teach coding concepts to beginners. Several grants and fundings also allowed to employ student contributors as university staff either part time during their studies or full time after they have graduated. As an example, these employees developed specific project relevant features, e.g. the Right-To-Left language version of Pocket Code, but also support students in their work. Employing students for FOSS development also worked for other universities, such as at the Oregon State University Open Source Lab [Casson and Hawthorn, 2011].

This approach allows students and researchers to focus on open source by making it their primary work. By doing so, opportunity costs are minimized, meaning that there is no disadvantage for the participating contributors (e.g., students could get slowed down in their studies if they work on open source unrelated to their curriculum, or an academic's output could be affected negatively if he works on open source besides his or her main research topic) [Lerner and Tirole, 2002]. This approach also ensures the sustainable development of the project, since students' contribution is not limited to a certain course. Time constraints can be challenging since they delay the contribution process and make the engagement of students in FOSS projects difficult [Pinto et al., 2017]. Especially the writing of a thesis or doing project work is not necessarily restricted to the official semesters and can also happen during summer or winter breaks. This gives the students further freedoms and independence, as long as they are discussed with the university's assistants in advance.

As outlined by previous research [Ellis et al., 2007, DeKoenigsberg, 2008], grading might become a problem in such settings. In general, theses and project works are graded individually, also independently of this approach. Thus, there is no additional effort needed compared to the traditional setting. Students taking the described courses are graded in the same individual way. All students have an initial meeting with the professor and one of the assistants, setting the conditions for their work, also outlining the expected outcome in regard of the learning success. This expected outcome and work is defined with respect to the course they take, the credits they will earn for it, their field of study, as well as their personal interest in a topic. The expected contribution for the defined outcome is hard to predict since the therefore needed work is a group effort and varies by a multitude of factors, e.g. previous experience of the student. Thus, all contributions rewarded by university credits are time boxed by the amount of time expected to be spent depending on the course credits for the student. This is similar to the way most regular employees are rewarded for their work, and allows students to plan their contribution in a tractable way, as well as it ensures that they do not get overloaded with work, e.g., by a too large defined expected outcome. It also forces us to split up the work in small parts, which works well with agile software development methods, and fosters cooperation between contributors. Still, although the contribution is time boxed, the outcome itself, in terms of quality and scope, must relate to the requirements of the taken course. After having their work completed, the students' work gets evaluated by the staff. Therefore, also the feedback of the community is used, since all pull requests to Catrobat get reviewed by experienced contributors before they get merged. Pure peer-grading would not be sufficient, since contributors might not be critical enough [Ellis et al., 2007] or too critical. All progress of the student is tracked through the project's coding tools (i.e., GitHub, Jira, and Confluence), helping the staff to get an overview of the completed work. Although the effort of grading is still higher compared to traditional grading, a good project's infrastructure setting and communication eases the process for educators. Also students benefit from this individual feedback since their strengths and weaknesses can be discussed in a final meeting, showing up potential room for improvement.

5.1.8. Motivation for students to contribute

Although this open source project was initiated at a university, and students may earn credits for their contribution, choosing to participate happens on a voluntary basis. This voluntary basis is not only legally required because of the special relationship between students and their university, as they are basically a kind of "customer" of their university, but also psychologically essential for the success of the project, since it is directly connected to the motivation of the contributors [Ye and Kishida, 2003]. The students can take this either as elective courses, project work, or as part of their thesis. Thus, the role of motivation also needs to be considered to attract motivated students and run such a project in a sustainable and successful way. Not just for contributors in general there is a diverse number of motivations to contribute, but also for students in particular [DeKoenigsberg, 2008]. Research by Ellis, Morelli and Hislop [Ellis et al., 2007] already pointed out several motivational aspects for students (e.g., working on real world projects) that we want to analyze with our survey.

5.1.9. Motivation to contribute to open source

Previous research pointed out different reasons why people contribute to open source projects (e.g., Hars & Ou [Hars and Ou, 2001], Ye & Kishida [Ye and Kishida, 2003], or van Krogh et al. [Von Krogh et al., 2012]). Contributing to open source comes along with a variety of benefits and costs for the contributors [Lerner and Tirole, 2002]. A main aspect is that contributors aim to receive a net-benefit, meaning that the benefits of contribution, or innovation, exceed their costs [Lerner and Tirole, 2002, Von Hippel, 2001]. In this respect, the net benefit is defined individually for each contributor. Failing to recognize the individual goals of community members is a potential source of failure for such projects and can hinder their success [Ehls, 2017]. A variety of motivators, such as reputation, one's own use or career, got identified in the literature and got pointed out as overall drivers that get people into open source [Von Krogh et al., 2012]. Motivation of contributors is a core research topic in open source and has attracted a large number of researchers [Von Krogh and Von Hippel, 2006]. Nevertheless, many aspects and questions related to the motivation of volunteer contributors in open source are still not answered properly [Von Krogh et al., 2012].

It is important to note that there are also paid contributors to open source projects, whose motivation might differ. Having long-term contributors, e.g., through hiring them, allows sustainability, since swapping programmers or introducing new ones slows down the development [Fogel, 2005]. Whole businesses emerged around open source and enable contributors to profit from their contributions [Chesbrough, 2006a]. Furthermore, corporations started actively to involve themselves in the development of open source software, since it provides manifold benefits [Fogel, 2005, Chesbrough, 2006a]. Also Catrobat benefits from university research projects in that developers can get employed for a certain timespan. Also grants for contributing to open source are available for students. As an example, Catrobat has been selected as a mentoring organization for Google Summer of Code for several years now, allowing the project to fund students from all over the world to work on our project during their summer break.

Surveying active and former project's students on motivational aspects

Following previous literature, we examined the motivation of currently active students of the Catroabt project. As described, we asked 103 active students and received feedback from 58 of them (56% response rate) in an anonymous online survey. The results align with previous studies that outline the importance of future rewards for the motivation of contributors [Hars and Ou, 2001]. As illustrated in Figure 5.3, the majority of students see the credits they earn for participating (72%), the learning of new skills (69%), and the experience they gain (64%) as motivators for their participation. At the same time, the project's vision and idea also got identified as an important motivating aspect for almost all of the students (78%). This matches research by Hars & Ou [Hars and Ou, 2001], who found that students are strongly motivated by intrinsic factors (self-determination and altruism), but also strengthening their human capital. One factor that surprisingly has not been rated as high as we expected by the surveyed students is the potential impact on their future career. Only 36% of the participating students claimed that they see it as a reference for their career

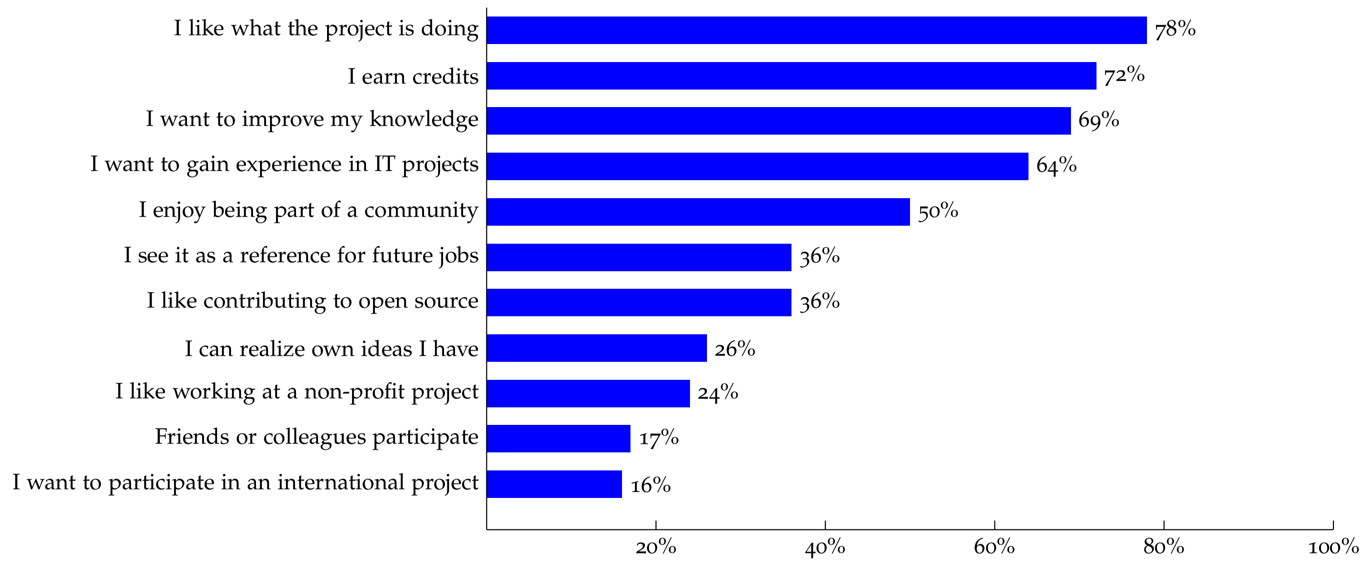


Figure 5.3.: Catrobat students' motivators to join the project (Figure as published in [Müller et al., 2019a] CC-BY-NC-ND)

after graduation. This also aligns with the results that came up for the questions how relevant they would rate their contribution to the project for their career. Students showed very mixed feelings from "not at all" to "very relevant" about this question, without allowing us to get a more in depth answer to this question. This is in contrast to the conducted survey of the project's alumni students who already graduated from university and stopped contributing to the project. We received answers from 31 out of 98 former students asked (response rate of 32%). 71% of these alumni, all of which are working in a related field (e.g., software development or project management), claimed that contributing to this project helped their career, compared to just 36% of currently involved students who subjectively see a benefit for their future career related to their contribution. One possible reason for this could be that students still at the university may not be able to clearly foresee the benefits for their later career, since they do not yet have had the experience of working in industry. Alumni students further had the opportunity to anonymously provide an optional comment within the survey. 13 of the alumni left a comment, of which 8 positively highlighted the impact of working on this project to their career. Especially the usage of professional tools such as Jira or Confluence, the application of agile methods, and working in interdisciplinary teams got pointed out in these comments. Although there is a strong probability of a non-response bias, since there is the chance that only students who have positive feelings about the project participated in the survey, we nevertheless can infer that participating in open source projects during university studies can have a positive impact on the students' later careers. This is also underpinned by the personal informal feedback we received from local ICT companies that employ former students. They state that the onboarding of these employees is sped up due to their previous knowledge in working in development teams, applying common software development methods, and using professional software tools (e.g., git or Atlassian).

Another factor we surveyed has been the contribution of students to other open source projects. 36% of the surveyed students of Catrobat have already been contributing to other open source projects before their involvement in Catrobat. This aligns with the results of an additional survey done with 104 students of a coding course at the university. A similar number of these students (35%) has already been contributing to open source projects. Also

42% of the surveyed alumni stated that they did so. We can see that students, but also alumni, have an active interest in open source. By establishing the possibility of working on open source projects during their studies, students are supported in their interest in such organizations and furthermore gain important experience in practical software engineering.

5.1.10. Discussion

This section describes a single case that has already been running for several years and was developed from the needs of the involved students and educators. We are aware that the results on the motivation of students and long-term effects on alumni can only be snapshots based on the data over a short period of time. Further insights into the long-term effects are expected from a continuous evaluation of new and leaving students of the project. The described setting shall help other institutions to establish similar projects. More published cases can help to further analyze and evaluate the development of open source software as part of students' university work. Our results suggest that this approach can be repeated at other universities and can help to prepare computer science students for their later career. Nevertheless, we want to encourage more researchers at universities to report their personal experiences of developing open source software in classes in order to create a larger basis for research in this field.

5.1.11. Conclusion

The presented approach of bringing open source development by students to universities comes with many benefits but also challenges, especially for the involved educators. The students' personal perception of this approach is very positive and considered as beneficial by alumni. There are strong indicators that this practical setting has a positive effect on the participating students' later career, since they get exposed to real-world problems, have to work in teams, and get to know common professional tools in the field. Furthermore, students in general show a strong interest in open source software. By enabling them to bring this interest to courses, they can gain knowledge and advance their studies at the same time. Also researchers can benefit from the described setting, as it fosters the general research in this domain and gives direct access to real-world problems for potential research projects. Manifold possibilities for research are created, as the case of Catrobat and Graz University of Technology shows. Especially from an organizational point of view (e.g., providing the infrastructure, guiding students, and keeping track of their involvement) additional work and resources have to be invested compared to traditional course settings. But the personal experience of all involved entities shows that the gained benefits, at least in the presented case, outweigh the effort.

5.2. Pocket Code: Enabling Teenagers to Create and Share Apps[4]

In the last years, a variety of visual coding tools emerged and made learning how to code easier than ever before. The block-based coding approach, used by most tools, follows common educational principles and aims to provide an excellent learning experience for students. Hence, many schools and teachers nowadays use such coding frameworks to teach principles of programming as well as to promote computational thinking, logical reasoning, and problem solving. Although there are some drawbacks, students prefer this visual approach and find it easier than text-based languages to learn coding [Weintrop and Wilensky, 2015]. The mentioned skills that arise from learning to code are beneficial for our economy and society, that more and more rely on computer technology. Block-based frameworks have proven to motivate students to take more computer science classes in the future [Weintrop and Wilensky, 2017]. However, coding should not only be seen in a career context, but also as a form of expression and way to learn something new [McManus, 2013]. Important aspects of all visual coding frameworks are sharing, communication, and collaboration. The social component of exchange, either programs, resources, is omnipresent on such platforms. As an example, users of the popular Scratch coding framework,

[4]This section is published as [Müller et al., 2018] ©2018 IEEE. DOI: 10.1109/ICOS.2018.8632815

developed by the Lifelong Kindergarten Group at the MIT Media Lab, as of July 2018 created and shared over 32 million projects and left more than 163 million comments on these projects [Lifelong Kindergarten Group at the MIT Media Lab, 2018]. This nature of sharing is a core principle for Scratch and motivates many of their users by getting advise, feedback, and reaching a large audience for their created games, animations, and other programs [Resnick et al., 2009]. Most frameworks inspire users to share their programs under an open license, fostering co-creative and collaborative communities with an open mindset.

Not only coding-tools are on their rise, but also smartphones are gaining more and more importance. Our workshops in different schools have shown that, at least in Europe, basically all teenagers have smartphones with them during class. A study in 2015 highlighted that about three quarters (73%) of teenagers in the US have access to smartphones [Lenhart et al., 2015]. Nevertheless, they get rarely used and educators still prefer traditional PCs provided by the schools. Additionally, smartphones become cheaper and are more available in wide parts of the world, hence they are a cost-effective way to gain access to the Internet.

In this section, the case of the FLOSS (Free/Libre Open Source Software) project Catrobat and its Android app Pocket Code, a block-based visual programming framework developed for smartphones is presented. This case study outlines why coding on smartphones can be beneficial for teenagers and how these benefits influenced the development of Pocket Code. Based on a literature review and comparison of currently available tools, we first introduce general approaches for block-based visual coding and how coding can be realized on mobile devices. Second, we outline challenges for these mobile approaches that have been identified in literature and especially the case study. Third, we give insights into Pocket Code, how it handles these challenges and enables teenagers to create and openly share apps without previous programming knowledge. The focus of this work is to foster the understanding of how mobile coding tools can be developed to support teenagers in learning coding competences by creating their own apps and share it with a community.

5.2.1. Computational Thinking on Smartphones

Why Visual Coding for Mobiles?

Learning should be fun, put into context, promote creativity through open ended tasks, and should also provide social support for students [Bruckman, 1999]. In the educational context of computer science, the Lifelong Kindergarten Group at the MIT Media Lab started in 2003 to develop Scratch, a visual programming environment that makes coding easy to learn and tries to engage users of all ages [Maloney et al., 2010]. Sandoval-Reyes et. al. [Sandoval-Reyes et al., 2011] state in their analysis of the popular frameworks Scratch, Alice, App Inventor and Greenfoot that these block based programming environments foster learning how to code by "Connecting users with their interests" (ibid, p. 443). These frameworks promote the aforementioned aspects, since they may be used in various creative contexts and try to connect users in communities. These coding tools use blocks that are visually similar to Lego® bricks. The bricks are organized in different categories and users can create their own programs by dragging them together. These frameworks let users focus directly on tasks and not on surrounding aspects such as dealing with syntactical errors, handling messages, or compiling issues [Maloney et al., 2010]. Another driver are the frameworks' online communities which serve as a place to gain inspiration, learn from existing projects, receive support from other users [Resnick et al., 2009] and make friends. Users can upload and publish their projects under an open license directly via the framework, so other users can access and learn from them.

Nowadays, these frameworks are used in classrooms all over the world, as shown by a huge number of experience reports on the Internet. An educational approach that is often connected to such block-based coding platforms is constructionism. A simple description of this concept is "learning by making" [Papert and Harel, 1991]. The theory of constructionism is built on the idea that better learning is achieved by having fun in doing it and creating something meaningful [Bruckman, 1999]. The concept is a hands-on approach that does not rely on verbally expressed knowledge but on directly constructing something [Papert and Harel, 1991]. Connecting this approach with social aspects provides even more powerful and effective opportunities for educators [Bruckman, 1999]. Nevertheless, most of the mentioned frameworks are designed to be used on traditional desktop computers, and many of them need an active Internet connection to be executed. As coding-workshops done by our team have shown, many schools are still struggling with problems when it comes to traditional computer infrastructure used for these frameworks. Desktop infrastructure is cost-intensive and needs to be maintained continuously by staff members. Introducing mobile tools comes along with various benefits for educators and students.

According to statistical forecasts from 2016 by eMarketer[5] [eMarketer, 2016] about smartphone users worldwide, an increase until 2020 to an overall of about 2.87 billion is estimated. Comparing the IDC statistics about personal computer shipments[6] and smartphone shipments[7] worldwide, one can see a trend that smartphone shipments are increasing to 1.697 billion [IDC, 2018a] whereas personal computer shipments level off around 250 million units [IDC, 2018b] per year until 2020. Considering these figures, it is safe to assert that smartphones matter. Smartphones are not only considerably cheaper than personal computers (PCs), but due to their size users can have access to full computer power anytime and anywhere. A drawback of them is the rather small screen size and the lack of a real keyboard which makes it difficult to create programs in a traditional text-based way. This can be alleviated by using the visual programming paradigm in conjunctions with a decent user interface which lets one create code without dealing with syntax but to concentrate on one's programming task. Other important advantages of visual programming are the easy way of viewing programs in one's own mother tongue, easy discoverability, as well as simplicity of manipulating the programming elements. A major feature of smartphones are the built-in sensors and effectors, e.g., inclination or geolocation sensors, the camera, or vibration, which can be used for input and feedback. As Price and Barnes state in their 2015 comparison study [Price and Barnes, 2015], visual programming environments not necessarily improve learning in terms of understanding, but learning happens at a faster rate. Furthermore, block-based programming is perceived as being easy due to the following reasons: a.) blocks are easier to read than text based code, b.) shape and

[5]https://tinyurl.com/emarketer-Growth-Internet
[6]https://www.idc.com/getdoc.jsp?containerId=prUS43596418
[7]https://www.idc.com/getdoc.jsp?containerId=prUS43548018

color of blocks convey additional information, c.) block-based code is easier to compose, and d.) blocks serve as memory aids. Weintrop and Wilensky in their 2017 comparative study [Weintrop and Wilensky, 2017] did not find evidence that the enjoyment of programming is larger using a block-based instead of a text-based language. Nevertheless, their results show that the students' interest in computer science increased after using a block-based language, whereas it decreased when a text-based language was used during their course investigations. Noone and Mooney [Noone and Mooney, 2018] come to a similar conclusion "[...] teaching a Visual Programming Language [...] can have a very positive effect on their interest and retention in Computer Science" (ibid, p. 164), reasoning that they are highly accessible, easier to experiment with, and that knowledge overhead is lower than with textual programming languages. Combining the benefits of block-based languages with the opportunity to write code on de-facto omnipresent mobile devices, has the potential to reach a broader audience especially in emerging markets and make coding available for everyone.

Current Mobile Coding Frameworks

As of 2018 there exist several well established visual block-based coding tools that enable users to create smartphone apps. In Table 5.1 we compare a selection of popular tools by common characteristics. All of them give users the opportunity to create apps in an easy block-style manner. However, they differ in their structure, provided features, and targeted user-group. Thus, we decided to base the comparison on general characteristics on the service and organization itself that especially matter in a collaborative and educational context. Besides commercial solutions, e.g., Thunkable, also non-profit and academic projects based on open source are common. The main difference is whether the projects are created directly on a mobile device or on a PC. If created on a PC, the project must be transferred to and tested on a smartphone afterwards. Another differentiator highlighted in our comparison is the existence of an integrated community or sharing platform. Whereas App-Inventor or Pocket Code provide such a feature, e.g., AppyBuilder established a Show-Off thread in their forum to share links to apps created with this framework. Although there are similarities, there are benefits and disadvantages for all of them. Hence, no advice can

Feature/App	**MIT App Inventor**	**Thunkable**	**AppyBuilder**	**Sketchware**	**Pocket Code**
Website	appinventor.mit.edu	thunkable.com	appybuilder.com	sketchware.io	catrob.at/pc
Environment	Browser (Desktop)	Browser (Desktop)	Browser (Desktop)	Mobile	Mobile
App platform	Android	Android, iOS	Android	Android	Android
Organization	academic	commercial	unknown	commercial	academic
Costs	free	most features free	free	most features free	free
Free Open Source	yes	no	no	no	yes
APK export	yes	yes	yes	yes	yes
Active Internet connection required	yes	yes	yes	no	no
Integrated sharing platform	yes	no (thread in forum)	no (thread in forum)	yes	yes

Table 5.1.: Comparing current popular visual coding tools for smartphone apps (Table as published in [Müller et al., 2018] ©2018 IEEE)

be given which tool should be used. It depends on personal preferences and needs.

5.2.2. Requirements for Mobile Coding Tools

Co-Creative User Communities

Enhancing learning by technology is strongly connected to a social component. New technologies enable learners not only to seek information, but also to create and share it [Dabbagh and Kitsantas, 2012]. The Internet provides the perfect opportunity to do so. About 92% of American teenagers use the Internet daily, almost all of them with mobile devices [Lenhart et al., 2015]. Similar to social media platforms [O'Keeffe et al., 2011], the online communities of visual coding frameworks provide opportunities for enhanced learning and create benefits of connecting people with similar backgrounds, sharing one's own ideas, and fostering a community. Thus, they provide manifold benefits for young users if used in a meaningful context. Getting feedback, advice, and inspiration by sharing is also a strong motivator to use visual coding services [Resnick et al., 2009]. Nevertheless, further research on the motivation to share projects in an educational context is needed. An aspect that is mentioned often in such communities is the possibility to reach a broad audience for the created projects [Resnick et al., 2009]. Social media platforms that act as a place to communicate, collaborate, and exchange, are an example how teenagers use the Internet for social interaction. Pictures, videos, and other content can be shared easily, reaching users from all over the world. Previous research already lined out that social media platforms are common among teenagers and used regularly [Lenhart et al., 2015]. Hence, online communities and sharing features are essential for visual coding platforms. Currently, most sharing functionality in visual coding tools is limited within a community or via links to it. Especially in the mobile world various app markets emerged that enable developers to share their software globally. Thus, such tools should also consider alternative ways to share projects, to reach an international audience. As outlined in the comparison, certain tools already enable users to export projects to a common data-format (APK), allowing them to share their projects easily on the web.

5.2.3. Multi-Language Support for an International User-Base

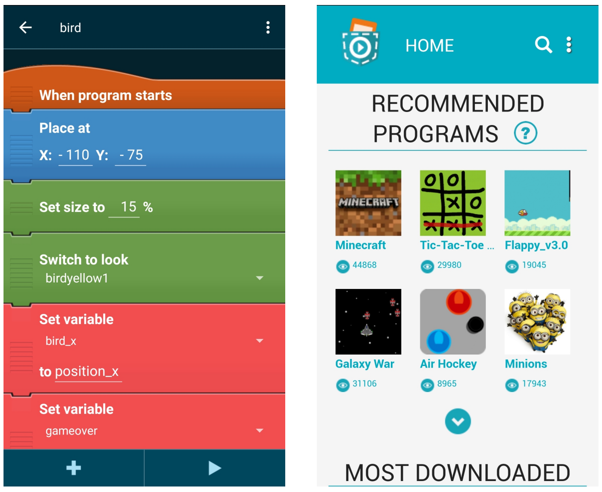
App stores help to reach an international audience and allow to scale apps easily to a global level. Nevertheless, doing so comes along with certain challenges. Several thousand different devices are currently running Android, having different screen sizes/resolutions, behavior, and technical specifications. Therefore, testing apps before publishing became essential to provide a proper user experience and to keep users. But even more important, different languages and cultural aspects need to be payed respect to, since users from various countries are likely to have different expectations and needs [Awwad and Slany, 2016]. As an example, the app abandonment rate is strongly related to internationalization and localization, as data collected by Google AdMob shows [Google, 2014]. Depending on the country the data was collected the number of users who have abandoned an app due to the lack of proper localization varies between 34% and 48%. Especially right-to-left (RTL) languages, such as Arabic or Hebrew, need special attention by software developers, since not only the text needs to be translated, but also the layout needs to be adapted [Awwad and Slany, 2016][Awwad et al., 2017]. Thus, the success of visual coding frameworks, especially on mobile devices, can be assumed to be strongly connected to these language aspects and needs to be considered in their development.

5.2.4. Creating and Sharing with Pocket Code

Pocket Code

The free Android app Pocket Code is a mobile visual coding environment designed for smartphones. Similar to existing desktop-based frameworks, such as Scratch or Snap!, the app uses graphical blocks as programming language and thus primarily targets beginners and students. Commands are categorized by their functionality, such as Event, Control, or Motion, and the framework prevents syntactical errors. Furthermore, special bricks for different hardware, e.g., Raspberry PIs, Arduino, or Lego® Robots, can get activated and used directly with the corresponding devices. A benefit, in comparison to desktop based solutions, is the possibility to access the device's sensors. A formula editor provides the opportunity to use this data from, e.g., the loudness-sensor, acceleration-sensor, or camera, in games, animations, and other projects. Thus, users can easily create interactive games, which are supported by an integrated physics engine. Furthermore, workshops at schools have already shown that such interactive games foster computational-thinking skills in physics or maths classes if students can try theoretical content directly with an hands-on approach on phones.

As illustrated in Figure 5.4, the app not only provides a coding interface but also an online community (presented in detail in section 5.2.4), that can get accessed within the app or via a web-browser. Whereas the app can be used offline without registration, the community features (e.g., uploading projects, and commenting) require to register for a free account and need an active Internet connection. Beside that, the app includes Pocket Paint,



```
When program starts
Place at
X: - 110 Y: - 75
Set size to 15 %
Switch to look
birdyellow1
Set variable
bird_x
to position_x
Set variable
gameover
```

Figure 5.4.: The Pocket Code coding interface and the community platform (Figure as published in [Müller et al., 2018] ©2018 IEEE)

an open source library developed by Catrobat, that enables users to create graphics for projects in a fast and easy way directly in Pocket Code. Features like essential drawing functionalities, layers, or transparency give users the freedom to creatively design their own characters or backgrounds. In addition to that, Catrobat also provides an online media library with a variety of free assets.

Community Platform

Smartphone penetration is still increasing. Walking through the streets nowadays underpins that mobile devices are part of our everyday life. However, in most cases users just download apps and are pure consumers of provided services. Pocket Code enables them to become active creators of their digital lives. As discussed before, especially sharing content on the web and participating in online communities for collaboration provides advantages for users and learners. Also Catrobat benefits from a growing online community, that provides feedback, support, and learning content. Looking up the statistics of Catrobat's sharing platform unveils that in the

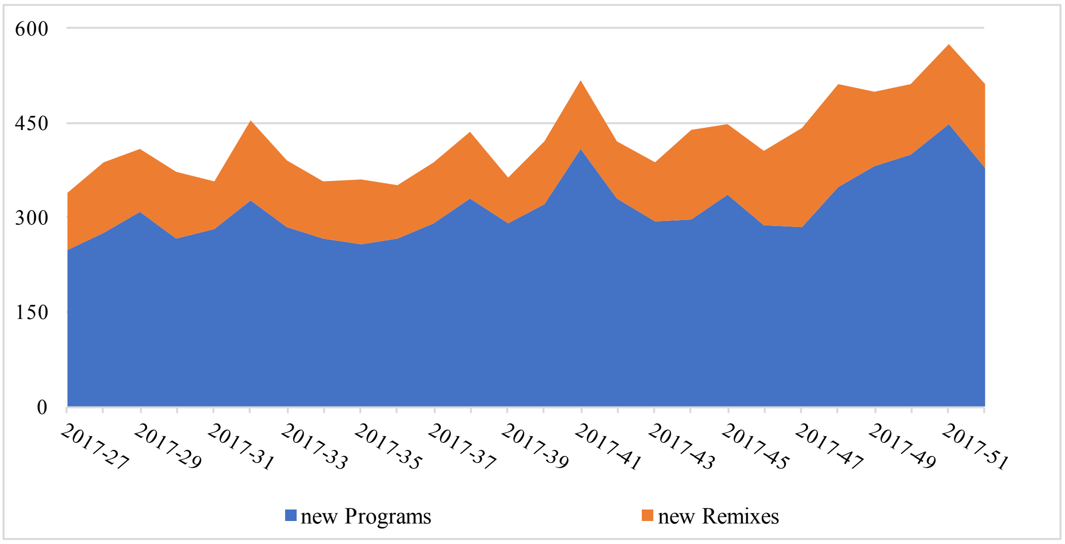


Figure 5.5.: Catrobat's sharing platform statistics for new uploaded projects (Figure as published as [Müller et al., 2018] ©2018 IEEE)

last years users uploaded more than 50,000 thousand projects (as of March 2018). As Figure 5.5 shows, each week on average 425 new projects, either totally new or remixes, have been shared between July and December 2017. It has to be mentioned that users are not forced to upload their projects, so it can be assumed that a much larger number of projects has already been created with Pocket Code. This proves that users take the chance to create their own games, animations, and other apps if they do have the possibility for it. In addition to that, these uploaded 50,000 projects have been downloaded more than 1,3 million times, implying that self-created apps are attractive for smartphone users.

Multi Language (In-App) Support

Looking up the download statistics of Catrobat's main app Pocket Code on Google Play shows that the support of different languages is essential for mobile applications. The overall download data (total user installs) from December 31st 2017 reveals that the app users choose a large variety of different languages, as Table 5.2 shows (dialects or regional accents not considered). The records show that Pocket Code users have 274 different language settings on their devices (including regional variants such as "en_US" or "de_AT"). These 274 settings represent 64 common languages, without considering different variants of a language. This shows that Pocket Code attracts a diverse user-base of different background and is not just focused on a certain region or group.

Of these languages, Pocket Code is at least partially translated into 57, also including the top 10 languages in Table 5.2. As a free open source project, Catrobat receives worldwide support from volunteers helping to translate all provided services. The possibility to use the app in one's mother tongue helps to decrease the abandonment rate mentioned before. Especially in a learning context, with young users that are not confident in foreign languages, this has proven as being beneficial. A further challenge that needs to be considered regarding the language aspect is the existence of different alphabets and scripts. As Table 5.2 shows, users are not only using Latin-languages, but also languages that are using scripts such as Cyrillic, Kanji, or Arabic. Figure 5.6 illustrates how this is handled in Pocket Code and outlines the representation of these alphabets and scripts. Most of these languages can get included into apps like Pocket Code relatively easily. However, right-to-left (RTL) languages, such as Arabic, come along with several additional challenges. Besides the pure textual translation, also context is needed. As an example, in Arabic numbers are still written from left-to-right, while the text needs to be adapted. Also layout elements, such as text views or buttons, need to be customized for such languages [Awwad and Slany, 2016][Awwad et al., 2017]. To achieve this, Catrobat gets supported by developers who are native speakers of RTL languages. This ensures that a large number of users can program with Pocket Code in their native language, especially improving the learning outcome of students.

Language	**Total installs**	**% total**	**% tracked**
Not tracked	169,104	38.10%	-
English	88,859	20.02%	32,34%
Russian	44,403	10.00%	12.51%
German	42,322	9.54%	10.59%
Spanish	17,890	4.03%	4.60%
Turkish	9,355	2.11%	2.20%
Polish	8,349	1.88%	1.92%
Japanese	8,258	1.86%	1.90%
Indonesian	7,315	1.65%	1.68%
French	6,889	1.55%	1.58%
Arabic	5,137	1.16%	1.18%

Table 5.2.: Top 10 used languages by Pocket Code users (measured on device installs, December 31st 2017) (Figure as published in [Müller et al., 2018] ©2018 IEEE)

Generating Standalone APKs

Pocket Code helps users to easily learn coding in an appealing and fun way. It empowers them to create feature rich applications directly on smartphones with no or little previous development experience. Users can share their projects on the community platform as open source programs for feedback, improvement, and as an examples for others to learn from. To

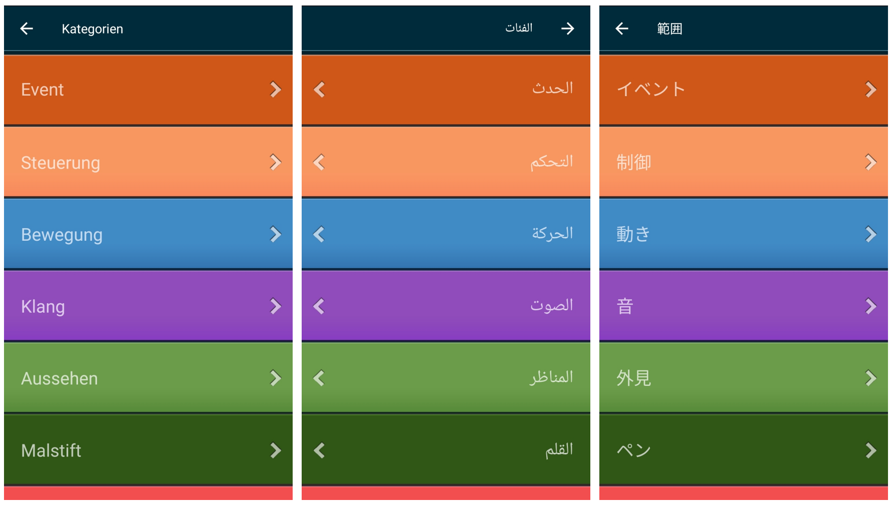


Figure 5.6.: Screenshots of the Pocket Code user interface for three different languages (German, Arabic, and Japanese) (Figure as published in [Müller et al., 2018] ©2018 IEEE)

reach a broader audience, e.g., to show the result to others without the need of having Pocket Code itself installed, users can transform projects to standalone Android application packages (APKs). These APKs can be installed on Android devices and used without the need of the Pocket Code IDE. Users can generate the APK of any project, but not yet customize their attributes (e.g., change the version code or version name). Furthermore, these APKs are currently created as debug versions and cannot be placed on app stores. For the first weeks of 2018, Figure 5.7 shows the number of APK generation jobs per week triggered by users. The rising trend demonstrates that there is an increasing demand for apps created with Pocket Code which can be shared and used standalone without the IDE.

5.2.5. Discussion and Future Work

Although, many benefits of this mobile approach were pointed out, there are still open questions that need further analysis. Especially for children and adolescents certain risks need to be considered regarding digital content and communication [O'Keeffe et al., 2011]. Legal issues, copyright, and

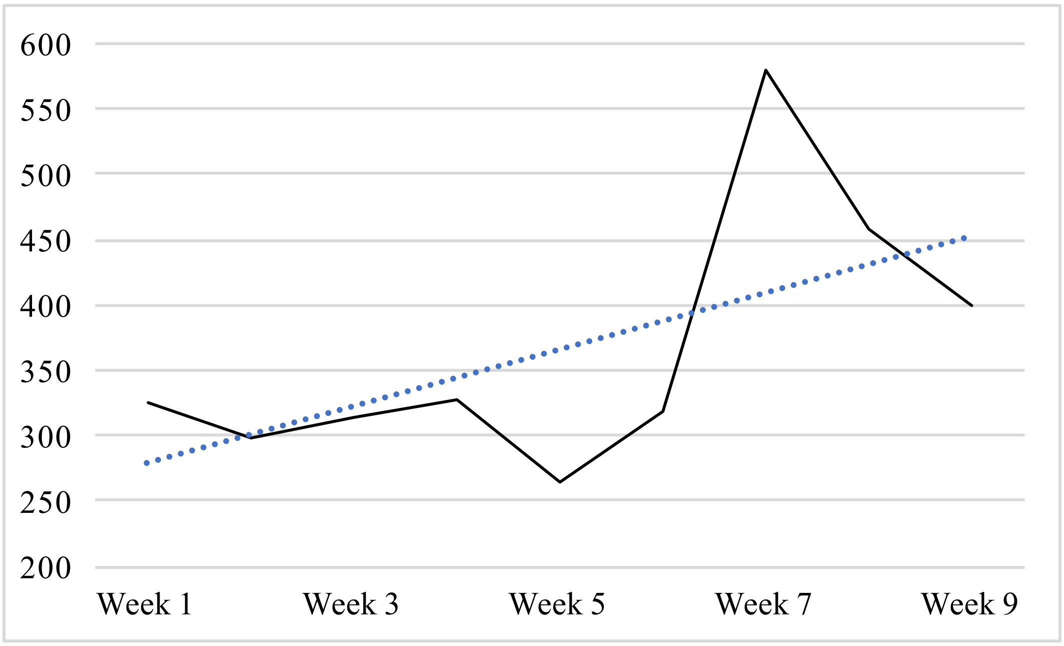


Figure 5.7.: Weekly requests by users for generated APK files in the first eight weeks of 2018 (trend highlighted in dots) (Figure as published in [Müller et al., 2018] ©2018 IEEE)

privacy need special attention. Since Pocket Code, that has been used as case for this work, is available to the public, it is necessary to limit data collection to an absolute minimum to protect the young target group. This includes that the terms of use must be kept simple, short, and easy to understand. However, this hampers research, since just limited data is available. Further work needs to be conducted how more knowledge can be gathered by ensuring the privacy and trust of the young users at the same time. Further, it is necessary to assess the young users' open mindset and their understanding of the sharing character. Many complaints by users show that especially children are upset if someone remixes or enhances their projects. This behavior has also been observed within Scratch, where certain features were added to encourage an open mindset [Resnick et al., 2009]. More information and context needs to be provided that sharing and remixing is a desirable activity which helps the community as a whole and nourishes innovation. Since exporting projects as APK files allows one to reach a broader audience, additional challenges might occur. Especially sharing remixed projects, or projects that have been created collaboratively, is potentially problematic. As outlined, the social and sharing aspect is an essential part of such frameworks. Further work is needed to gain deeper knowledge in these social challenges to provide a learning experience for young users that fosters an open mindset and the willingness to collaborate in a community.

5.2.6. Conclusion

Providing a mobile visual coding platform is an ideal complement for existing desktop-based frameworks. Users are enabled to learn coding and sharing their outcomes directly with a broad audience. Various tools are available that slightly differ in their nature, e.g., proprietary vs. open source, on what platform they can be used, or the communities' structure. We see many benefits for the mobile approach of Catrobat since coding can happen "on the go". By making the developed projects available on the sharing platform, users learn important skills for their future and help others to learn from their contributions. In combination with the supporting approach of visual coding, letting users solely focus on creativity and the semantic of programming, a promising learning environment is provided. Using therefore mobile devices is a cost-effective alternative to traditional PC based settings and pays respect to the changing digital society. Catrobat and its services give more users the opportunity to learn programming and unfold their creativity in the mobile world.

5.3. A Mobile Visual Programming Framework for App Development[8]

The usage of mobile apps in our every days' life became prevalent. Besides the professional software industry developing these apps, we also see the need for mobile apps that can get created without an extensive development-setup, within a short time and just little domain-knowledge. This need gets represented by the emergence of new tools, aiming to "democratize" (app) development and enable everyone to become a developer [Wolber et al., 2015]. Two examples illustrating this need are the domains of rapid prototyping and the maker movement, both increasingly using mobile tools and solutions especially in an IoT context. In these fields developers often just need proof of concepts or simple implementations, not intended for mass markets. Whether it is personal use or simple demonstrations in a professional context, resources to develop these solutions are rather limited and should, from our point of view, be possible without the necessity of a whole development team. One approach therefore are visual programming tools. Although primarily designed for educational purposes, frameworks such as the *MIT App Inventor* allow different kinds of mobile implementations also in a rapid prototyping context [Kang et al., 2015, Wolber et al., 2015, Adiono et al., 2019]. The visual and easy interface, that not necessarily requires previous programming knowledge, enables users to focus on what to implement and not how to implement it [Wolber et al., 2015]. This also refers to the growing Maker movement in that people more and more want to realize and share ideas in a community of same-minded people [Dougherty, 2012]. But also in a professional context these solutions enable the fast development of prototypes in various domains such as Smart Homes [Adiono et al., 2019].

In this work we present Pocket Code, a free open source application for Android[9] and iOS[10], developed by the Catrobat project[11] at Graz University

[8]This Section is published in [Müller et al., 2019c] ©2019 IEEE. DOI: 10.1109/MOBILESoft.2019.00027

[9]https://catrob.at/pc

[10]https://catrob.at/PCios

[11]https://catrobat.org

of Technology. This app allows to develop mobile projects in a fast and simple visual way directly on smartphones without the need for a PC or an Internet connection. Whereas both app-versions allow full access to the devices' sensors and cameras, the Android version further allows to control external devices via Bluetooth and WiFi, e.g., Arduino Boards or Raspberry Pis. In contrast to existing solutions, Catrobat's mobile approach makes programming even more accessible and eases the creation of apps "on the go". This enables manifold possibilities to quickly create apps not only in an educational, but also in a rapid prototyping and maker context.

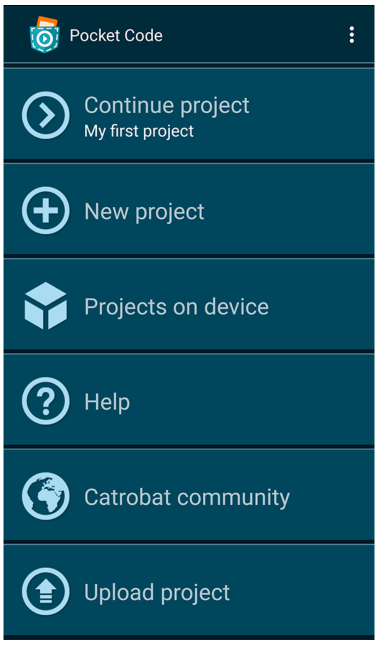
5.3.1. Catrobat & Pocket Code

Catrobat is a free open source project as well as a brick-based visual mobile programming language inspired by MIT's Scratch. The main focus of the project is the development of Pocket Code, an integrated development environment (IDE) for the language *Catrobat* on Android and iOS. With Pocket Code one can write programs directly on mobile devices. This is made possible through the paradigm of functional blocks which can be dragged together to form an executable program. Projects made with Pocket Code can be published on an integrated community sharing site[12] and also be transformed to APK files. Pocket Code is internationalized (i18n) and can be extensively localized (l10) including right-to-left languages. This makes it possible to be used by a global community where especially young users prefer to work with an app in their local language, minimizing barriers to start developing an app without previous experience in it.

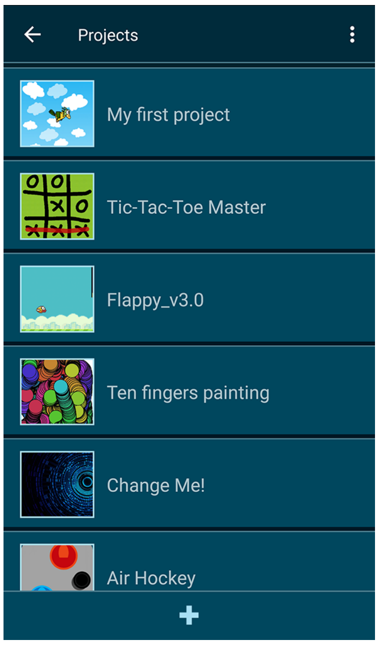
The app consists of four main parts (illustrated in Fig. 5.8):

- a.) The main screen to create new projects, access the existing projects on the device, explore/download projects created by other users from the Catrobat sharing site and upload new own projects to it.
- b.) The program list where all available programs on the device, either created by the user or downloaded, are listed to be examined in detail or executed.

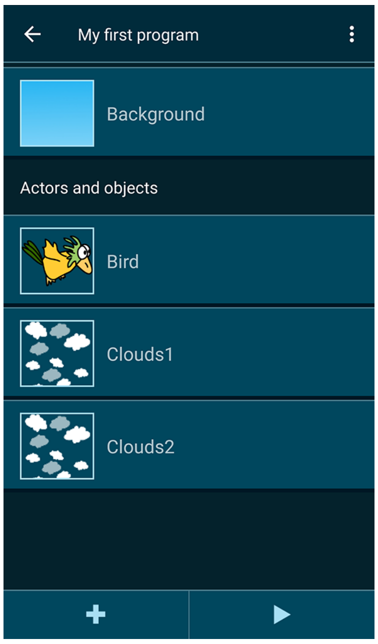
[12]https://share.catrob.at/pocketcode/



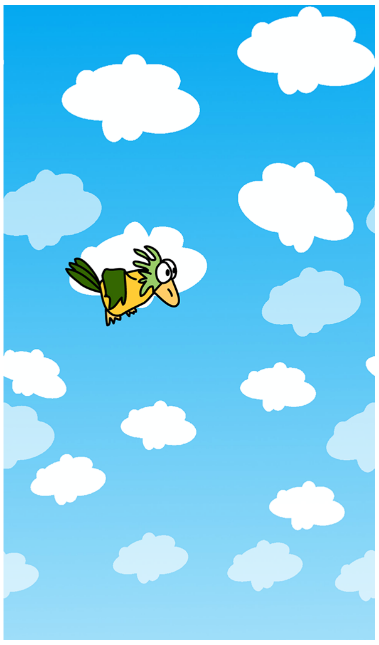
a.) Main screen



b.) Program list



c.) *My first project* overview



d.) Stage

Figure 5.8.: Pocket Code's main parts (Figure as published in [Müller et al., 2019c] ©2019 IEEE)

- c.) The project overview to manage objects, their graphics/sounds, as well as to create and edit program code.
- d.) The stage where the action and interaction happens when a project is executed.

A typical Pocket Code project consists of the background and optional objects (see Fig. 5.8c), each consisting of scripts, looks and sounds (see Fig. 5.10a-d). The object oriented paradigm of encapsulation is reflected by this structure. Tapping on *Scripts* opens the script view (shown in Fig. 5.10b), where one can add functional blocks which describe the behavior of the object. Initially, the script area is empty and must be filled by the user. This can be done by tapping the +-sign at the bottom of the script view and choose a brick from the various functional categories. In case of Pocket Code's default project ("My first project"), we have the background, the bird and two cloud objects (see Fig. 5.8c.). After tapping on the *Bird*-object and then opening its *Scripts*, one can see when scrolling down the screen, that the *Bird*-object contains three scripts. Two of them are having the same entry point - <*When scene starts*> (see Fig. 5.10b) - resulting in a multithreaded parallel execution when the project is started. The third script, defining the play-back of the object's sounds (defined in an dedicated area illustrated in Fig. 5.10d), is started when the *Bird*-object is tapped.

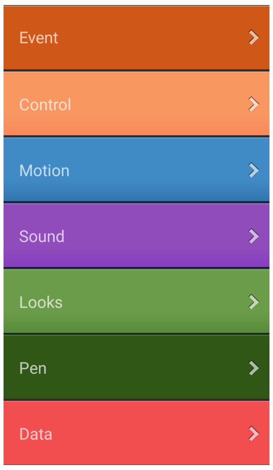
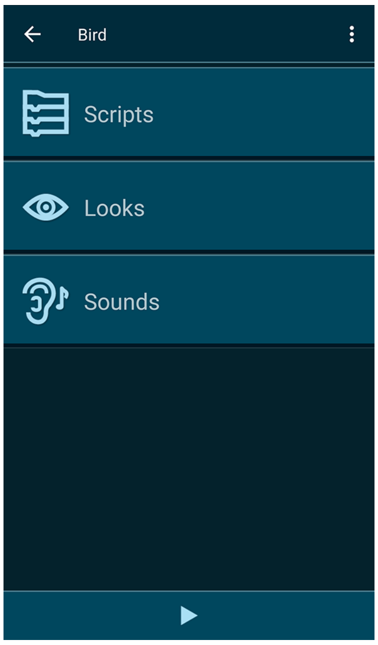


Figure 5.9.: Brick categories (Figure as published in [Müller et al., 2019c] ©2019 IEEE)

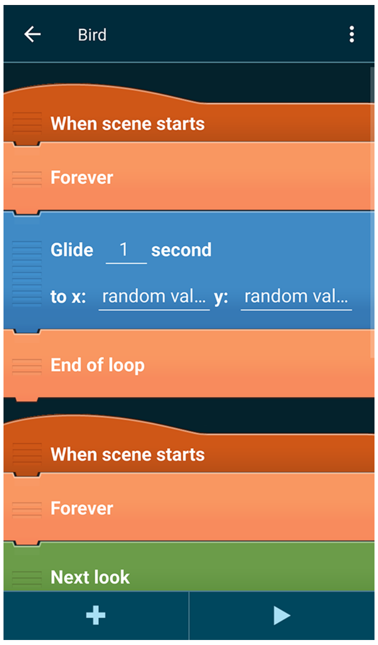
The colors of the blocks refer to their functional category. As illustrated in Fig. 5.9, seven default-categories (not including the extensions discussed in Section 5.3.2) are available:

- the ***event*-category** contains bricks related to events like *<When scene starts>*, *<When tapped>*, *<When stage is tapped>*, *<When you receive>* messages or *<When ... becomes true>*
- the ***control*-category** contains bricks related to the program flow like *<Wait ... seconds>*, *<Wait until ...>*, *<Forever>*, *<Repeat until ...>*, *<If ... is true then ... else . . . >*, etc.
- the ***motion*-category** contains bricks related to the movement of objects such as *<Place at X: ..., Y: ...>*, *<Change X/Y by . . . >*, *<Move ... steps>*, and *<Turn left/right ... degrees>*
- the ***sound*-category** provides bricks to *<Start sound>*, *<Stop all sounds>*, *<Set/Change volume . . . >*, *<Speak ....>* and others
- the ***looks*-category** contains bricks related to an object's graphical representation such as *<Switch to look>*, *<Next/Previous look>*, *<Set size to . . . %>*, *<Hide/Show>*, *<Set/Change transparency/color>*, etc.
- the ***Pen*-category** is related to drawing and stamping functionalities on the stage, such as *<Pen up/down>*, *<Set pen size/color to . . . >*, *<Stamp>*, and *<Clear>*
- the ***data*-category** contains variable related bricks like, *<Set/Change variable . . . >*, *<Show/Hide variable>*, *<Insert item into list . . . >*, *<Delete item from list . . . >*, and *<Replace item in list at position ... with . . . >*

With the bricks contained in these categories it is possible to form complex programs, which can be seen when exploring existing projects on the community platform. This is also related to the introduced concept of *scenes*. Scenes allow to structure several of the above described constructs within one project. This allows to create more sophisticated projects with different contexts, e.g., game levels where one can switch from one to another.

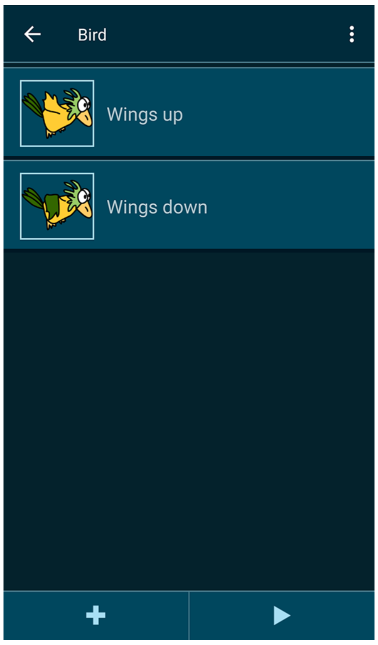


a.) Bird object details

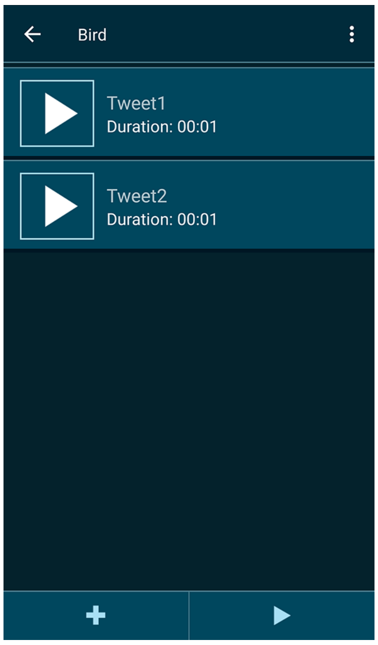


When scene starts
Forever
Glide 1 second
to x: random val... y: random val...
End of loop
When scene starts
Forever
Next look

b.) Bird scripts (partially illustrated)



c.) Bird looks



d.) Bird sounds

Figure 5.10.: Elements of the app's default project (Figure as published in [Müller et al., 2019c] ©2019 IEEE)

To further create interactive projects with Pocket Code, the following phone's actuators and sensors are accessible and can be integrated into one's programs:

- **Actuators**
  - Speaker/Sound
  - Flashlight
  - Vibration
- **Sensors**
  - loudness
  - touches_finger
  - acceleration x/y/z
  - inclination x/y/z
  - compass-direction
  - latitude/longitude (GPS)
  - touch detection (several sub functions)
  - face detection from the camera

This enables Pocket Code projects to react on the device's inclination and use this information to manipulate an object on screen. Furthermore, with the supported actuators one can make collisions between objects perceptible. Pocket Code also comes with an integrated painting app which supports layers, cropping, rotation, zooming until pixel level and allows to manipulate the alpha channel. With this integrated tool it is easily possible to draw simple objects to use them in Pocket Code projects which renders Pocket Code a versatile prototyping tool for creation of proof of concepts for simple games.

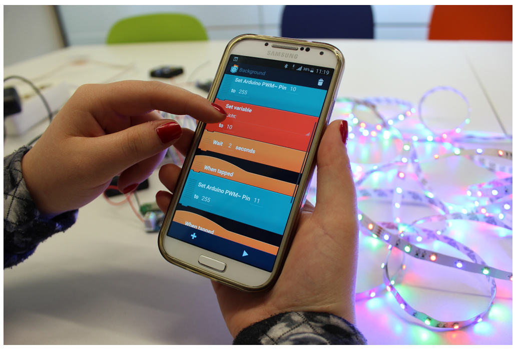


Figure 5.11.: The integrated Arduino extension used with a LED-strip (Figure as published in [Müller et al., 2019c] ©2019 IEEE)

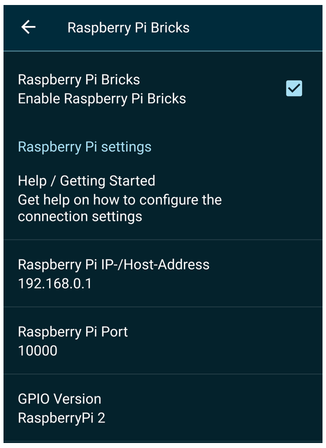
5.3.2. Hardware Extensions

Pocket Code comes with a variety of included extensions, allowing users to control external hardware within their created projects. Although these special bricks are disabled by default, the functionality is already included in the standard app and can get activated in the settings without any further download required. The extensions target educators, but also makers and developers with the need of fast working and easily created IoT solutions.

Arduino

An extension of Pocket Code especially targeting the IoT community allows the direct control of Arduino boards via Bluetooth. As illustrated in Fig. 5.11, visual bricks can be used to set the boards' pins to control external hardware, e.g. LED-strips, but also to read their digital as well as analog input pins as sensor values. This allows a two-way communication with connected Arduino boards and manifold possibilities to control hardware connected to the board. Further examples, available online[13], show how this feature may also be used in combination with other extensions, enabling users, supported by the multithreading character of the app, to create complex IoT projects in short time and without the need for deep domain knowledge.

[13]https://youtu.be/BHU2sgCRPtQ


Raspberry Pi Bricks

Raspberry Pi Bricks
Enable Raspberry Pi Bricks ✓

Raspberry Pi settings

Help / Getting Started
Get help on how to configure the connection settings

Raspberry Pi IP-/Host-Address
192.168.0.1

Raspberry Pi Port
10000

GPIO Version
RaspberryPi 2

Figure 5.12.: Settings for the Raspberry Pi extension of Pocket Code (Figure as published in [Müller et al., 2019c] ©2019 IEEE)

Raspberry Pi

Through a special provided server configuration for Raspberry Pis[14], optional bricks within Pocket Code also allow to create projects that interact with these devices. As with the Arduino boards, also these devices' pins can be set and read. The big benefit of this approach, as shown in Fig. 5.12, is that once the server is set up on the device, which can be done by simply following the instructions, just minimal settings are required on the smartphone to establish the connection and control it. This helps users to focus on the actual creation process of their ideas and not on complex connection settings.

Miscellaneous

Beside these two mentioned common IoT-devices, Pocket Code also supports a variety of devices with a more educational and end-user focus. Following additional extensions are currently included within the app and may be used as bricks in users' projects:

[14]https://catrob.at/RaspberryPi

- Lego NXT Robots
- Lego EV3 Robots
- Parrot AR.Drone 2.0 and Jumping Sumo
- Phiro Robot by Robotix Learning Solutions
- Google Chromecast
- NFC Tags
- Embroidery machines (currently in development)

New extensions, particularly enabled by Bluetooth and WiFi technology, are continuously developed depending on the needs of the users. The project tries to attract a broad target group to the app, also motivating users to try something new, e.g., tinker with Arduino boards or programing robots instead of solely being a consumer of predefined programs. Therefore, this list is likely to get extended within the next years, especially since innovation in ICT is coming fast and more possibilities for developers constantly arise.

5.3.3. Conclusion

In this section we presented Pocket Code, a mobile solution to create apps without the need of any extensive hardware setting in an easy visual way. Although visual coding is often solely considered in an educational context, this work highlights its potential to be used in a more creative development context. The still rising mobile market makes it increasingly necessary to quickly realize and test innovative ideas, either for own purposes, e.g., related to the maker movement, or for entrepreneurial objectives, e.g., rapid prototyping. The presented app provides essential coding functionalities to create apps connected to external devices without any great effort. This is not only beneficial for coding-beginners or people without professional background, but also for developers looking for a fast approach to implement and test prototypes. In contrast to existing solutions, Pocket Code runs solely on mobile devices, taking usage of the benefits of mobile technologies.

6. Catrobat's Ecosystem

6.1. A Complex Software Ecosystem[1]

As described in Section 3.1, the V²-framework is an adequate tool to analyze, communicate, and improve ecosystems through a value network perspective. The presented approach has also been applied to Catrobat, helping to better understand the involved actors and to identify potential bottlenecks. Figure 6.1 illustrates the project's ecosystem in V²-notation, which has been created in workshops with long-term contributors of Catrobat. The identified actors can be roughly classified in *Development Contributors* (who are actively and regularly engaged in the development of the provided services), *Nondevelopment-Contributors* (that just occasionally contribute to the project), *Supporters* (who provide additional services or other kind of support to contributors, the project and its users), and the actual *Users* (that are either just passively using the services, or also act as an active part of the community). Applying this approach brought to light several insights:

- Balanced value-exchange relations can be found within the network.
- Five dominant value-exchange relations can get identified as *Value Engines*.
- There is the need for an additional actor (visualized as *Supporter* with yellow arrows in Figure 6.1), that acts as a multiplier in terms of promoting Catrobat in, and networking with the user-community,

But, not only these insights have been beneficial for the project. The project's management further gained an overview of the involved actors that can get used for future strategic decisions. These decisions are intended to be made

[1]This section is based on [Vorraber et al., 2019a] ©2019 IEEE. DOI: 10.1109/MS.2018.290100810

by the newly formed *Product Owner Board*, that gets described in detail in Chapter 7 and requires detailed knowledge about the ecosystem to balance the needs of all members of Catrobat's community through the recently introduced processes.

6.2. A New Perspective[2]

The benefits of the enhanced approach, as described in Section 3.2, in practice can be shown on this case that was also used to define the framework, outlining the different points of view this enhanced layer structure provides compared to the before described V^2 notation.

As Free Open Source Software (FOSS) project "Catrobat" presents a collaborative project driven by a community situated in a complex ecosystem of various actors. As an educational project, Catrobat has a non-profit character following the vision of enabling teenagers to actively create their own apps instead of being merely consumers. The project thus provides various mobile- and web-services free of charge and is backed by volunteer contributors who are helping to bring this vision to life. In addition to the more than 600 contributors who have already been involved in the project, various partners from academia and industry are also providing their support to the project. Many different values are exchanged in a co-creative manner through the charitable ecosystem consisting of actors with different motivations, needs and wants [Vorraber et al., 2019a]. The collaborative approach of Catrobat's actors already enabled the project to reach more than 750,000 users with its free app "Pocket Code". Thus, it represents an interesting case of a co-creative community, generating a variety of primarily intangible social values by aligning different actors' actions. By collaboratively co-creating multiple values, not following money as currency for value and being accepted by society, this case from the domain of open source software is also highly related to common features that have been identified for sustainable business models [Jonker, 2012] (p. 30). However, this ICT enabled innovative project must also ensure its survivability through

[2]This section is published in [Vorraber and Müller, 2019] CC-BY. DOI: 10.3390/su11216018

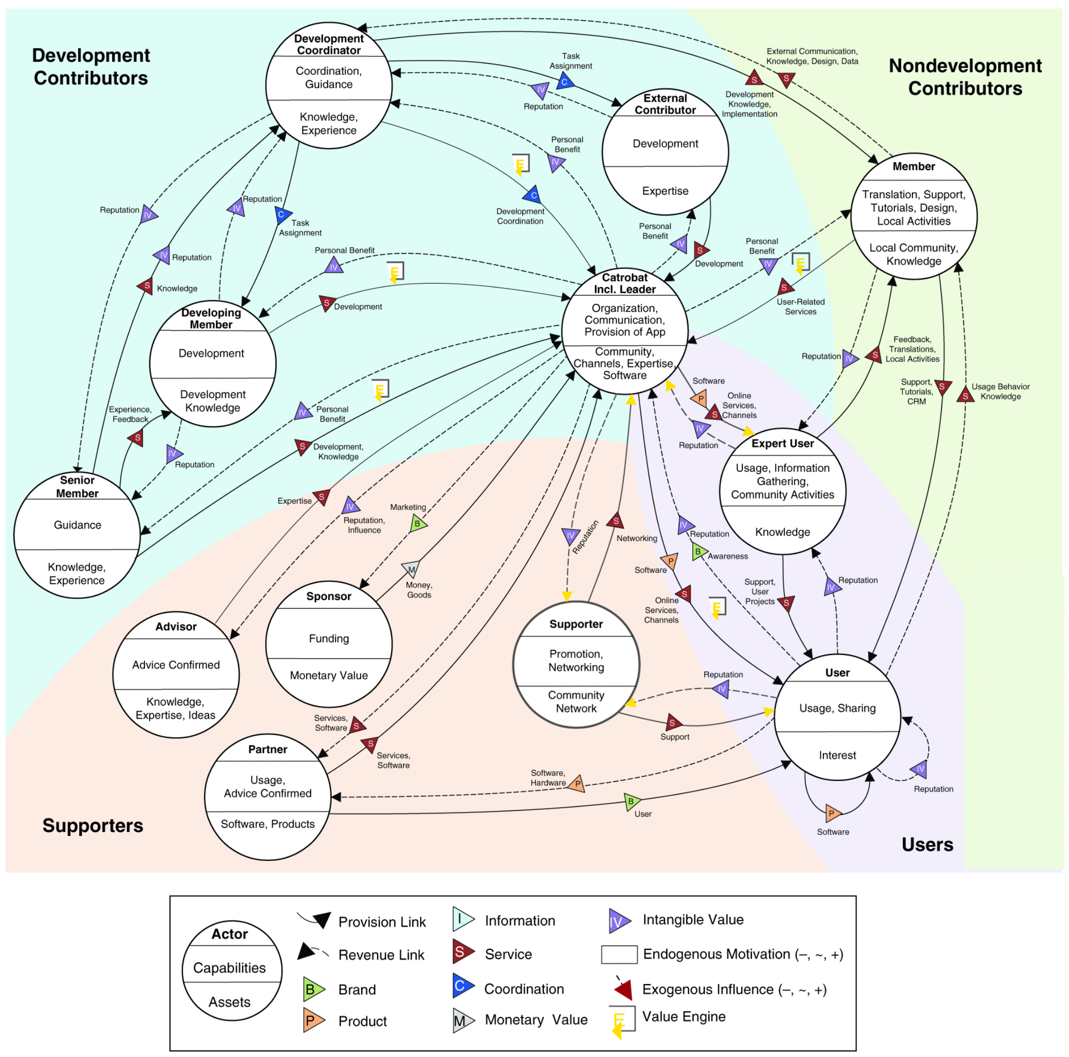


Figure 6.1.: The ecosystem network of the Catrobat project. The findings are highlighted.[Vorraber et al., 2019a] (Figure as published in [Vorraber et al., 2019a] ©2019 IEEE).

sufficient financial funding. Therefore, a healthy balance of these monetary values with the intangible values of the volunteer contributors, as well as with the ethical compliance to the project's vision, must be achieved.

The representation of this ecosystem in the existing integrated V^2 notation (see Section 6.1), covering the "Value Exchange and Resources" and "Dynamics and Motivation" layer in one model, has already highlighted benefits for analyzing and managing such co-creative systems that can be found especially in ICT [Vorraber et al., 2019a]. The application of the enhanced V^2 notation with multiple layers eased even further the analysis process for specific stakeholder groups. Although the applied approach in [Vorraber et al., 2019a] provides a holistic view over the basic system, which is useful for general management and decision making, the analyzing process is eased by the enhanced notation when certain aspects only, e.g., legal concerns or individual actors in relation to the overall system, need to be considered.

Furthermore, the presented framework supports the evaluation of certain actors in more detail, helping to achieve a better understanding of them and to highlight potential gaps that need to be filled (e.g., unmet needs) in the network. Whereas the overall value network notation of V^2, without enhancements, helps in providing a quick holistic view of the system, the presented layer-based approach is supposed to zoom in on actor level, as shown in Figure 6.2, to gain a high-level understanding that might be necessary for management and strategic decisions. The framework has been initially created in an expert workshop, as suggested above. Within this workshop of different long-term contributors of the project, each having a different background, the actors of the ecosystem were identified before drafting the network. By then evaluating the needs of each actor, as described in the networked values and needs layer section, and afterwards putting it in connection to the received values and relationships to other actors, the understanding of the if and how these needs are met can be easily created, visualized and communicated. Within this case, especially the motivational aspects and needs were further backed by a qualitative study in the scope of surveys [Müller et al., 2019a], helping to obtain an even more detailed understanding of them. In following iterations, in a smaller setting, the initial network was extended to the presented layers, creating a holistic picture of the ecosystem. As the example of Catrobat shows, this approach enables a focus on individual actors in a precise matter by also putting it

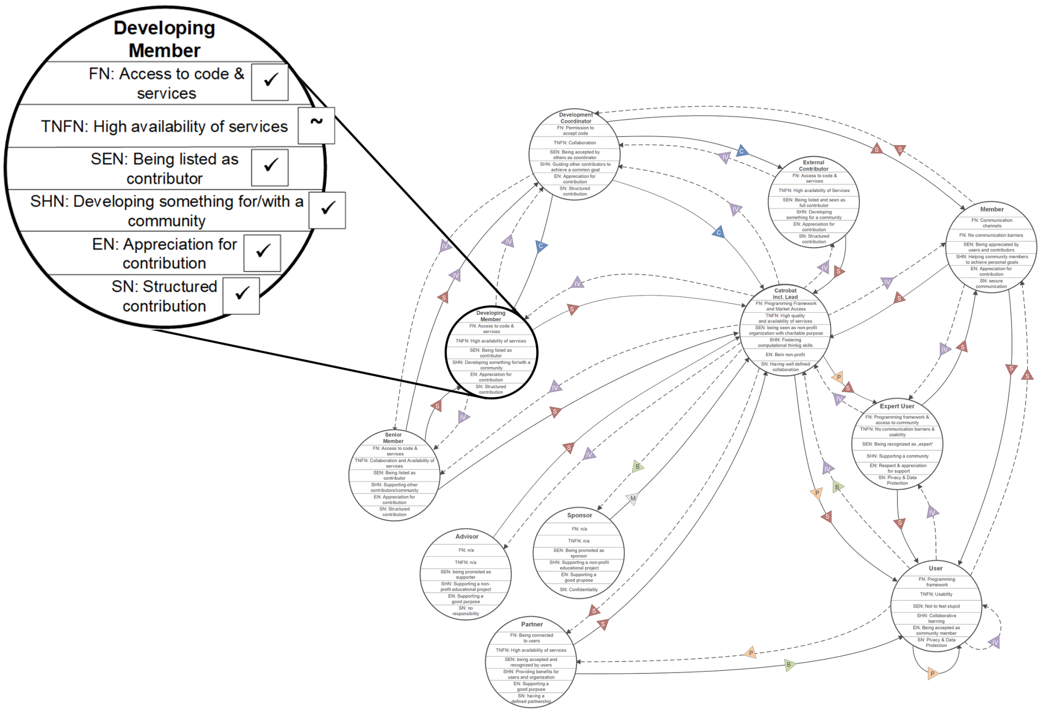

Developing Member
FN: Access to code & services ✓
TNFN: High availability of services ~
SEN: Being listed as contributor ✓
SHN: Developing something for/with a community ✓
EN: Appreciation for contribution ✓
SN: Structured contribution ✓
Development Coordinator
External Contributor
Member
Catrobat incl. Lead
Expert User
Senior Member
Advisor
Sponsor
Partner
User

Figure 6.2.: Zooming into the *Developing Members'* individual needs within the networked values and needs layer (Figure as published in [Vorraber and Müller, 2019] CC-BY).

into context of the value network within one single notation. To illustrate this benefit of the multi-layer network approach, we zoom in to actor level and stress the insights that can be gained in practice.

As already lined out in previous research, students, the main contributors of Catrobat, participate in this project for various personal reasons such as supporting the project's goal, seeing it as reference for their later career, or being part of a community [Müller et al., 2019a]. These valuable insights of specific actors can be represented in a very detailed level depending on the organizations' purpose of analysis, as illustrated in Figure 6.2. Especially needs and values that are met by received intangible values, e.g., in this case related to co-creating value with others or doing good for a community, can be represented in this layer and its different perspectives provided by the categorization. As an example, the developers have a social human need (SHN) to develop something for/with a community, which is directly related to the project's vision. However, the identified developers' need to have a reference for their later career is directly linked to the illustrated social economic need (SEN) to be listed, so being visible and recognized, as a contributor of the project in return, ensuring that the actor may gain an intangible net-benefit for their contribution. This representation of specific needs also fosters the understanding of NBMs that are often based on intangible and social values transferred in and co-created by open communities. As [Jonker, 2012] (p. 21) described it, collaboration and exchanging tangible as well as intangible values are important aspects for sustainable business models. Lining out these values and needs, as it was done in this case study, helps to better understand the personal motivation of contributing actors to participate in the ecosystem, as well as how their (social) needs can be met in return to keep them in the long term. The identified motivations, as done for this specific case in [Müller et al., 2019a], can be put into the context of the different categories and be seen in relation to the whole ecosystem. In this case, the enhanced visual layer representation also eases the communication process and documents the outcomes over time, which has already been outlined as beneficial in analyzing and managing complex ecosystems [Vorraber et al., 2019a]. This holistic understanding provided by the networked values and needs layer can be even further beneficial if detailed knowledge of a variety of different actors in a community is needed to manage them into a common direction, which, as in this case, is not necessarily primarily based on monetary profit or tangible values.

One example of how this layer-based viewpoint further proved to be beneficial is the legal layer of Catrobat, as illustrated in Figure 6.3. Whereas the connection between contributing actors is well defined by formulated open source software licenses and the joint work with partners on an individual basis, e.g., contracts or NDAs, the relationship to the end-users needs special attention. Especially currently ongoing and proposed changes in privacy/data protection and copyright law need to be considered for these specific connections. Adaptations to the end-users' terms of use and privacy policy might be needed in the foreseeable future and will therefore need to be prepared adequately. Especially the need for privacy, which is ensured through legal regulations, is directly related to social needs of the users, who expect their data to be safe, resulting in a want that needs to get represented on that layer. This shows that these layers may influence each other, making it important to combine them in one toolkit and visualize them for everyone involved in the decision making process.

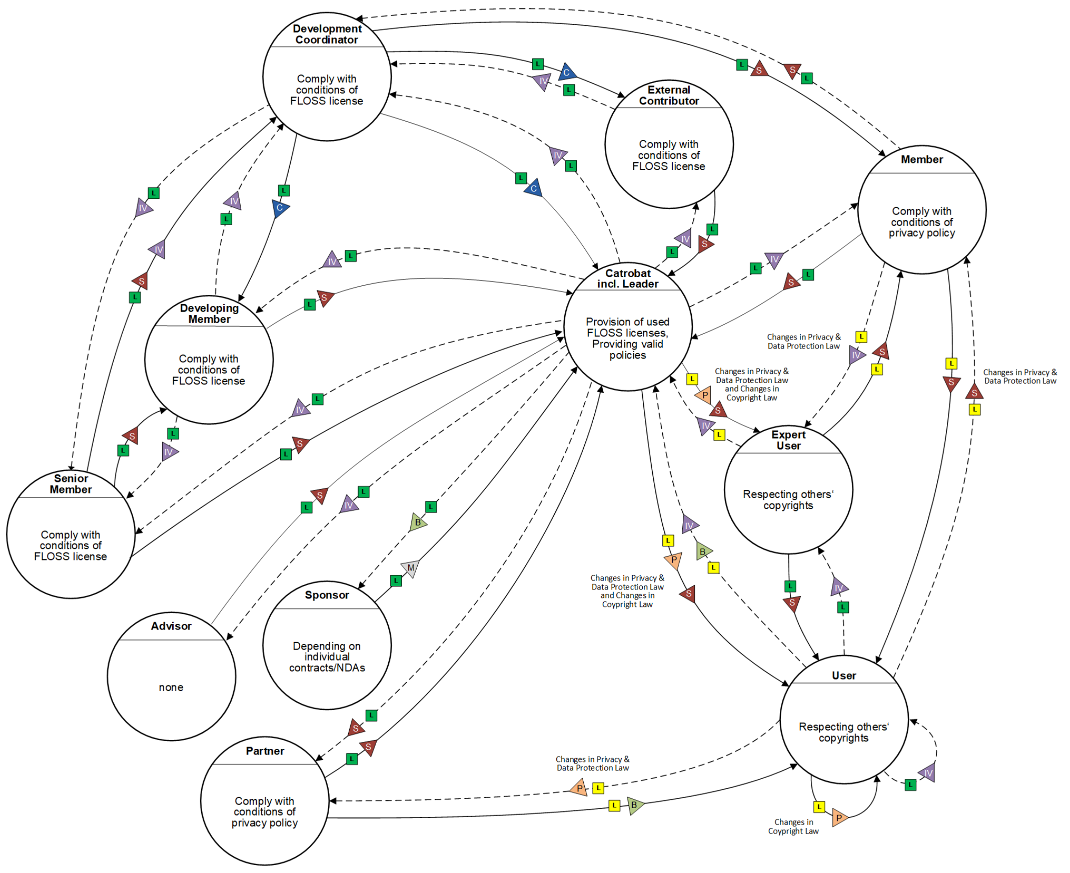

Development Coordinator
Comply with conditions of FLOSS license
External Contributor
Comply with conditions of FLOSS license
Member
Comply with conditions of privacy policy
Catrobat incl. Leader
Provision of used FLOSS licenses, Providing valid policies
Developing Member
Comply with conditions of FLOSS license
Senior Member
Comply with conditions of FLOSS license
Expert User
Respecting others' copyrights
Advisor
none
Sponsor
Depending on individual contracts/NDAs
Partner
Comply with conditions of privacy policy
User
Respecting others' copyrights
Changes in Privacy & Data Protection Law
Changes in Privacy & Data Protection Law and Changes in Copyright Law
Changes in Copyright Law

Figure 6.3.: The legal and value exchange layer of the Catrobat project (Figure as published in [Vorraber and Müller, 2019] CC-BY).

7. Introducing an Agile Workflow in a FLOSS Project

7.1. Challenges and Chances[1]

Free/Libre Open Source Software (FLOSS) can nowadays be found openly available in a rising number of different fields. Also firms identified various benefits of open principles for their work and are today actively involved in business models built around open source software [Chesbrough, 2006b]. Previous research already highlighted the potential of FLOSS practices also for other areas of economic and social activity [Von Krogh and Von Hippel, 2006]. In general, open source systems evolve while they are collaboratively driven by their communities [Nakakoji et al., 2002]. This collaboration in communities can be an active driver for open innovation and is therefore beneficial for businesses to engage in open source projects and the communities behind them [Von Hippel, 2001]. As a result, FLOSS and an open mindset are today well established not only in software development, but also in a variety of domains that benefit from collaboration and co-creation. However, governing these collaborative open communities can be challenging [Von Krogh and Von Hippel, 2006]. Whereas common software projects are often managed in an agile way, open projects come along with certain barriers when it comes to the application of agile frameworks [Koch, 2004].

Previous scientific work has already dealt with agile principles and frameworks, like Scrum or Kanban, for managing FLOSS projects and lined out chances, similarities, and dissimilarities [Warsta and Abrahamsson,

[1]This section is published as [Müller, 2018] ©2018 IEEE. DOI: 10.1109/ICOS.2018.8632819

2003, Koch, 2004, Harzl, 2017]. Taking a closer look on Raymond's characterization of open source, defined in his work "The Cathedral and the Bazaar" [Raymond, 1999], unveils that several aspects directly align with the principles stated in the Agile Manifesto [Beck et al., 2001]. Especially the focus on people, following a certain vision or idea, and continuous change are crucial for both. Introducing agile methods in such open projects potentially provides chances to improve the outcome, enhance contribution, and foster the contributors' community. As Cockburn and Highsmith describe, especially software related projects are ecosystems that benefit from different people, skills, and personalities and thus need processes that fit to the specific circumstances of the ecosystem [Cockburn and J., 2001]. This work evaluates how agile processes, respectively frameworks, are influenced by the characteristics of open (source) systems and how development as well as governing processes could be adopted to these systems' needs in practice.

Therefore, following research questions are considered:

- **RQ1**: How does the open character of FLOSS projects influence agile methods?
- **RQ2**: How can agile principles be practically fostered in a FLOSS project?

Researchers already dealt with similar questions, however, most of these publications have been written several years ago and FLOSS as well as agile methods evolved since then. Based on different data, this work aims to evaluate if this research on agile frameworks and open source is still valid, how open projects practically can address identified agile challenges and chances, and what actions can support agility in such projects today. Besides a research oriented view, these lessons learned also aims to target practitioners in industry and academia.

To do so, the case study of Catrobat[2], a non-profit FLOSS project established at Graz University of Technology, is considered. Within the last years open source projects became an important part of research and higher education.

[2]https://www.catrobat.org

As one can see on the public FLOSS directory Black Duck Open Hub[3] there are numerous educational institutions actively driving FLOSS projects. This backs the assumption that open source and media became an essential resource for research and educating students. Therefore, the presented case-study represents a common domain for open projects and provides valuable insights into the agile management of FLOSS communities.

7.1.1. Methodology

Previous literature (e.g., [Koch, 2004], [Turnu et al., 2006], [Tsirakidis et al., 2009], [Harzl, 2017]) already dealt with applying agile methods to FLOSS projects. Furthermore, examples (e.g., [Warsta and Abrahamsson, 2003]) have shown how open source principles, especially the aspect of collaboration, can also support agile projects. Hence, there is a broad basis for combining these fields, which has been considered for this work. The proposed research questions are of a descriptive and exploratory nature, targeting to foster the understanding in this practice-oriented domain. Using the case study methodology, combined with a literature review, is well suited to answer the presented "how" questions [Yin, 2009]. The research questions are focusing on well-defined factors in literature that may influence the agile management of such open cases, setting a clear boundary that is important for this methodology [Baxter and Jack, 2008]. Catrobat represents a typical open source project that is an appropriate case for the used single-case study design [Yin, 2009]. Additional quantitative data and the results of an online survey (sample of 58 contributors of Catrobat in spring 2018) [4] further help to enhance the answers to the research questions and to foster the understanding of the topic [Baxter and Jack, 2008]. This work not only aims to give a basis for further research, but also to provide insights that might be useful for practitioners and researchers working on similar situated projects.

[3]https://www.openhub.net/explore/orgs
[4]http://catrob.at/membersurvey

7.1.2. The Case of Catrobat

Founded as FLOSS project at Graz University of Technology in 2010, Catrobat aims at creating mobile coding tools for teenagers. The project is enabling them to create apps directly on the phone through an easy to use visual programming framework. In addition to that, it fosters openness by sharing all created programs under an open license. Although Catrobat attracted contributors from all over the world, the project is primarily driven by students from the university, who may contribute voluntarily to it as part of their curriculum. Although they have many freedoms in the way they work and on what they work, an overall direction is provided to ensure a certain outcome for the project. Such a provided direction, specified by project leaders, can also be found in other open source projects and is not specific for this case [Nakakoji et al., 2002, Lerner and Tirole, 2002, Ye and Kishida, 2003].

Since its kick-off eight years ago, more than 600 contributors worked on the project in different fields. Together they created the free Android app "Pocket Code"[5] and other connected services (e.g., extensions for robots or a recommender system). More than 500,000 users already downloaded the app from Google Play, not including users who obtained it from other marketplaces. As of August 2018, users uploaded more than 60,000 therewith created programs on the project's sharing and community platform[6] and made their created programs available for everyone. Especially this sharing character helps to spread the principles of open source and an open mindset among the young target group. Tools, such as the app and connected services, allow users to innovate on their own and, by freely sharing these innovative programs, also to provide benefits to other members of the community [Von Hippel, 2001]. Therefore, not only contributors are an active part of the project's community, but also users who need to be considered in Catrobat's development and management processes.

The project's development process already paid attention to aspects of XP, Test Driven Development (TDD), and Kanban [Harzl, 2017, Fellhofer et al., 2015]. But till today just specific parts of these frameworks have been applied

[5]https://play.google.com/store/apps/details?id=org.catrobat.catroid
[6]https://share.catrob.at

to the project (e.g., a workflow similar to Kanban, planning games as specified in Scrum, or a clean code policy originating from XP). Nevertheless, these actions primarily focused on development aspects of the project, just loosely related to organizational and managing tasks. Especially the internationalization and growth of the project make it necessary to actively foster communication and project management, to enable further success and avoid additional challenges in the future [Fellhofer et al., 2015]. As it can be observed, many, mainly commercial, software projects already profited from agile frameworks and their application in practice for these tasks. Another factor that has been considered in the decision to introduce agile methods is that these have shown positive results in teaching environments [Chao and Brown, 2009]. With more than one third of newcomers being students, they got identified as a major group of contributors joining open source projects [Hannebauer and Gruhn, 2017]. Thus, bringing more agility into the project will also influence participating students and their learning outcome, helping to drive the project forward.

By its structure, the project is situated in a complex environment of different contributors, stakeholders, and users that are constantly changing. With this ongoing change and the open mindset the project is embracing, it is well suited to be used as a case for the proposed research and to give detailed insights into the addressed domains.

7.1.3. Agile Chances and Challenges for FLOSS

Since Catrobat already paid attention to agile methods in the past, first results in regard to the proposed research questions can get identified by analyzing contribution and development data. The used services for development, in particular the project's git repository[7] and a public issue tracking system[8], provide detailed empirical data on specific aspects of the project, e.g., structure and development processes, and help to gain an in-depth understanding of the case. Additional qualitative insights into the community are provided by the results of the mentioned survey.

[7]https://github.com/Catrobat
[8]https://jira.catrob.at

The Dynamics of Open Communities

Open source projects have a steady base of core-contributors and a large number of short-time or one-time contributors [Lee and Carver, 2017]. Furthermore, not all contributors work on the projects on a constant basis, episodic contribution is common and important, but often gets unconsidered [Barcomb et al., 2018]. In general, within open source communities several roles can be identified that vary from project to project but more or less fulfill similar tasks [Nakakoji et al., 2002, Ye and Kishida, 2003, Fogel, 2005, Crowston et al., 2006]. A common hierarchical model for these roles is based on an onion-like structure, differentiating highly-active core contributors from co-developers and users who have less impact on the project [Crowston et al., 2006].

Analyzing commit data from 150 contributors to Catrobat's main repositories outlines that an average contributors makes commits for a time span of about a year (52 weeks). One third (34%) of the developers only contributed for a short period of less than half a year (26 weeks). In contrast, as shown in Fig.7.1, there are just a few contributors (21%) that have been involved for more than two years. As a result, there is a high turnover within the team throughout a year. Even though there are contributors that stay for several years, every couple of weeks contributors disappear and new ones need to be introduced. Just in the first six months of 2018, more than 50 new contributors joined Catrobat, not regarding one-time contributors or peripheral contributors that not officially became part of the community (defined as creating an account on the project's development platform). This aligns with the common observation that FLOSS communities constantly change and evolve [Nakakoji et al., 2002].

This proves previous research ([Koch, 2004, Crowston et al., 2006, Lerner and Tirole, 2002]) that states that although many people contribute to a specific FLOSS project, only a small number of (core-)contributors is responsible for the major outcome. But in contrast to the results observed by Koch [Koch, 2004], Catrobat's top 10% of contributors, measured by code contribution (defined as sum of lines of code added and deleted to the main repositories), are responsible for just 68% of the total change, compared to an average of 80% (of the total source code) observed in his work. Similar results as at

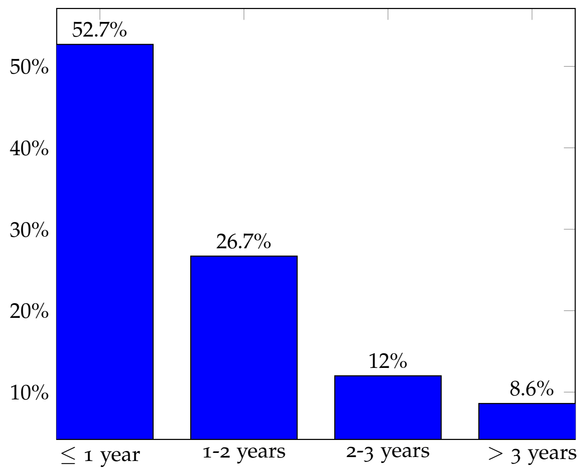


Figure 7.1.: Time span 150 analyzed contributors made commits to Catrobat's main repositories from the project start in 2010 to the end of 2017 (Figure as published in [Müller, 2018] ©2018 IEEE)

Catrobat, in detail 71% of changes by 11% of the contributors, have been observed in the analysis of the GIMP project by Ye and Kishida [Ye and Kishida, 2003]. We can see that FLOSS communities vary in their individual structure, but are all facing similar challenges of changing communities and a low amount of active long-term contributors.

Beside this repository analysis, 65 students from Graz University of Technology, representing the majority of community members, tracked during 2017 their contribution in a timesheet, giving further insights into their work. Visualizing this data, as done in Fig. 7.2, shows that contribution varies by the time of year and outlines that there is no steady development. Whereas many hours are spent on the project in spring, there are significant lows in summer and at the beginning of winter. In addition to that, by the nature of open source, this work is done remotely at different times of a day. As illustrated in Fig. 7.3, most contribution in this case is done on weekdays, but varies by the day of week, i.e., most hours are spent on Wednesdays, whereas the least have been tracked on Saturdays. This backs the assumption that no continuous development is going on and the contribution is dependent on the available time and commitment of the contributors.

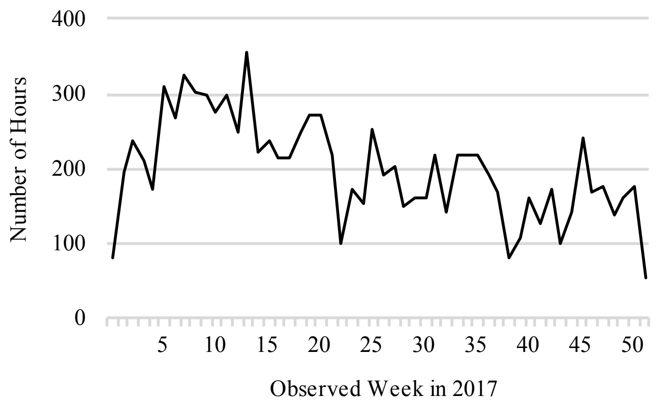


Figure 7.2.: Total hours spent by (65 tracked) student-contributors on the project per week in 2017 (Figure as published in [Müller, 2018] ©2018 IEEE)

A problem that comes along with the short contribution-time is the high effort to get into the project. As outlined by Harzl [Harzl, 2017], most students that get involved in the project only have limited knowledge in agile software development and programming skills vary. As literature on agile software development methods shows, teams get slowed down if new members need to get introduced to a project and agile methods [Brechner, 2015, Pichler, 2010]. This issue can also get observed within Catrobat. The conducted survey revealed that 36% of asked contributors see room for improvement in onboarding new members. Personal interaction with long-term contributors of the project pointed out that new members need a lot of guidance to get to know the project and need a notable amount of time to learn how to work within the existing project-structure, including its community, processes, and tools. This aligns with research on one-time contributors that unveiled that more than half of them encounter barriers, e.g., lack of knowledge or problems to get into the project, that cause them to give up [Lee and Carver, 2017]. These barriers are defined individually and vary, also depending on the mental models new contributors have of the community they are joining [Hannebauer and Gruhn, 2017]. However, contributors that overcome these barriers can evolve to more active roles in the community, strengthen it, and ensure a sustainable development [Hars and Ou, 2001, Nakakoji et al., 2002].

Although new members potentially slow down software projects, they are essential for the viability of FLOSS communities in the long run. Consequently, barriers to become an active contributor need to be minimized. Therefore, organizations must be aware of what group of contributors they want to attract, since different contributors will face different barriers [Hannebauer and Gruhn, 2017]. A proper onboarding process and building a relationship with new contributors can facilitate this process of joining a FLOSS project [Barcomb et al., 2018]. At Catrobat, the introduction of pair-programming is supposed to do so and to ease the entry into the community for new developers. Pairing is common for agile teams and describes the social activity of two developers working together on site [Robinson and Sharp, 2010]. Introducing online pair-programming, as it has been suggested by 31% of the contributors in the survey, is a first step to ease the entry in the case and already shows first good results in individual cases. Nevertheless, the open and distributed character of FLOSS projects, resulting in a steady change of the community and its structure, comes with certain drawbacks when it comes to this agile method.

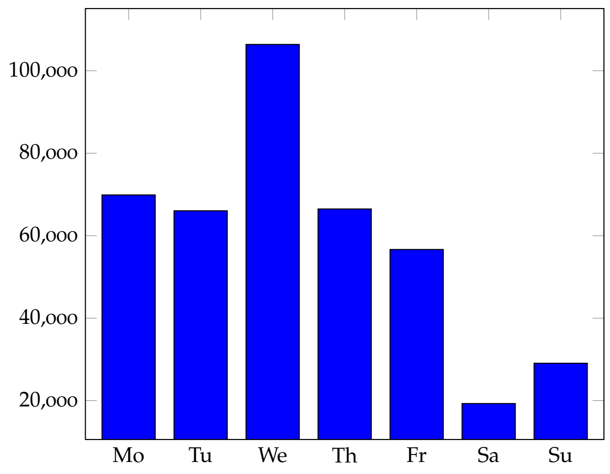


Figure 7.3.: Accumulated hours per weekday 65 tracked contributors of Catrobat spent on the project in 2017 (Figure as published in [Müller, 2018] ©2018 IEEE)

Communication

Catrobat's community not only consists of students, but also a rising number of external contributors. As in any FLOSS project they need to be considered by the project's leadership and actively connected with the community. Communication within open source communities is essential, but difficult to scale when projects grow [Fogel, 2005]. Especially if a lot of interaction with other contributors is needed within a project's community it is likely to get unmanageable [Lerner and Tirole, 2002]. Co-located events, such as face to face meetings, stand ups, or pair programming are common for agile approaches such as Scrum or XP [Robinson and Sharp, 2010, Brechner, 2015, Pichler, 2010, Schwaber and Sutherland, 2017]. However, as it turned out also for Catrobat, they are just possible on a limited basis for FLOSS projects due to the physical distribution of contributors and the varying times worked on the project. Thus, virtual communication has already been pointed out as a main challenge for open projects like Catrobat [Fellhofer et al., 2015]. The survey conducted in early 2018 also backs this, since 38% of the contributors state that there is still room for improvement in such communicational aspects. Currently several tools get evaluated to enable this virtual communication within Catrobat, supporting volunteers to collaborate and to share knowledge within the project.

Managed the right way communication in FLOSS can also be a key to success [Tsirakidis et al., 2009]. Whereas Catrobat has used IRC (Internet Relay Chat) in the past, it switched in 2017 to Slack[9], a modern and by the contributors accepted communication channel. Just from June 2017 to June 2018 more than 13,000 messages have been sent by 116 Catrobat members on these channels. Furthermore, the number of weekly active users is constantly rising, indicating that the community is accepting Slack as main communication platform. As these numbers show, the social and community aspect in FLOSS organizations, that is crucial for the motivation of contributors [Hars and Ou, 2001, Ye and Kishida, 2003] and the agile focus on people [Cockburn and J., 2001], gets fostered by such communication channels. They encounter the disadvantage of not being able to hold regular physical meetings by having steady exchange about ongoing work on these

[9]https://catrobat.slack.com

channels. As a result, although a notable percentage of contributors is placed in Austria, Catrobat, just like other FLOSS communities [Ye and Kishida, 2003], is dependent on a distributed virtual community, hindering the proposed co-located approach of various agile frameworks. However, this is also a chance for open projects as this case shows. It eases the entry for new contributors to get into the community, since it can happen online independently of time and place and also forces contributors to constantly exchange about the project's progress.

Self-Organized Teams

Over the years several teams emerged within Catrobat, taking responsibility of defined parts of the project (e.g., developing certain features or creating educational resources). Most of these teams originated either in external requirements (e.g., to foster partnerships) or in the initiative of interested contributors. Teams that have its roots in individuals, as it happened for extensions for Arduino or Raspberry Pi, align with the principle of open source that good software starts by solving a developer's personal itch [Raymond, 1999]. These students have a strong personal motivation to participate in the project, as it can be found in other FLOSS projects too [Ye and Kishida, 2003, Hars and Ou, 2001]. In general, if the benefits, defined by motivation, exceed the costs of contribution, contributors engage in open projects and even drive innovation [Von Hippel, 2001, Lerner and Tirole, 2002]. To prevent failing, the greater goal of the community and contributors must be understood and considered in managing such open projects [Ehls, 2017]. This motivation and the correlating goals also impact the structure of and processes in the community.

Each team of Catrobat consists of between 3 and 20 persons, depending on the domain and current number of contributors. Agile frameworks usually define team size to be large enough to handle a project and to have all skills needed directly in the team [Schwaber and Sutherland, 2017, Pichler, 2010, Brechner, 2015]. A factor that needs to be considered for FLOSS teams is the non-constant work going on. Developers strongly vary in their weekly contribution. As a result, open source teams can generally be assumed to be larger to achieve the same outcome in the same time. However, it also seems to get problematic if too many contributors are working in a team, since communication increases with the number of involved contributors. Therefore, team-size can also be spotted to potentially hinder agility in such projects.

Another problem with these teams is that they are often dependent on specific contributors, usually the initiator who had the idea for it and other core-contributors. In addition to that, such teams partially rely on external requirements, in many cases originating from externally developed systems or hardware. As a result, there have been teams whose work has never been released, since the contributors disappeared, the requirements changed, or the work got obsolete. As an example, development of a Windows-Phone 8 version for Pocket Code has been suspended before it got released. But especially the mentioned dependency on contributors is potentially risky for FLOSS communities. Open projects need to be able to survive even if specific participants or also the projects' founders quit [Fogel, 2005]. Restructuring the teams by reducing dependencies on individual developers and keeping them on a size that is large enough to run it in a sustainable way, but also small enough to keep communication feasible, seems to be unavoidable to increase efficiency and reduce the risk of unfinished work. It is proposed to make the teams more adaptable and agile, what has been identified as effective way to respond to steady change and an uncertain future [Highsmith, 2002].

It is important to note, that change in FLOSS projects has to happen together with the contributors who participate voluntarily and therefore their personal needs have to be met. 60% of the asked Catrobat contributors stated that they feel to have the possibility to contribute their own ideas. This leads to the conclusion that open source contributors need the freedom to explore new things and therefore requirements must be reduced. If this freedom is not given and the direction of the project is more or less dictated by the project leaders, the chances of a fork, a separation of contributors, rises [Fogel, 2005]. Consensus must be reached that allows a general project direction but that also ensures the freedom of the individuals. As a result, instead of introducing new teams for individual ideas or requirements, Catrobat started to bring together contributors with similar interests (e.g., on infrastructure or web-development). Working in groups on these topics allows developers to bring in their own ideas, but also ensures sustainable development in a general domain that is beneficial for the project.

Governing FLOSS Projects

Open source projects rely on different actors, e.g., contributors and users. Governing such projects can be challenging as research in this domain already lined out [Von Krogh and Von Hippel, 2006]. Management in FLOSS should be shared either on a formal basis by defined roles or informally within the community [Fogel, 2005]. Having a leading role in FLOSS communities, also collaboratively fulfilled by several persons, is common for such projects [Lerner and Tirole, 2002, Ye and Kishida, 2003, Nakakoji et al., 2002]. Leadership in open projects is crucial for their success, since changes of the governing structure or missing to meet the needs of the community can potentially bring them to collapse [Ehls, 2017]. Also agile projects rely on a collaborative leadership that is based on information and trust [Cockburn and J., 2001]. Product owners, as described in Scrum, ensure, among others, value creation and specification of requirements in projects [Schwaber and Sutherland, 2017]. Scrum has a clear definition of this role, however, in practice projects adopted it to their personal needs [Sverrisdottir et al., 2014].

Although Scrum defines this role as a single person and not a committee [Pichler, 2010, Schwaber and Sutherland, 2017], a board of currently four persons, all experienced long-term contributors, has been introduced in Catrobat. Splitting this role is not new to industry and adapting Scrum to the individual needs of a software project can have certain benefits [Sverrisdottir et al., 2014]. As described, FLOSS projects face constant change in the community, reshaping the social structure on a constant basis [Nakakoji et al., 2002]. Hence, there is the need of a leading role that keeps track of the project, acts as communicator, and represents stakeholders, including all contributors, users, and external parties, by providing a common direction in the interest of the community. Especially providing this direction as well as being a motivating leader for the contributors can be seen as crucial tasks of this board. This provision of a total vision and helping to bring it alive, has also been spotted as an important characteristic of product owners in industrial practice [Sverrisdottir et al., 2014]. Since there are no fixed working hours and days, resulting in work and communication throughout a day and week, it got necessary to introduce a board instead of one single individual to fulfill the role of a product owner. This allows faster response times, less dependency on certain persons, and more direct contact to the large community of contributors and users. This role is in certain aspects already existing in many FLOSS project as leaders. Leaders in FLOSS projects are already now supposed to provide a shared vision and keeping the community together [Lerner and Tirole, 2002]. However, regarding these aspects it can be assumed to be beneficial if this role also considers the well-defined duties and responsibilities of agile product owners. Based on the presented insights this may be a perfect fit, since similar actions are already now implicitly performed by this leader role.

7.1.4. Findings

The presented case highlights that previously identified challenges for agile methods in open projects are still existing. A changing community, unregular commitment, and virtual communication challenge these projects to be managed and understood in their entirety by agile methods. Although FLOSS communities and agile principles evolved over time and have today many aspects in common, the dynamics of openness, arising from structural change and unpredictability of contribution, are characteristics that influence agility in FLOSS projects. These outcomes, in respect to RQ1, align with previous research, but are enhancing it with the introduced case and additional empirical data.

The approach of regarding FLOSS project managers as agile product owners seems to be a practical concept in regard to RQ2. The outlined influencing characteristics can also be an opportunity if managed the right way, as the case of Catrobat highlights. As it has been shown, agile methods and open source development have several aspects in common. Since the formal role of product owners at Catrobat has just been introduced in the end of 2017, there is currently no empirical data of its success. But, first reactions of contributors are positive, the response time for questions has been reduced, and communication got improved. Furthermore, workflows in the project (e.g., for documentation or development) have been simplified and have been made more transparent by this board. By introducing this board a first step has been made that has the potential to introduce more agility together with the contributors and adapt the processes to their needs.

7.1.5. Conclusion

Although the presented case of Catrobat is well-established and successful, this work shows that there is room for improvement in managing it an agile way, i.e., the full application of frameworks such as Scrum or Kanban. The results outline many potential advantages of agile methods in the presented project, but also highlight various drawbacks. Adapting processes to the social structure of a community and its contributors can help to handle these drawbacks. Catrobat reacted to the gained insights and defined steps to address the identified issues. Constantly analyzing the community with its individual needs, strengthening communication, and introducing the agile role of product owners, who are implicitly already existing in many FLOSS projects, may also help other open projects to drive them collaboratively in a common, innovative, and future-proof direction. However, open communities evolve over time and therefore also these processes must evolve and be adapted from time to time. By paying respect to this circumstance, agile methods may ensure survivability and success. This work indicates that there is still room for research on open communities and their management. Further cases, giving insights into such communities, can be identified as first step to create a basis for further work in this domain.

7.2. Introducing Agile Product Owners in a FLOSS Project[10]

An all-time issue within the project has been that the number of proposed issues, representing user-stories and bugs, grew faster than the number of solved issues. Therefore, the need of prioritization came up to ensure that urgent and important requirements get finished in time. Furthermore, contributors did not thoroughly maintain the publicly reported issues, e.g., bugs or missing functionalities, resulting in a constantly growing issue pool. As also outlined in previous research [Heppler et al., 2016], features requested by external parties often get unrecognized by core-developers. Lacking to meet requests from users can be frustrating for the community and may impact the project negatively [Ehls, 2017]. These aspects made it necessary to introduce a role to keep track of, sort, prioritize issues independently from their origin. Catrobat already applies several individual agile methods and various chances and challenges of these methods were already discussed in the past [Fellhofer et al., 2015, Harzl, 2016, Müller, 2018]. Due to the positive experiences with agile principles, also this new role was supposed to be based on agile methodologies.

7.2.1. Product Owner within Catrobat

The introduction of product owners required several organizational changes. Besides defining this formal role, also processes and communication needed to be adapted to the new circumstances, as outlined in the following sections. Although this role is common in industry, special attention is needed in open communities such as Catrobat. Their character requires to focus on the balance between the different needs of contributors, users and external stakeholders, that are all involved for different individual reasons. Also the constant change of the community and direct involvement of users comes along with further challenges.

[10]This section is published in [Müller et al., 2019b] - Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature Introducing Agile Product Owners in a FLOSS Project by Matthias Müller, Christian Schindler, Wolfgang Slany ©2019. DOI: 10.1007/978-3-030-20883-7_4

The Role "Product Owner"

Leaders in open source projects implicitly perform actions and fulfill responsibilities as they are described for product owners. In Scrum, a product owner has the responsibility for the backlog to maximize value and represent external interests [Lacey, 2012, Schwaber, 2004]. Furthermore, they need to communicate the vision of the desired product and be a leader for the team [Pichler, 2010]. It is important to note that these responsibilities are based on collaboration with the team, making it necessary to have a common understanding and language [Schwaber, 2004]. However, decisions by the product owner must be made visible to and be respected by all people involved [Schwaber and Sutherland, 2017]. *"The product owner is the one person ultimately responsible for the success or failure of the project"* [Lacey, 2012]. Therefore, the founder of the project and experienced contributors have been assigned with this role. Although, Scrum defines this role for a single person [Pichler, 2010, Schwaber and Sutherland, 2017], these product owners form a board, similar to a committee that is common for FLOSS projects [Lerner and Tirole, 2002]. Derivations of Scrum, also having multiple product owners, can be found in successful industry projects too [Sverrisdottir et al., 2014]. The decision therefore was based on the constant need of having a product owner available to the contributors, since previous research has shown contributors are working on an irregular schedule [Müller, 2018]. Therefore, constant availability for information exchange must be ensured. This has been identified as a main success factor for Scrum in industry [Sverrisdottir et al., 2014]. Furthermore, whereas in industry this role should be performed as full-time position [Pichler, 2010], in Catrobat, for a lack of resources, this role can just be fulfilled on a part-time basis, resulting in the need for several people.

Specific parts of the project also have *project owners* that are long-term and experienced contributors, having the same responsibilities and possibilities as the product owners within their specific scope. However, they are not allowed to develop user stories that they have specified themselves. This shall foster collaboration between all involved contributors. The co-structure of product owners representing sponsored goals, e.g., by research, cooperation or user-feedback, and projects owners originating from the community shall increase the commitment to the project. Introducing a governing structure that pays respect to both sides can be considered a main task for such open source communities [West and O'Mahony, 2005].

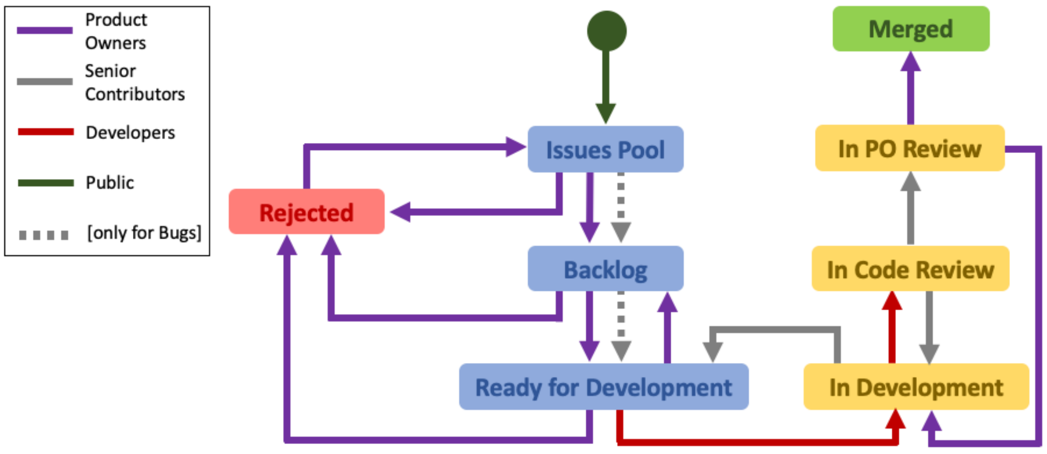


Figure 7.4.: Catrobat's development workflow considering product owner interactions (Figure as published in [Müller et al., 2019b] - Reprinted/adapted by permission from Springer Nature Customer Service Centre GmbH: Springer Nature Introducing Agile Product Owners in a FLOSS Project by Matthias Müller, Christian Schindler, Wolfgang Slany ©2019).

Development Workflow

To introduce this new role, a development process, as illustrated in Figure 7.4, needed to be introduced. Catrobat's development is managed through a issue tracking system, which is open for all interested contributors. Also non-programmers are enabled by such issue tracking systems to report bugs and feature requests, which reflect the ideas of the users [Heppler et al., 2016]. To prevent duplicate work and ensure the quality standards are met also external contributors are invited to work with this system. Besides that, this workflow is intended to allow frequent and fast releases, which is a challenge for many community driven projects [West and O'Mahony, 2005]. Therefore, product owners in this workflow have three major tasks:

- Defining and prioritizing requirements and issues for the developers. Whereas, external contributors are not necessarily required to follow these predefined issues (e.g. work freely on ideas), participating students have to choose issues provided in *Ready for development*. However, also common contributors are asked to communicate their ideas to avoid rejection in the acceptance phase. Therefore, the creation of new issues is open for the public, also allowing to consider these ideas in the workflow, avoiding the mentioned rejection afterwards and foster involvement of users and the community.

- Discussing proposed requirements in planning games to clearly communicate the objectives and to get the developers' commitment .
- Functional acceptance of issues and merging them into the main repository. This step is necessary for all contributors to ensure the quality and prevent unfinished, buggy or inappropriate work being published.

The transition of issues from *Backlog* to *Ready for development* happens in a joint planning game. In this, the issues are estimated and developers assure a joint understanding of them. Whereas product owners have the key role in the first and last phase of the process, development is managed entirely by the developers. This includes that issues are not preassigned to contributors, reducing dependencies on certain individuals. Therefore, discussing proposed requirements with the developers and getting their commitment is essential for this agile workflow. Developers are also asked to review the work of others if it complies to the project's quality standards. This shall strengthen the collective code ownership in this process. An exemption in the workflow exists for bugs. They can be claimed by developers at any time without the involvement of product owners. Although, final product owner approval must be granted just like with scheduled issues. This supports the benefit of open source software projects - that bugs can be found, fixed and released quickly.

Communication

An important requirement for the introduction of product owners has been strengthening communication within the project. This step also focused on encouraging exchange between contributors, e.g., jointly discussing development tasks. Regular on-site meetings with student-contributors get reinforced and Slack got introduced as main communication platform for all contributors, stimulating discussion in different topic specific channels. This is also intended to document decisions and information for currently absent contributors that might be involved in the following implementation phase. Beside the communication within the team, also product owners need regular meetings and exchange about the project's status. Therefore, following activities have been put into focus:

- Weekly Get-Together of to discuss upcoming features and requirements. This also includes backlog refinement and obtaining feedback from contributors.
- Monthly Planning-Games of product owners with the development team to schedule issues for *Ready for development*. During this event also the specification of issues is discussed, e.g., if developers need further clarifications, or if they are blocked by each other. Product owners hand over the prioritized issues for the next month to the contributors.
- Continuous exchange about current issues on designated Slack-channels and on an individual basis, e.g., via e-mail or comments on Jira and GitHub.

All this is intended to be open and transparent, what is essential for successfully governing FLOSS projects [Lerner and Tirole, 2002]. A challenge is the efficient communication between product owners. Especially for the case when one product owner answers questions from contributors, it is important that all others are informed about decisions made in their absence. This makes documentation and communication essential for this multi-person product owner approach.

7.2.2. Discussion

Although this workflow has just been introduced in early 2018, positive feedback is received from contributors. A challenge identified is to centralize communication that got essential for the collaborative nature of the workflow. Whereas contributors before exchanged in a way preferable for them, they must now be streamlined on dedicated channels. This is also true for the introduced product owners. An increasing number of messages and online time by contributors is outlining a preferable progress to tackle this challenge. Since this work solely points out the experience of this specific case, further long-term research on this approach is needed to be able to evaluate its success.

7.2.3. Conclusion

This work provides an approach for sponsored open source projects to balance the involved parties' needs and the freedom of contributors in an agile way. A simple workflow with clear responsibilities is provided that might be a response to the growing involvement of businesses or public institutions in open source communities. The introduced role of product owners can be seen as an extension to the already often defined role of leaders governing a FLOSS project. However, by following the proposed workflow and role definition based on Scrum, collaboration and communication in this open setting can be fostered, by ensuring the development of required business objectives at the same time.

7.3. Managing Value within Catrobat[11]

Today, software gets created by diverse teams in complex ecosystems of various involved actors. Their needs and requirements must be understood and considered in all development phases to ensure success in the long run. This is especially true in Free Open Source Software (FOSS) projects, in which different stakeholders contribute in various ways for a variety of reasons. In contrast to commercial projects, FOSS projects not restrict contribution, opening it to all persons who are motivated and have the necessary skills for contributing [Hippel and Krogh, 2003]. These, often non-profit, projects must align the diverse stakeholders towards a shared direction, ideally to create value for all involved actors. To encounter the

[11]This section is published in [Müller et al., 2019d] ©2019 Copyright held by the owner/author(s). Publication rights licensed to ACM. DOI: 10.1145/3344948.3344976

traditional approach of value-neutral software engineering, an agenda, including beyond others the areas of requirements engineering, design, planning, and development, got introduced towards value-based software engineering [Boehm, 2003]. Especially in open and innovative communities, it is essential to create a direct or indirect net-benefit for involved actors (i.e., contributors), defined that the value they receive outweighs the costs (i.e., effort or time) [Lerner and Tirole, 2002]. But, this gets in particular challenging if the value is intangible and hard to measure. Furthermore, in FOSS projects many contributors are just peripherally involved, not having knowledge about the whole ecosystem. Consequently, it gets hard to set up a structure and processes that support this (intangible) value driven approach of software development. Besides the created value for the actors, this collaborative process is also directly influencing the software's quality, maintainability, and architecture, impacting a project's sustainability and long-term development.

To outline the challenges and dependencies that can occur in such a setting, we present the case of Catrobat. First, we provide some introduction to the project. Second, we review the theoretical requirements that directly but often unnoticedly influence such an open project. Third, we highlight the impact of a diverse ecosystem on software design and development, directly influencing requirements engineering, the software's architecture, quality, and collaboration. We therefore try to create a holistic picture, capturing the viewpoints of involved actors in a FOSS project. We line out how these factors are interconnected to each other in practice and that value creation in such a diverse project can be challenging to manage and direct into a common direction, ensuring that the needs of all involved parties are considered.

7.3.1. The Case of Catrobat

Catrobat is a non-profit open source project initiated at Graz University of Technology in 2010. It is aiming to develop free mobile visual coding frameworks that foster computational thinking and provide a low entrance barrier for beginners. The currently most successful service provided by the project is the freely available Android app *Pocket Code*, that already attracted more than 700,000 users. Whereas the project is primarily driven by students, also an increasing number of external contributors and partners, including other universities, organizations, and companies, are engaging in it. Within the past years, several hundred persons have been involved in it for very individual reasons, contributing different kind of value, e.g., development effort, support for users, or translations. The project over the years constantly grew, requiring regular adaptations of the processes to pay respect to all involved actors. Especially the fact that Catrobat has been established at a university shaped the project and its contributors. Students can participate on a voluntary basis as part of their curriculum. Therefore, they engage for various reasons, such as a personal interest on the project's mission, gaining new knowledge and skills, or creating a reference for their future career [Müller et al., 2019a]. Thus, also the values expected for their contribution differ, resulting in a very diverse community.

7.3.2. Requirements on non-profit Open Source projects

FOSS projects are situated in complex ecosystems of different actors, resulting in the need for a structure, management process, and also shared understanding of the system for all involved parties. The created values are primarily intangible, making them hard to understand and communicate.

How to define Value?

It is essential to know and align the software development process and management structure to meet the needs of all participating actors to create value for each of them. Contributing and consuming actors of a FOSS ecosystem need to be seen in a socio-technical system context [Trist, 1981, Mumford, 2006, Baxter and Sommerville, 2011], where each actor operates in an organizational context, that is interrelated with a surrounding external environment [Vorraber et al., 2019b]. The willingness and motivation to contribute is influenced by intrinsic motivation and exogenous influences and is determined by the personal expected and received outcome [Porter and Lawler, 1968, Vroom, 1964, Vorraber and Vössner, 2011]. Types of needs and therefore sources of motivation can be diverse and overlapping ranging from general and basic needs as described by Max-Neef [Max-Neef et al., 1991] (e.g., creation) to monetary, social, process or task specific needs [Vorraber et al., 2019b] driven by internal motivation or external influence. Motivating factors for contributors to the Catrobat project could be course credits when contributing as part of their studies or gaining knowledge [Müller et al., 2019a]. Especially, the often not so obvious intangible values of users and contributors have to be carefully considered when arranging and managing a FOSS ecosystem. Examples from past ICT-based innovation projects show that unmet values of users may be a show stopper for large projects. An example thereof is the failed electronic patient record system project in the Netherlands, which was abandoned in 2011 [von Schomberg, 2013]. In this project, the management failed to consider the intangible value "privacy" of the end users properly, resulting in a €300 million loss.

Understanding Contribution in and Organization of FOSS Projects

The different motivations and the willingness to contribute is also related to the role contributors take in FOSS projects. They are usually backed by a diverse community of different contributors, having different backgrounds and kind of contribution. These individuals differ in aspects, such as contribution-frequency, kind of contribution, time spent on the project, or contact to the project's community, resulting in an individual structure for every project [Nakakoji et al., 2002]. Consequently, the definition of an open source community can be broad, including many actors that are either directly or indirectly influencing, but also benefiting from, a project. Therefore, managing, or respectively leading such a project based on trust requires knowledge about the involved parties and their expectations to be successful in the long run. Therewith connected is also the fact that the benefit created for the contributors is often ambiguous and not directly expressed. This makes it in particular hard to understand the value expected and created through contribution, since it is often intangible on a social level and not obvious for others. Therefore, open source projects which rely on open communities that are connected to other organizations as well, represent highly complex social ecosystems. If managed the right way, these communities can be key to success. Otherwise, if not all required actors are considered accordingly, they might be the cause for a project's collapse.

Consequently, having a leading role providing a shared direction and vision is common for such projects [Nakakoji et al., 2002]. We can also observe a large number of legal entities, i.e., foundations or associations, forming the official structure for these projects. Whereas in the beginnings, FOSS projects have often been driven by informal communities, today they are in most cases well organized. We identify several benefits of this approach. First, contribution and participation can get formalized, allowing to define these roles and including their viewpoints in governing the project. This also opens the projects to build alliances and actively involve interested (external) advisors, resulting in strengthening the vision of the individual projects and basing them on solid ground. Second, marketing and public relations can get performed under one unique umbrella, helping to raise awareness and reach out to the public. Within this step, also awareness of the targeted user group and potential use-cases can be raised within the project, helping to better understand and meet the needs of the passive users, that are not necessarily involved in the development process. Third, the non-profit character of these projects can get legally underpinned, helping to ensure to follow social values in an sustainable and long-lasting way. Therewith, trust can be built up within involved groups, also supporting to gain funding to reach the projects' higher goals.

7.3.3. Challenges in Practice

On the case of Catrobat, we line out challenges and issues that arise in such a project. The presented case is in a current stage of change, which directly stresses value related challenges on different related levels. Although recently new processes and principles got introduced, existing issues from the past still negatively impact the project in various ways and need to be considered.

7.3.4. Setting up an organizational Structure

Catrobat made efforts to build an organizational structure that allows contributors and partners to get actively involved in it. Besides the project's structure that has already existed before, an association got founded. This association enables external contributors and supporters to get officially engaged in the project. Although this legal entity builds the frame for contribution, defining roles and responsibilities is rather difficult as experience showed. In earlier times so called external (peripheral) contributors, members, and senior contributors have been distinguished, each related to certain responsibilities and rights. Nevertheless, these role definitions became ambiguous due to a multitude of needed special cases and exceptions. In comparison to industry settings, roles of contributing parties are rather vague and might change frequently. Furthermore, each contributing actor is creating, expecting, and receiving different kinds of value, resulting in the aforementioned net-benefit. Therefore, predefined responsibilities rather seemed to hamper the collaborative development process, resulting in missing responsibility for and relation to the overall project. But, on the other side there is also the need of a structure to ensure that for every part of the project is taken care of. The following section lines out how especially for the management of requirements and their prioritization, missing responsibilities might negatively impact a collaborative software project.

Agile Management

In 2018 Catrobat introduced an agile workflow, aiming to define the already existing leader role as Product Owners (POs) who balance the needs of the different involved parties [Müller et al., 2019b]. As already outlined, contributors follow various motivations to contribute. Same is true for other involved entities, such as businesses or other universities. They either use the developed services themselves, want to support the project's mission of fostering computational thinking, or have other reasons to support Catrobat. Whereas this new workflow is showing first positive results, also challenges come to light. Before this new workflow has been introduced, many contributors more or less defined their work themselves. In the end, many stories proposed in the project's ticketing system by external actors, e.g., users or partners, have gone unnoticed. Tickets by the contributors that create value for themselves, e.g., because they have a personal interest in it or it is fun to implement, have been favored. Furthermore, these tickets often have not been developed in relation to each other, resulting in techt debt. In

November 2017 it became obvious that specific actions are needed, since the project's quality decreased and the complexity of the architecture increased, so that certain trivial bugs became almost impossible to get fixed. At this point also the number of proposed but untouched tickets got unmanageable (i.e., more than 400 proposed tickets by contributors, users, and partners that have not been reviewed in certain cases even for years). In addition to that, refactoring and quality related tasks often got postponed in favor of new features, increasing the mentioned issues with code quality and architecture.

Since Product Owners now take the responsibility to select, prioritize and communicate the next development steps, the new introduced agile workflow prevents proposed tickets, independently by whom they get created, to get unnoticed. Nevertheless, as the last months show, this new management instance alone can not immediately solve the existing issues of bad software quality and architecture, that in most parts can be blamed on unprioritized work and contribution that has been driven by personal values instead of being based on a holistic view on all involved actors. As described below, issues that arose through unmanaged value driven software development also require explicit development and communication actions in addition to management efforts. Lacking to manage to regard all actors' expectations on the software in the past, now leads to the demand of heavy refactoring. As an example, setting up a proper software architecture that enables improvements of the User Experience (UX), including changes to the interface and requested features, is already in progress for several months.

Collaborative Software Development

Developing open source software is a team effort, relying on the contribution of the different developers. Already on from its beginning, Catrobat has been following agile principles from XP. A CI framework and TDD (test driven development) approach support developers if used correctly. However, an issue arises in the matter of collective code ownership, as proposed by XP, which defines that every developer may change every part of the code at any time [Beck and Gamma, 2000]. Collective code ownership as a policy is not as efficient as having it as best practice applied by the developers, since they otherwise may be allowed to change all code, but do not see it as their responsibility [Sedano et al., 2016]. This is also a problem for Catrobat that got reinforced through refactoring that got neglected over a long time and by many developers. Contributors focused solely on the task they have personally been working on, instead of having a bigger picture in mind. Furthermore, specific contributors solely worked on individual features, resulting in knowledge silos and lack of code ownership. Such silos result in the fact that other contributors have no knowledge of these parts and although they would be allowed to are not able to change them [Sedano et al., 2016]. Ultimately, this leads to a pro-forma collaborative code ownership, which is allowed by policy, but not brought into practice. This results in a just weak responsibility of the contributors for the code and project.

This lack of collective code ownership also impacts the software quality and architecture, which not got regarded sufficiently in the past. This is amplified by a lack of documentation. Most FOSS projects just rarely follow common standards for documentation of architecture and in turn rely on informal documentation represented through natural language, i.e., in communication [Ding et al., 2014]. This can also be seen at Catrobat. The architecture of the code base got complex also for the lack of responsibility felt by the developers. Whereas this already hampers contribution, it gets even more complicated without proper documentation. Since contributors focused primarily on their own issues, they did not document their work and architectural decisions. Information is transferred often informally via direct communication between contributors. Thus, a general documentation of design decisions and architecture is currently missing. This potentially makes things even worse, since this leads to an even bigger lack of taking ownership, reinforcing the issue [Sedano et al., 2016]. The missing documentation also directly impacts new contributors willing to participate in the project, since they rely on communication instead of documentation.

Communication Challenges

Investigating multiple contributions within the scope of the Google Summer of Code program at Catrobat revealed that there are different approaches to make the first step towards development. Most contributors first try to get in touch with the community before starting development, while only a few directly make submissions to issues they found on the public ticketing system. Developers with a proper professional experience are able to make contributions to technical issues in a fully autonomous way, however there is a tendency of failure when it comes to domain-related concerns (e.g., features or UX improvements), requiring a large amount of communication with the rest of the community. Different to other FOSS projects, where software is developed by the same people who use it [Hippel and Krogh, 2003], this is not the case for Catrobat whose target users are teenagers. Typically, a developer is interested in both a high-quality and an innovation-driven contribution, motivated by the incentives to gain additional value as an end-user. Having different values expected by developers and users, as it is the case at Catrobat, a lack of domain knowledge and end-user experience has been observed. This makes it hard for newcomers to start contributing to complex issues requiring a general understanding of the overall picture. As a result the need for documentation and specification has been regularly addressed. While co-located contributors of this agile FOSS project are profiting from regular planning games, where domain-related requirements are discussed prior to development, there is a compelling need to clarify (functional) requirements and pre-action plans with external newcomers. Additionally, contributors often complain about encountering unexpected technical hurdles arising from (a) bad code quality, (b) complex and outdated code; and, (c) high interdependencies and low modularity. While all these issues push open-source developers away, inexperienced contributors strongly depend on interactions with the community. Since development can not be continued without further guidance, it is important that appropriate support is received in a timely manner, a crucial challenge FOSS projects have to deal with in order to retain a constant flow of newcomers [Steinmacher et al., 2015]. Experience has shown that issues related to domain knowledge, technical affairs and lacking awareness can be overcome with a large amount of frequent and direct communication, stressing the necessity of socio-technical interactions, i.e., regular and open communication, in FOSS projects.

7.3.5. Conclusion

The illustrated case of Catrobat highlights the different effects that occur through the socio-technical system context of open source projects. Various actors, i.e., contributors, stakeholders, or users, are jointly involved in the software creation process. Each of them for individual reasons and most importantly, each of them is creating and expecting very different values in return. Thus, all human related aspects of the project, in particular organization, management, collaboration, and communication, are interrelated and directly impacting the outcome in terms of provided user experience, quality, and architecture. Missing to pay respect to the social aspects can easily result in long lasting impacts on the code-base. Even with frequently changing and refactoring them, enormous resources are needed to maintain an appropriate status. More importantly, it even threatens the health of the system if not every actors' value perspective is regarded. In the case of Catrobat, still extensive refactoring in the means of defining a proper software architecture and improving the existing code-base is needed to ease the development of new features and to fix known bugs. This is related to the experience and value the developers are gaining during their contribution. Until that is accomplished, also creating value for the users and involved stakeholders is hindered, in the worst case even frustrating them. For the majority, these circumstances can be blamed to a missing streamlining of the different actors, that prioritizes and manages the arising issues in accordance to the socio-technical context of the project and not just from a single viewpoint, e.g., from the users or developers. However, as the case shows, introducing this holistic streamlining potentially helps to prevent such situations in the future, but even with such a process, a lot of effort is required to fix consequences that result from a missing streamlining of value in the past.

7.4. Implications for Catrobat after the introduction of the workflow[12]

7.4.1. Introduction

Software engineering students must be equipped with a large number of theoretical, practical and social skills when starting to work in the industry. While literature suggests that theoretical knowledge (i.e., learning by studying) should complement practical experience (i.e., learning by doing), it can be difficult to imitate practical settings in an educational context at university [Ghezzi and Mandrioli, 2005]. Providing a practice-oriented and industry-relevant environment, in terms of realistic software projects, for computer science and software engineering students at universities becomes more and more important. Teaching software engineering through contribution to open source projects has already proven as a beneficial approach that provides a positive and enjoyable experience for educators, students, and the involved open source communities [Tafliovich et al., 2019, Pinto et al., 2019]. Furthermore, it provides the chance to transfer skills and competences in global software engineering. Students in such a setting also gain experience in communication, discipline knowledge and project-oriented work, similar to professional environments [Tafliovich et al., 2019]. Nevertheless, one of the major challenges identified in research (e.g., [Pinto et al., 2017]) is to find an appropriate project students can contribute to. Although different approaches with a varying amount of freedom for this selection are commonly used in educational settings [Pinto et al., 2019], the presented case describes an environment where students are exclusively enabled to contribute to one specific project, which is managed by university staff and provides a known and clear surrounding for the participating students.

In this work, we provide lessons learned from an agile educational open source project directly established at university. The *Catrobat* project, founded at Graz University of Technology in 2010, provides an educational setting in which students can participate in an open source project as part of their

[12]Based on unpublished working-paper by Michael Herold, Matthias Müller, Christian Schindler, Vesna Krnjic and Wolfgang Slany

studies [Müller et al., 2019a]. Whereas the educational setting has already been discussed in the past, this work highlights its practical processes and approaches to enable this unique environment. New data is provided to emphasize the individual aspects of this approach and to highlight implications for the project managers in charge. New results presented as a case study in the following sections give insights into the success prospects of this setting and provide a rough proposal on how open source projects can be introduced and managed at university for educational purposes. A special focus is put on the students' perspective and on how to design processes and activities to both ease their contribution and to provide a close-to-practice teaching environment that offers essential preparation for the professional world after graduation and thus also can be beneficial for their future careers.

7.4.2. Methodology

This sections intends to outline the practical lessons learned from running an educational agile open source project at university. Thus, it is supposed to provide a real-world realistic case that provides insights into this specific domain and may encourage further similar cases in educational environments. Based on the classification of Stol and Fitzgerald [Stol and Fitzgerald, 2018], this work is following a *field study* strategy in a natural setting. As common practice [Yin, 2017], the applied single-case study methodology is underpinned by empirical data originating from surveys and an analysis of the resources employed within the case (i.e., Slack and git). The empirical data emerges from an initial survey with 58 participants in May 2018 that was used to identify issues and potential room for improvement. Based on those results a continuous survey was conducted with 67 students who started contributing to Catrobat between July 2018 and August 2019 (i.e., students filled out this survey before their first contribution), as well as a final survey about the students' contribution conducted in August 2019 (submitted by 31 students). Additional data to underpin those results and to put them into context is based on an analysis of Slack, git and expert interviews within the case.

This work aims to shed some light on the following aspects:

- How a close-to-practice environment for software development courses at university can be established
- How industry-near processes at an educational open source project can be simulated
- How students benefit from contributing to a real-world open source project
- The challenges that students can be facing during their (first) contributions

The overall section is organized in the means of the IMRaD format, providing a clear and common structure for scientific articles [Nair and Nair, 2014]. After we have described the goal of our work and methodology, we will present the results and outcomes of the case study in Section 7.4.3. Thereafter, we will outline how students can participate in practice and highlight potential opportunities and benefits of this contribution approach. In Section 7.4.5 we illustrate how students jointly develop open source software and how the project's long-term development is ensured. In the following Section 7.4.6, we then describe the case in the context of global software engineering and present ways of creating an international community for the project. Last we discuss the presented outcomes and conclude on its implications for research.

7.4.3. The Catrobat Project

Catrobat follows the charitable vision of fostering computational thinking by enabling teenagers to create their own mobile apps directly on their smartphones in an easy-to-use visual way. By sticking together graphical blocks, complex apps and games can be created without the need for any previous knowledge in programming. This approach has the benefit that it (a) motivates students to learn the concepts of programming, (b) provides meaningful activity for teenagers in their leisure time; and, (c) gives them the chance to gain important skills for a possible later career in ICT. Catrobat is organized as a non-profit open source project, established at Graz University in 2010, and has already been described in various scientific articles (i.e., [Slany, 2012, Slany, 2014b, Müller et al., 2019b, Schranz et al., 2019]).

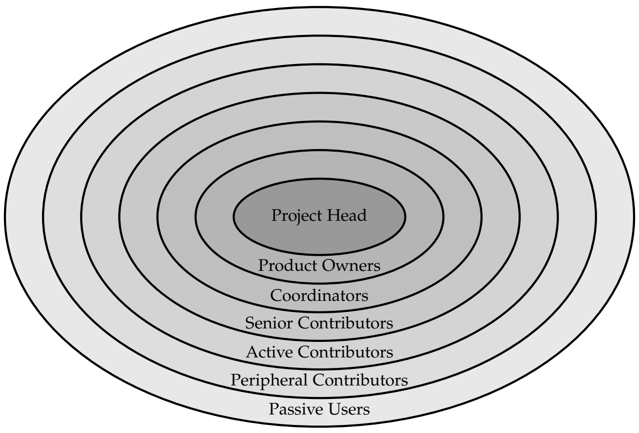


Figure 7.5.: Roles in the Catrobat community represented in the contributor's onion model as described by [Ye and Kishida, 2003] (adapted from [Herold, 2019])

The Organization

As illustrated in Figure 7.5, the project is organized in an onion-like structure that is common for open source projects [Ye and Kishida, 2003]. Whereas the project-lead in the center is fulfilled by the project's founder, all other roles within the project can also be taken by contributors and students, depending on the time they spend with the project and the responsibility they are willing to take. This is also supported by offering students the possibility of a long-term contribution as part of their study program [Müller et al., 2019a]. In addition to development-related competences, this also allows contributors to gain experience and skills in various other fields (i.e., mentoring others, managing parts of the project, working in teams, marketing, graphic design, usability, etc.).

7.4.4. The Contributors

Although the contribution is open for everyone and different actions are taken to attract more external developers (e.g., see Section 7.4.6), the majority of the developers have some connection to Graz University of Technology. Since the project has been founded, more than 300 students have contributed to Catrobat both on a voluntary basis and as part of their studies [Müller et al., 2019a]. Whereas the data provided in previous work about the project (e.g., by Müller et al [Müller et al., 2019a]) solely represents a certain point in time, the survey conducted during this research covers 14 months, thus providing a continuous perspective about new students that are willing to contribute to this open source project during their studies. The educational aim is to let students gain experience in a practical environment, near to real-world industry conditions, but without any pressure and consequences for failure (i.e., motivating them to try things out). The large majority of students (94.0%) start their contribution already during their Bachelor's program, hence, at an early stage of their studies. This also leads to a low amount of practical experience in software engineering, which is intended to get increased during their contribution. As illustrated in Figure 7.6, 67.2% of the students have not worked in any professional setting before. Only 7.5% have more than 3 years of experience in software engineering. Consequently, students also rate their software engineering skills rather moderately, similar to a Gauss distribution, on a scale between 1 (beginner) and 5 (expert). Although only 13.4% of these students have contributed to any open source projects before, students are at least aware of this specific project, since 71.6% of the surveyed students have used any of Catrobat's services prior to joining the project. From a practical software educational viewpoint, one third (31.3%) applied agile methods in practice and another third (35.8%) at least has heard of them in theory. Since Catrobat follows agile principles (i.e., Clean Code, Test Driven Development, etc.), this eases the contribution and lowers the barrier to contribute on a theoretical level. Nevertheless, the entrance requirements for students are kept to a minimum, i.e., limited to basic programming knowledge in Java, so that the contribution as part of their studies is enabled independently of the students' current progress in their study program. However, this consequently results in a steep learning curve in the beginning if skills are not that advanced. This requires a high degree of motivation for the students since this progress can be time-consuming and also disappointing. Especially a comparatively weak architecture of the code base and other quality-related issues may hinder a contribution and requires additional effort by beginners, which not all are willing to take [Schranz et al., 2019].

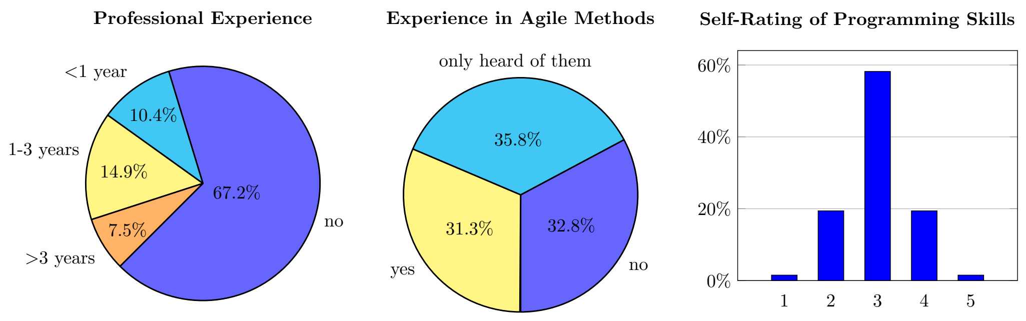


Figure 7.6.: Skills when students start contributing to the Catrobat project

7.4.5. Contributing to the Project

The hybrid open source nature of the Catrobat project offers a unique setting where students can collaborate with external experts by further using a toolset frequently employed in the industry. Thus, contributing to Catrobat and other open source projects provides essential preparation for the professional world after graduation [Tafliovich et al., 2019]. As described by [Ye and Kishida, 2003], a contribution can be versatile ranging from a simple bug report to complex long-term contributions (i.e., the implementation of new features), which is also represented by the different roles described in Section 7.4.3. To ensure the long-term success, Catrobat, like many other projects, strongly relies on the constant acquisition of newcomers who eventually transform into more mature senior roles [Jensen et al., 2011, Steinmacher et al., 2012]. Since the majority of Catrobat's contributors are university students participating as part of their studies, there is a high fluctuation within the project, resulting in an average contribution time of only one year or less [Müller et al., 2019a]. The controversy of the aforementioned steep learning curve and the high fluctuation constitutes several challenges as described in the remainder of this section.

Development Process

Catrobat is following the *copy-modify-merge* development model as suggested by Fogel and Bar [Fogel and Bar, 1999]. Contributors are allowed to fork the source code on GitHub, make changes to the local code base and submit their changes using a pull request. Before a pull request can become part of the main code repository, these changes need to be reviewed by the community. At Catrobat a two-way code review process, inspired by Scrum, has been introduced in 2018 [Müller et al., 2019b] which involves a technical code review by a contributor and a functional code review by a Product Owner (PO). This new agile process helps to ensure high-quality software from both the technical point of view and the users' perspective. Due to the aforementioned knowledge of the students in agile methodologies, this also gives them the opportunity to apply a Scrum-like process in practice. Especially quality has been highlighted by the vast majority of contributors (77.6%) in the first 2018 survey as a domain that needed to be improved. According to the questionnaire conducted in August 2019, the introduction of code reviews has also been perceived positively by most of the contributors (96.9%) who either agree or strongly agree that code reviews usually lead to a higher source code quality. A majority of the contributors surveyed (56.3%) either agree or strongly agree that the supplementary PO review helps to further ensure high quality. Besides these technical benefits, code reviews offer a valuable technique for a bidirectional knowledge transfer and to increase team awareness among the students (e.g., [Bacchelli and Bird, 2013], [Cockburn and Williams, 2001]). Since these reviews usually represent the first contact between a contributor and the community, it is important to streamline this process because otherwise newcomers may get discouraged and stop contributing. Experience has shown that code reviews should be performed as fast as possible and, if possible, in person or at least on a synchronous basis. By doing so misunderstandings and ambiguities are reduced and knowledge transfer is promoted. It is shown that long-lasting code reviews and a large number of change requests may lead to demotivation among the contributors. In addition to code reviews, the project focuses on test-driven development (TDD) and regression tests in order to ensure high code quality and to reduce the overall bug rate, as suggested by [Beck and Gamma, 2000]. A feature or a bug is usually defined in the form of a ticket in the project-wide issue tracking system Jira[13]. Students can freely pick one of the tickets marked as *Ready for Development* and initiate the development process. At the beginning of every release local *Planning Games* are held were POs and

[13]an issue tracking tool provided by Atlassian, https://jira.catrob.at

student developers meet to mutually agree on the scope of the next release by jointly estimating the development effort of a ticket. This constitutes an essential step in Catrobat's agile workflow which helps to effectively communicate the current goals and which provides a place where students can clarify domain- and ticket-related questions, helping to create a common understanding of the planned tickets. This is in line with the outcomes of the survey which illustrates that 84.4% of the students either agree or strongly agree that *Planning Games* are highly beneficial to get a better understanding of the tickets in development.

Onboarding

Besides the required technical experience, newcomers need to have profound domain expertise as well as comprehensive knowledge about the project's guidelines, rules, and standards. These prerequisites offer potential hurdles for students which makes it particularly challenging for Catrobat's educational setting involving a large number of diverse technical seniority among the students. This is also backed by the recent numbers, which show that 48.1% of the contributors see room for improvement at the onboarding process. This may also be seen in relation to the aforementioned varying background (professional expertise and programming skills) that influence this process. Hence, in order to enable a swift start into the project, it is necessary to provide proper guidance for all parties involved. Expert interviews with long-term contributors and the thorough analysis of multiple contributions within the scope of the Google Summer of Code program (see Section 7.4.6) demonstrate that there is an urgent need for community-driven guidance at Catrobat. Contributors are likely to have either (a) a lacking programming expertise, (b) a lack of domain knowledge; or, (c) an insufficient awareness of the project's strict guidelines (e.g., about software testing). This is in accordance with the results of the survey which shows that 58.3% of the newcomers find it struggling or even hard to start writing their first piece of code. To counteract these hurdles, the expert interviews reveal that frequent and direct communication, as well as mentoring, pair programming and synchronous code reviews, have a positive effect on a newcomer's contribution. Based on our results, this can be assumed to be true for all newcomers, independently of their background and previous experience.

Communication

Although the Graz University of Technology offers a dedicated room for all contributors, most students prefer to work remotely [Müller et al., 2019a]. Hence, the major part of the communication takes place on web-based services. Catrobat provides various tools to enable efficient inter- and intra-team communication. Among other things, Slack has been introduced in 2017 replacing the project-wide IRC channel which seemed old-fashioned to the contributors, was rarely used and thus turned out to be no practicable solution for the long run [Fellhofer et al., 2015]. As a consequence of the initial 2018 survey, in which 34.5% saw room for improvement in inter-project communication, the newly introduced workflow put a focus on exchange and discussion in all development processes (i.e., also remotely via Slack), trying to provide an open environment. The conducted end-survey reveals that a majority of the contributors (90.6%) experience Slack as a valuable tool to enhance inter-team communication that helps the project. Furthermore, a retrospective with several of the most active student-contributors in October 2019 highlighted the positive aspects of this open setting. All of them perceived the current way of communication as a positive factor for the project. Thus, having an open culture of communication fostered through well-established tools from industry (e.g., Slack) is positively perceived by the participating students and can, therefore, be seen as a key success factor for such a project. Additionally, Jira is used for ticket-related conversions and GitHub is employed to communicate code-related issues. Nevertheless, 87.5% of the surveyed student contributors still highlight the dedicated on-site working room at the university as beneficial, fostering direct collaboration and interpersonal exchange. From a technical point of view, interviews conducted with senior contributors within the Catrobat community show that there is almost no difference between the remote and co-located collaboration if the proper tools are available (e.g., Slack). However, the expert interviews reveal that face-to-face meetings involve several advantages in comparison to remote collaboration, among other things because "interpersonal matters helped to break down social barriers". From an educational perspective, both online and offline possibilities for communication must be provided to create the biggest output for the students. In addition, to offer students the possibility to get familiar with tools frequently employed in the business sphere, this hybrid setting also enables students to collaborate in person (e.g., in order to perform pair programming) but also to contribute independently of their location (easing contributing remotely).

Long-term Contribution

While most students contribute as part of their studies earning credits for courses that are part of their curricula, there are students who continue to participate after these courses have finished, contributing to Catrobat for more than six years [Müller et al., 2019a]. During these years, long-term contributors may transform into different roles towards the core of the onion model [Ye and Kishida, 2003] as illustrated in Section 7.4.3. By doing so, they benefit from having more privileges and greater involvement in project decisions. Additionally, students gain a deep understanding of practical areas related to their studies as well as a profound experience in the collaborative software engineering and management process. Thus, long-term contributions can have a benefit for the students' further careers [Von Krogh et al., 2012]. This was also shown in previous studies on Catrobat (e.g. [Müller et al., 2019a]), which depicts that 36% of the students see their contributions as a reference for their future career path.

7.4.6. The International Perspective

As of May 2019, the main code repository[14] consists of commits from 203 different developers. Having a closer look at the contributors, Figure 7.7 illustrates that more than 80% were contributing from Austria. Nevertheless, there are various contributions from all over the world, among others arising through the participation in various coding engagements initiatives

[14]https://github.com/Catrobat/Catroid

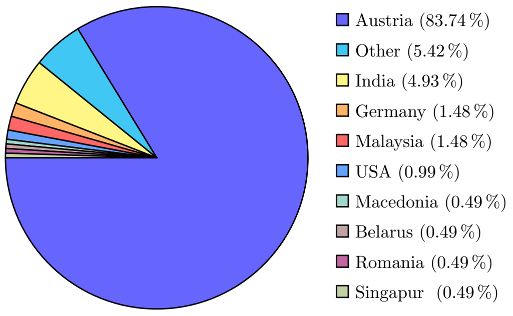


Figure 7.7.: Contributors of the *Catroid* repository grouped by country (rounded to two decimals) (adapted from [Herold, 2019])

(see Section 7.4.6) and several strategic cooperations (see Section 7.4.6). This emphasizes the importance of this unique educational setting where students have the opportunity to collaborate with international experts from all around the globe. The data was manually collected from the *develop* branch of the main code repository *Catroid* and assigned to countries by evaluating the contributors' profile page on GitHub and by looking up the contributors' email addresses, GitHub handles and full names in Catrobat's internal Confluence[15] page. Contributors who could not be allocated to a specific country were assigned to the country named *Other*.

Open Source Coding Initiatives

Catrobat is participating in various coding initiatives which all have the main goal to attract new students from all over the world and introduce them to the world of open source development. Among others, Catrobat has been part of the Google Summer of Code (GSoC) program since 2014 which represents the largest initiative. Founded in 2005 by Google, this annual event wants to inspire young students and provide them the opportunity to do study-related work during the summer, thus *"flip bits, not burgers"* as stated on their website[16]. During this three-month program, students from accredited universities are paired with at least one mentor and collectively contribute to the project (e.g., by the implementation of new features).

[15]A collaboration software published by Atlassian
[16]https://summerofcode.withgoogle.com

Similar to that, Catrobat took place in the Google Code-in program whose main purpose is to attract pre-university students with the age of 13-17 to become part of an open source community. By doing so, participating students profit from working together with field experts, getting used to tools employed at industry and among other things have the opportunity to win prizes (Google Code-in) or earn a stipend (GSoC). On the other hand, this program helps open source communities to find and retain newcomers who potentially become long-term contributors. While a study [Trainer et al., 2014] found out that around 18% of the GSoC students are applying as mentors in subsequent years, other research [Silva et al., 2017] claims that around 64% of the students stop interacting with the community one month after the program has finished. Furthermore, local students can benefit from these initiatives since they can participate as mentors and thus gain valuable experience in the areas of mentoring, performing code reviews, onboarding, compiling meaningful milestones throughout the coding period and planning to submit deliverables on time. Finally, these initiatives may have a positive influence on the students' social skills since they are encouraged to collaborate with contributors from many different cultures.

Scientific Cooperations

Due to the project's relation to the university, various cooperations with partners from academia are enabled and fostered by the project's leaders. Besides just working with other students from Graz University of Technology or individual external contributors, efforts are made to establish teams at other institutions, also jointly contributing to the Catrobat project. One of these efforts is made together with the *Code The Change*[17] team at Stanford University where students are joining forces to contribute to various non-profit projects during their academic year. This team is self-organized in a club, aiming to do something for the social good[18]. Cooperation with Catrobat has been kicked off in the academic year 2018, in which students from this team at Stanford agreed to work on specific features for Catrobat.

[17]http://codethechange.stanford.edu

[18]https://haas.stanford.edu/students/cardinal-commitment/code-change-0

Therefore, students from Graz traveled to Stanford, to provide them a basic introduction to the code base and processes of the project. Furthermore, they cooperate remotely on the aforementioned Slack channels, to work together independently of place and time in a global software engineering context. This again provides a valuable opportunity for students to gain experience in a distributed software engineering team. Further cooperation on the university level is enabled through a strategic partnership between Peter the Great St. Petersburg Polytechnic University and Graz University of Technology. Within the scope of *Project Marathons*, students can jointly work on tasks related to this open source project. Funded by the universities, participating students get supervised by scientific staff on both sides, working together on a given problem. This allows students from both sides to gain competences in global software engineering and coordinate their joint contribution remotely.

7.4.7. Discussion

Instead of trying to generalize the outcomes of this section to a wider population (statistical generalization [Yin, 2017]), research has been conducted to offer insights into an educational agile open source project at university. The observations are solely based on one single case (*Catrobat*) and thus may not be applicable to other open source projects and/or universities. Nevertheless, the real-world setting of this field study offers the possibility to gain a profound understanding of the research topic acquired in a practical and realistic context, thus being of high relevance for both educators at universities or project leaders of other open source projects. During this study, neither the authors nor the students did have any control over the outcomes, thus having no opportunity to influence the results (internal validity). Data was collected from multiple sources to increase construct validity [Yin, 2017]. The data emerged from an initial survey in spring 2018, a final survey in fall 2019 and a continuous survey in that data was collected during a period of 14 months starting from July 2018. Data covering earlier years is not available which can be seen as a limitation. Further studies could continue to collect data for a longer period, also helping to identify possible changes over time. To get a more representative insight into the contributors' demographics, it would be necessary to apply the analysis described in Section 7.4.6 to all (sub-)repositories of the Catrobat foundation which may lead to different results. Additionally, further research could be conducted by comparing multiple cases of different open source projects with each other. However, due to a large number of different organizational and educational structures, it might be difficult to draw cross-case conclusions.

While previous research (e.g., [Fellhofer et al., 2015, Müller et al., 2019b, Müller et al., 2019a]) has focused on the organizational aspects from the educators' point of view, this work offers insights into the students' and contributors' perspective of open source software development at university. Starting to contribute to open source projects involves several challenges for both the students and the project's organizers. This section has shown that it is possible to establish such a setting at university and that students can benefit from contributing to open source projects within the scope of their study curricula. By sharing these insights the authors want to motivate other institutions to demonstrate their experience with academic open source software development courses at university.

7.4.8. Conclusion

In this section, we showed that by offering students the possibility to contribute to an agile open source project during their studies, it is possible to establish a close-to-practice environment for software development at university. To better understand the challenges and benefits of such a setting, data was collected from (a) three surveys, (b) an analysis of the communication recorded on Slack and GitHub; and, (c) semi-conducted expert interviews of long-term contributors to the Catrobat project. Since most students do not have any professional nor any other experience in software engineering, open source software development offers valuable and positive preparation for the professional world after graduation. The university-near organization of the Catrobat project allocates a familiar environment for students and provides them with necessary guidance and support. At Catrobat industry-near processes are simulated and students learn to handle tools frequently employed in the business sphere. In addition to this, students benefit from gaining communication and social skills by mentoring others and by collaborating with contributors from all around the world. From an educational perspective, this offers a unique setting at the university where both students and open source communities can benefit enormously. This section wants to motivate other institutions to set up similar working environments and share their experience of open source software development at university. Additionally, academics can gain valuable insights into real-world problems, thus facilitating further research in the areas of software development, distributed software development as well as open source software development.

8. Summary

8.1. Findings

This cumulative work sheds light onto current challenges and chances of not only OSS projects, but community-driven open projects in general. By identifying them, approaches got introduced and analyzed that help to conquer these challenges, supporting to understand and manage such projects in the long-term. This processes therewith also focus on creating a benefit for all involved stakeholders, providing the chance to ensure success and to drive innovation.

In regard to *RO1* (economics of open source), the limitations of common business model and analysis tools in open environments have been highlighted in Chapters 2and 3. The large amount of intangible values that is created and captured in OSS projects (*RO1.1*), brought up similarities to New Business Models and sustainable innovation. These research strands have similar characteristics to the investigated domain (i.e., often non-profit related and different forms of collaboration that is based upon intangible values). Consequently, applying tools that are emerging in these domains can also help to analyze and understand open source communities in a more holistic way. Especially the different incentives to contribute to such organizations can be covered by them and help to better understand how business and personal aspects are jointly present in the underlying ecosystems. This allows not only to illustrate and communicate all forms of value within these open settings, but also how the underlying systems work (*RO1.3*). Applying value-network analysis is capable to represent ecosystems that involve intangible value creation and capturing. It brought up the high dynamics of the relations between the different stakeholders (represented as actors in the value network frameworks) involved in OSS projects. As this work shows, applying and combining methods and (conceptual) tools from various domains enable to better understand the complex systems OSS organizations are today situated in. In contrast to the beginnings of OSS, these stakeholders are today not primarily contributors, but often businesses and other public stakeholders that influence the communities, but are in return also (often highly) dependent on them (*RO1.2*). Many commercial software products rely on OSS parts (e.g., libraries, operating systems, etc.) and therefore companies have a strong interest in getting actively involved in their development. Hence, the success and durability of OSS projects is dependent on if the needs of these involved commercial entities, but in particular also those of the volunteer contributors and other stakeholders can get balanced, creating a net-benefit for all involved parties.

The OSS analysis in Chapter 4 underpins previous research, outlining that the majority of work on the code-base of OSS projects is done by only by a small number of contributors (*RO2.1*). This active core is therefore needed for the success and long-term development of such projects. This aspect is especially important for managing such projects, since there seems to be just little change within this core and a continuous high dependency on the single top most contributor over several years (*RO2.2*). However, empirical research on OSS, as done in this thesis, can just represent code-contribution on a larger scale, therefore not measuring other forms of contributions (e.g., community support, code-reviews, or management tasks). As a result, to understand the impact of individual contributors on a specific project, further dimensions must be regarded and analyzed in detail. This needs to get incorporated in a bigger picture, also covering intangible value perspectives as mentioned before.

As described in Chapter 5, OSS projects in educational settings, such as Catrobat, have various benefits for involved students, the related educational institutes, and users (*RO 3.1*). On one hand the students get the chance to gain experience in a real-world project with an extensive code and active user base. On the other hand, especially in this project, they are involved in an environment creating value for others. In the presented case, the developed services are creating a positive outcome by support young users in teaching them how to program, which is also beneficial for educational entities using these services. These incentives have also been highlighted by the conducted survey of active students. Besides personal direct benefits (i.e., earning credits, working on a real project and gaining knowledge), the majority of active contributors highlighted the project's idea and activities as a motivator for them. It has also to been noted, as the results of the survey show, that many of these students have already before been engaged in another open source projects, indicating their interest for this kind of development and communities. This also provides a possibility for Universities, that can offer an appealing and relevant course-setting for students by introducing OSS development projects.

Projects as Catrobat, are built upon communities of primarily volunteer contributors, even though they are situated at a University (i.e., contribution is not a mandatory subject). This results in complex networks of different actors, each contributing different kind of value and expecting an individual benefit in return, as depicted with value network tools in Section 6. As described in detail in Section 7.1, these individual actors and their freedoms impact underlying projects in various ways (*RO3.2*). First, their aforementioned motives to contribute are very individual, but they are always based on a volunteer character (i.e., they have a personal aim they want to achieve). Hence, they need a certain autonomy and possibility to follow their own interests to start contributing and to stay with a project. This results in the need to balance these interests with the needs of the other involved (partly passive) stakeholders, such as users, supporters, or other organizations (also including commercial ones). Second, open source communities are usually distributed, often even over several time zones, which can also be seen at Catrobat. Working times and the availability for the project vary. This directly influences the predictability of the outcome, manageability of the whole project, and reachability of individual contributors for communication within the project. Pre-defined artifacts, such as specific meetings, fixed timelines, or a steady structure of the community, cannot be ensured or predicted at any time. Exactly these circumstances challenge agile software development methodologies, that are common to industry (*RO3.3*) and would provide the focus on people, that is needed in such projects. Also contributors highlighted these issues as hurdle to their contribution, although elements of *Extreme Programming* and *Kanban* have already been in use and were intended to support developers in their work. Thereout resulted the need for new processes and structures that cover these dynamics of the community.

These hurdles hinder that agile methodologies can be used "out of the box". But, certain tools out of these methodologies can get combined to achieve the needed balance, considering the needs of all actors involved (*R3.4*). The presented approach in Chapter 7 focuses on people, by providing certain freedoms, adaptability and inclusion of the contributors' ideas. Further, it provides more steerability for the project's management through constant communication and transparency, that also focuses on the needs of just peripherally involved stakeholders (e.g., users or supporters from industry). The conducted surveys (before and after introducing the workflow) underpin that in particular the improvements in communication and meetings (e.g., planning), are perceived positively by the contributors. Furthermore, having one dedicated center for coordination (i.e., the newly introduced Product Owner board), helped that the number of unhandled tickets and user requests got lowered, implicating a faster response to and inclusion of their individual needs.

8.2. Limitations

The conceptual and theoretical work in the knowledge-seeking part of this thesis, are in large parts related to research on *New Business Models (NBMs)* and OSS within an *Open Innovation (OI)* context. Especially the field of NBMs is still young and primarily focuses on sustainable domains. Therefore, the tools that emerged from this domain have just barely been used in pure technological information-systems. Therefore, application of tools out of this domain in this context must be carefully evaluated, as done in this thesis. Further research, as highlighted in the following section, might help to give a better understanding if these tools can be generally applied or there are limits in certain domains. Consequently, no general assumption on the applicability of NBM tools in different areas (i.e., others than NBM and OI) can be made, but they delivered reasonable and useful results at least in the domain of OSS and OI.

As described in the Introduction (Chapter 1), the aim of this work's solution-seeking part was to improve the processes in Catrobat. The therefore used realistic-oriented research strategy is suited to provide the needed specific real-world picture on the project, however, does not allow to draft a theory valid for all OSS projects. Although the presented approach of Catrobat can be seen as well-working case, the dynamics and different environments of open source projects, that got discussed in the previous chapters, hinder to make any assumption on how the introduced workflow or educational approach might be applicable for other projects and settings. It would be interesting to introduce similar processes in other OSS organizations and compare how they behave, also in respect to their ecosystem. But, this has not been possible for the author as external actor (researcher). Doing so would require to have access to, and trust in communities where this is intended to get done. This was also the reason to choose a research strategy that focuses on Catrobat, where the author is already part of the community for several years and has the trust of and access to the community. Therefore, the used revelatory case-study methodology situated in a real-word context, was the most adequate and insightful in regard to the specified aim of the thesis.

8.3. Future Work

Two main fields for future work can get derived from this thesis. First, the theoretical background of open source systems from a business side. The used methodologies from the research strands of *New Business Models* and *Sustainable Business Models* showed that they are applicable and well suited for the ecosystems of OSS projects. These strands are still increasingly gaining attention in academia and also importance for practitioners. Accordingly, new methodologies and concepts are constantly emerging. Considering open projects in validating them may be worth considering, but also applying them can be beneficial for the OSS organizations themselves, as this work shows. OSS characteristics are very similar to those of social and sustainable projects, therefore implicating that research in these fields can potentially be combined. This leaves room for investigating further cases not only from OSS, but also from OI, and drawing deeper implications for theory out of it. Especially the networked aspect of complex systems is expected to impact more domains in the future, underpinning the need for a more holistic research perspective by combining different domains.

Second, the Catrobat project is an ongoing project in an environment that is faced with constant change. Therefore, the proposed processes are suited and seem to be beneficial for the current situation, but need to be constantly evaluated and adapted. In addition, the introduced workflow is designed for the ongoing work of the project and its contributors, but does not provide any considerations for how to get into this workflow, i.e. how contributors can be supported in starting to work with Catrobat. As research has shown, the first experience with a project is essential to whether a contributor stays with a project or not. Investigating this first-contribution-period and providing an onboarding, but also the offboarding process, would be an interesting topic with manifold open questions for research (i.e., evaluating the contributors experience, reasons to leave a project, etc.). The motivational aspects investigated in this thesis can build the frame to design a research strategy and also introduce first steps within the project.

8.4. Conclusion

This thesis approached current chances and challenges that emerged through the evolution of OSS organizations, described in detail on the specific example of Catrobat. Whereas such projects have their origins in research settings, they are today an important part of our society and economy. Thus, they are increasingly connected to different entities, such as firms, government agencies, or educational institutes, ultimately resulting in complex ecosystems of diverse needs, motives, and value-activities. As the author shows, the corresponding value processes (i.e., creating and capturing value) are therefore hard to understand and manage, but doing so is crucial to ensure their long-lasting success. Common approaches from the classic business domain lack in representing the intangible values (i.e., non-monetary such as knowledge exchange, provision of services, or information flow) that are essential for these organizations. Analyzing such open projects with tools from comparatively new research-strands, such as value network analysis frameworks in New Business Models, that focus on sustainable businesses and entities, helps to overcome these shortcomings. As analyzing and comparing the domains showed, there are many similarities that can be utilized to gain a better understanding of value-dynamics in networked settings of both domains. Creating a bigger picture of an ecosystem, covering different dimensions, also including a social and motivational perspective, helps to create a shared understanding between all stakeholders of a system, which can be used to base management decisions and processes on. How this can be applied to OSS gets illustrated in the practical part of this work.

Analyzing OSS organizations from this new viewpoint and gaining deep insights into their ecosystems, especially their contributors, brought to light the related challenges of common management processes and frameworks from the software industry. The fact of high dynamics and the constant change of peripheral contributors and actors in the ecosystem, by having a steady and impactful core of members at the same time, amplifies these challenges. Whereas agile settings, e.g. Scrum or Kanban, are widely used and proofed as beneficial in practice for commercial projects, the dynamics of OSS organizations bring hurdles to them due to the projects' open settings. The individual motives of contributors, very diverse needs/expectations of the stakeholders, highly distributed developers, and the aforementioned dynamics between varying actors, got identified as main challenges that must be regarded in managing such projects. Creating a shared and holistic understanding of the whole ecosystem is consequently inevitable precondition to endorse change of the processes in such a project. The aim of these processes must be, to balance the needs and expectations of all involved actors. Volunteer contributors, that implement the services must keep their autonomy to reach their individual goals (e.g., learn a certain technology, gaining knowledge, working together in a community, etc.). At the same time, the requests of passive users must get considered, providing them useful and user-friendly services. Last, also external stakeholders, such as for Catrobat Universities and commercial entities that support the projects, have certain expectations (e.g., the development of a certain feature, supporting a specific target group, etc.), which must be regarded by the process. The introduced Product Owner workflow, provides the possibility to meet these requirements, by introducing a central point of contact with clear responsibilities, providing goals and transparency in creating value for all involved parties. This is intended to ensure long-lasting success for this OSS organization by jointly creating a net-benefit that is visible for all actors of the ecosystem.

Bibliography

[Ackermann and Eden, 2011] Ackermann, F. and Eden, C. (2011). Strategic management of stakeholders: Theory and practice. *Long range planning*, 44(3):179–196.

[Adiono et al., 2019] Adiono, T., Anindya, S. F., Fuada, S., Afifah, K., and Purwanda, I. G. (2019). Efficient android software development using mit app inventor 2 for bluetooth-based smart home. *Wireless Personal Communications*.

[Allee, 2009] Allee, V. (2009). Value creating networks: organizational issues and challenges. *The Learning Organization*, 16(6):427–442.

[Amit and Zott, 2001] Amit, R. and Zott, C. (2001). Value creation in e-business. *Strategic management journal*, 22(6-7):493–520.

[Amit et al., 2010] Amit, R., Zott, C., et al. (2010). Business model innovation: Creating value in times of change. Technical report, IESE Business School working paper No. 870.

[Andersen-Gott et al., 2012] Andersen-Gott, M., Ghinea, G., and Bygstad, B. (2012). Why do commercial companies contribute to open source software? *International Journal of Information Management*, 32(2):106–117.

[Awwad et al., 2017] Awwad, A. M. A., Schindler, C., Luhana, K. K., Ali, Z., and Spieler, B. (2017). Improving pocket paint usability via material design compliance and internationalization & localization support on application level. In *Proceedings of the 19th International Conference on Human-Computer Interaction with Mobile Devices and Services*, page 99. ACM.

[Awwad and Slany, 2016] Awwad, A. M. A. and Slany, W. (2016). Automated bidirectional languages localization testing for android apps with rich gui. *Mobile Information Systems*, 2016.

[Bacchelli and Bird, 2013] Bacchelli, A. and Bird, C. (2013). Expectations, outcomes, and challenges of modern code review. In *Proceedings of the 2013 international conference on software engineering*, pages 712–721. IEEE Press.

[Baden-Fuller and Haefliger, 2013] Baden-Fuller, C. and Haefliger, S. (2013). Business models and technological innovation. *Long range planning*, 46(6):419–426.

[Baden-Fuller and Morgan, 2010] Baden-Fuller, C. and Morgan, M. S. (2010). Business models as models. *Long range planning*, 43(2-3):156–171.

[Barcomb et al., 2018] Barcomb, A., Kaufmann, A., Riehle, D., Stol, K.-J., and Fitzgerald, B. (2018). Uncovering the periphery: A qualitative survey of episodic volunteering in free/libre and open source software communities. *IEEE Transactions on Software Engineering*.

[Baxter and Sommerville, 2011] Baxter, G. and Sommerville, I. (2011). Sociotechnical systems: From design methods to systems engineering. *Interacting with Computers*, 23(1):4–17.

[Baxter and Jack, 2008] Baxter, P. and Jack, S. (2008). Qualitative case study methodology: Study design and implementation for novice researchers. *The qualitative report*, 13(4):544–559.

[Beck et al., 2001] Beck, K., Beedle, M., Van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., et al. (2001). The agile manifesto.

[Beck and Gamma, 2000] Beck, K. and Gamma, E. (2000). *Extreme programming explained: embrace change*. addison-wesley professional.

[Becker et al., 2011] Becker, J., Breuker, D., and Rauer, H. P. (2011). On guidelines for representing business models-a design science approach. In *AMCIS*.

[Beecham et al., 2017] Beecham, S., Clear, T., Damian, D., Barr, J., Noll, J., and Scacchi, W. (2017). How best to teach global software engineering? educators are divided. *IEEE Software*, 34(1):16–19.

[Behnamghader et al., 2017] Behnamghader, P., Alfayez, R., Srisopha, K., and Boehm, B. (2017). Towards better understanding of software quality evolution through commit-impact analysis. In *Proceedings of the 2017 IEEE International Conference on Software Quality, Reliability and Security*, pages 251–262.

[Biem and Caswell, 2008] Biem, A. and Caswell, N. (2008). A value network model for strategic analysis. In *Proceedings of the 41st annual Hawaii international conference on system sciences (HICSS 2008)*, pages 361–361. IEEE.

[Biloslavo et al., 2018] Biloslavo, R., Bagnoli, C., and Edgar, D. (2018). An eco-critical perspective on business models: The value triangle as an approach to closing the sustainability gap. *Journal of cleaner production*, 174:746–762.

[Bocken et al., 2013] Bocken, N., Short, S., Rana, P., and Evans, S. (2013). A value mapping tool for sustainable business modelling. *Corporate Governance*, 13(5):482–497.

[Boehm, 2003] Boehm, B. (2003). Value-based software engineering. *ACM SIGSOFT Software Engineering Notes*, 28(2):4.

[Bouwman et al., 2008] Bouwman, H., Vos, J. D., and Haaker, T. (2008). *Mobile Service Innovation and Business Models*. Springer-Verlag Berlin Heidelberg.

[Brechner, 2015] Brechner, E. (2015). *Agile Project Management with Kanban*. Microsoft Press.

[Brehmer et al., 2018] Brehmer, M., Podoynitsyna, K., and Langerak, F. (2018). Sustainable business models as boundary-spanning systems of value transfers. *Journal of Cleaner Production*, 172:4514 – 4531.

[Breuer et al., 2018] Breuer, H., Fichter, K., Lüdeke-Freund, F., Tiemann, I., et al. (2018). Sustainability-oriented business model development: Principles, criteria, and tools. *International Journal of Entrepreneurial Venturing*, 10(2):256–286.

[Breuer and Ketabdar, 2012] Breuer, H. and Ketabdar, H. (2012). User-driven business model innovation–new formats and methods in business modeling and interaction design, and the case of magitact. In *Proceedings of IADIS International Conference on E-Society*, pages 211–218.

[Breuer and Lüdeke-Freund, 2014] Breuer, H. and Lüdeke-Freund, F. (2014). Normative innovation for sustainable business models in value networks. In *The Proceedings of XXV ISPIM Conference-Innovation for Sustainable Economy and Society*, pages 120–148.

[Breuer and Lüdeke-Freund, 2017] Breuer, H. and Lüdeke-Freund, F. (2017). *Values-Based Innovation Management: Innovating by What We Care About*. Macmillan International Higher Education.

[Breuer and Lüdeke-Freund, 2017] Breuer, H. and Lüdeke-Freund, F. (2017). Values-based network and business model innovation. *International Journal of Innovation Management*, 21(03):1750028.

[Bruckman, 1999] Bruckman, A. (1999). Can educational be fun. In *Game developers conference*, volume 99, pages 75–79.

[Bryson, 2004] Bryson, J. M. (2004). What to do when stakeholders matter: stakeholder identification and analysis techniques. *Public management review*, 6(1):21–53.

[Casson and Hawthorn, 2011] Casson, A. and Hawthorn, L. (2011). Introducing the oregon state university open source lab. *Open Source Business Resource*.

[Chao and Brown, 2009] Chao, T. and Brown, K. (2009). Empowering students and the community through agile software development service-learning. In *Agile Processes in Software Engineering and Extreme Programming*, pages 104–113.

[Chesbrough, 2006a] Chesbrough, H. (2006a). *Open business models: How to thrive in the new innovation landscape*. Harvard Business Press.

[Chesbrough, 2006b] Chesbrough, H. (2006b). *Open business models: How to thrive in the new innovation landscape*. Harvard Business Press.

[Chesbrough and Rosenbloom, 2002] Chesbrough, H. and Rosenbloom, R. S. (2002). The role of the business model in capturing value from innovation: evidence from xerox corporation's technology spin-off companies. *Industrial and corporate change*, 11(3):529–555.

[Chesbrough, 2006c] Chesbrough, H. W. (2006c). *Open innovation: The new imperative for creating and profiting from technology*. Harvard Business Press.

[Chesbrough, 2007] Chesbrough, H. W. (2007). Why companies should have open business models. *MIT Sloan management review*, 48(2):22.

[Chesbrough and Appleyard, 2007] Chesbrough, H. W. and Appleyard, M. M. (2007). Open innovation and strategy. *California management review*, 50(1):57–76.

[Cockburn and J., 2001] Cockburn, A. and J., H. (2001). Agile software development: The people factor. *Computer*, 34(11):131–133.

[Cockburn and Williams, 2001] Cockburn, A. and Williams, L. (2001). The costs and benefits of pair programming. In *Extreme Programming Examined*, pages 223–243. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Crowston et al., 2008] Crowston, K., Wei, K., Howison, J., and Wiggins, A. (2008). Free/libre open-source software development: What we know and what we do not know. *ACM Comput. Surv.*, 44(2):7:1–7:35.

[Crowston et al., 2006] Crowston, K., Wei, K., Li, Q., and Howison, J. (2006). Core and periphery in free/libre and open source software team communications. In *Proceedings of the 39th Annual Hawaii International Conference on System Sciences, 2006. HICSS'06.*, volume 6, pages 118a–118a. IEEE.

[Dabbagh and Kitsantas, 2012] Dabbagh, N. and Kitsantas, A. (2012). Personal learning environments, social media, and self-regulated learning: A natural formula for connecting formal and informal learning. *The Internet and higher education*, 15(1):3–8.

[DeKoenigsberg, 2008] DeKoenigsberg, G. (2008). How successful open source projects work, and how and why to introduce students to the open source world. In *Software Engineering Education and Training, 2008. CSEET'08. IEEE 21st Conference on*, pages 274–276. IEEE.

[Delone and McLean, 2003] Delone, W. H. and McLean, E. R. (2003). The delone and mclean model of information systems success: A ten-year update. *J. Manage. Inf. Syst.*, 19(4).

[Dentchev et al., 2016] Dentchev, N., Baumgartner, R., Dieleman, H., Jóhannsdóttir, L., Jonker, J., Nyberg, T., Rauter, R., Rosano, M., Snihur, Y., Tang, X., et al. (2016). Embracing the variety of sustainable business models: social entrepreneurship, corporate intrapreneurship, creativity, innovation, and other approaches to sustainability challenges. *Journal of Cleaner Production*.

[Ding et al., 2014] Ding, W., Liang, P., Tang, A., Van Vliet, H., and Shahin, M. (2014). How do open source communities document software architecture: An exploratory survey. In *2014 19th International conference on engineering of complex computer systems*, pages 136–145. IEEE.

[Dionisio et al., 2007] Dionisio, J. D. N., Dickson, C. L., August, S. E., Dorin, P. M., and Toal, R. (2007). An open source software culture in the undergraduate computer science curriculum. *ACM SIGCSE Bulletin*, 39(2):70–74.

[Dougherty, 2012] Dougherty, D. (2012). The maker movement. *Innovations: Technology, Governance, Globalization*, 7(3):11–14.

[Ehls, 2017] Ehls, D. (2017). Open source project collapse–sources and patterns of failure. In *Proceedings of the 50th Hawaii International Conference on System Sciences*.

[Ehrenfeld, 2000] Ehrenfeld, J. R. (2000). Colorless green ideas sleep furiously: Is the emergence of" sustainable" practices meaningful? *Reflections: The sol journal*, 1(4):34–47.

[Ellis et al., 2007] Ellis, H. J., Morelli, R. A., De Lanerolle, T. R., and Hislop, G. W. (2007). Holistic software engineering education based on a humanitarian open source project. In *Software Engineering Education & Training, 2007. CSEET'07. 20th Conference on*, pages 327–335. IEEE.

[eMarketer, 2016] eMarketer (2016). Number of smartphone users worldwide from 2014 to 2020 (in billions). Statista - The Statistics Portal, Statista. accessed: 12th June, 2018.

[Evans, 2003] Evans, D. S. (2003). Some empirical aspects of multi-sided platform industries. *Review of Network Economics*, 2(3).

[Fassin, 2009] Fassin, Y. (2009). The stakeholder model refined. *Journal of Business Ethics*, 84(1):113–135.

[Fellhofer et al., 2015] Fellhofer, S., Harzl, A., and Slany, W. (2015). Scaling and internationalizing an agile foss project: Lessons learned. In *IFIP International Conference on Open Source Systems*, pages 13–22. Springer.

[Fink, 2003] Fink, M. (2003). *The business and economics of Linux and open source*. Prentice Hall PTR.

[Finley, 2016] Finley, K. (2016). Open source won. so, now what? *Wired*. Accessed: 12 July 2019.

[Fogel, 2005] Fogel, K. (2005). *Producing open source software: How to run a successful free software project*. " O'Reilly Media, Inc.".

[Fogel and Bar, 1999] Fogel, K. and Bar, M. (1999). *Open source development with CVS*. Coriolis Group Books.

[Foxon et al., 2015] Foxon, T. J., Bale, C. S., Busch, J., Bush, R., Hall, S., and Roelich, K. (2015). Low carbon infrastructure investment: extending business models for sustainability. *Infrastructure Complexity*, 2(1):4.

[Furseth and Cuthbertson, 2013] Furseth, P. I. and Cuthbertson, R. (2013). The service innovation triangle: a tool for exploring value creation through service innovation. *International Journal of Technology Marketing*, 8(2):159–176.

[Gacek and Arief, 2004] Gacek, C. and Arief, B. (2004). The many meanings of open source. *IEEE software*, 21(1):34–40.

[Gassmann et al., 2010] Gassmann, O., Enkel, E., and Chesbrough, H. (2010). The future of open innovation. *R&d Management*, 40(3):213–221.

[Ghezzi and Mandrioli, 2005] Ghezzi, C. and Mandrioli, D. (2005). The challenges of software engineering education. In *International Conference on Software Engineering*, pages 115–127. Springer.

[Gjerde et al., 2007] Gjerde, I., Eskedal, T., and Venturin, R. (2007). Biztekon a framework for business modelling and techno-economic analysis. *Telecommunications, 2007. ConTel 2007. 9th International Conference on*.

[Google, 2014] Google (2014). Share of App Users Who Have Stopped Using An App Because It Was Not Localized Properly as of March 2014. Statista - The Statistics Portal, Statista. accessed: 25th June, 2018.

[Gordijn et al., 2000] Gordijn, J., Akkermans, H., and van Vliet, H. (2000). Business modelling is not process modelling. In Liddle, S. W., Mayr, H. C., and Thalheim, B., editors, *Conceptual Modeling for E-Business and the Web*, pages 40–51, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Hannebauer and Gruhn, 2017] Hannebauer, C. and Gruhn, V. (2017). On the relationship between newcomer motivations and contribution barriers in open source projects. In *Proceedings of the 13th International Symposium on Open Collaboration*, page 2. ACM.

[Hars and Ou, 2001] Hars, A. and Ou, S. (2001). Working for free? - motivations of participating in open source projects. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)-Volume 7 - Volume 7*, HICSS '01.

[Hars and Ou, 2002] Hars, A. and Ou, S. (2002). Working for free? motivations for participating in open-source projects. *International Journal of Electronic Commerce*, 6(3):25–39.

[Harzl, 2016] Harzl, A. (2016). Combining foss and kanban: An action research. In *IFIP International Conference on Open Source Systems*, pages 71–84. Springer.

[Harzl, 2017] Harzl, A. (2017). Can foss projects benefit from integrating kanban: a case study. *Journal of Internet Services and Applications*, 8(1).

[Heppler et al., 2016] Heppler, L., Eckert, R., and Stuermer, M. (2016). Who cares about my feature request? In *IFIP International Conference on Open Source Systems*, pages 85–96. Springer.

[Herold, 2019] Herold, M. (2019). Communication in an agile foss project - a socio-technical case study. Unpublished Master's Thesis at Graz University of Technology.

[Highsmith, 2002] Highsmith, J. (2002). What is agile software development? *The Journal of Defense Software Engineering*, 14(10):4–9.

[Hippel and Krogh, 2003] Hippel, E. v. and Krogh, G. v. (2003). Open source software and the "private-collective" innovation model: Issues for organization science. *Organization science*, 14(2):209–223.

[IDC, 2018a] IDC (2018a). Global smartphone shipments forecast from 2010 to 2021 (in million units). Statista - The Statistics Portal, Statista. accessed: 12th July, 2018.

[IDC, 2018b] IDC (2018b). Personal computer (pc) shipments (desktop and portable/notebook) worldwide from 2009 to 2022 (in million units). Statista - The Statistics Portal, Statista. accessed: 12th June, 2018.

[Jensen et al., 2011] Jensen, C., King, S., and Kuechler, V. (2011). Joining free/open source software communities: An analysis of newbies' first interactions on project mailing lists. In *2011 44th Hawaii international conference on system sciences*, pages 1–10. IEEE.

[Johnson et al., 2008] Johnson, M. W., Christensen, C. M., and Kagermann, H. (2008). Reinventing your business model. *Harvard business review*, 86(12):57–68.

[Jones and Upward, 2014] Jones, P. and Upward, A. (2014). Caring for the future: The systemic design of flourishing enterprises. In *Proceedings of RSD3, Third Symposium of Relating Systems Thinking to Design*, pages 1–8.

[Jonker, 2012] Jonker, J. (2012). *New Business Models: a explorative study of changing transactions creating multiple values*. Doetinchem: Jab management consultants bv.

[Joyce and Paquin, 2016] Joyce, A. and Paquin, R. L. (2016). The triple layered business model canvas: A tool to design more sustainable business models. *Journal of Cleaner Production*, 135:1474–1486.

[Kagdi et al., 2008] Kagdi, H., Hammad, M., and Maletic, J. I. (2008). Who can help me with this source code change? In *Software Maintenance, 2008. ICSM 2008. IEEE International Conference on*, pages 157–166. IEEE.

[Kamargianni and Matyas, 2017] Kamargianni, M. and Matyas, M. (2017). The business ecosystem of mobility-as-a-service. In *transportation research board*, volume 96. Transportation Research Board.

[Kang et al., 2015] Kang, H., Cho, J., and Kim, H. (2015). Application study on android application prototyping method using app inventor. *Indian Journal of Science and Technology*, 8(19).

[Koch, 2004] Koch, S. (2004). Agile principles and open source software development: A theoretical and empirical discussion. In Eckstein, J. and Baumeister, H., editors, *Extreme Programming and Agile Processes in Software Engineering*, pages 85–93, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Krishnamurthy, 2005a] Krishnamurthy, S. (2005a). An analysis of open source business models. In *Eds.) Perspectives on Free and Open Source Software*. The MIT Press.

[Krishnamurthy, 2005b] Krishnamurthy, S. (2005b). Cave or community? an empirical examination of 100 mature open source projects (originally published in volume 7, number 6, june 2002). *First Monday*.

[Lacey, 2012] Lacey, M. (2012). *The scrum field guide: Practical advice for your first year*. Addison-Wesley Professional.

[Lakhan and Jhunjhunwala, 2008] Lakhan, S. E. and Jhunjhunwala, K. (2008). Open source software in education. *Educause Quarterly*, 31(2):32.

[Lee and Carver, 2017] Lee, A. and Carver, J. C. (2017). Are one-time contributors different? a comparison to core and periphery developers in FLOSS repositories. In *Empirical Software Engineering and Measurement (ESEM), 2017 ACM/IEEE International Symposium on*, pages 1–10. IEEE.

[Lee et al., 2018] Lee, M., Yun, J., Pyka, A., Won, D., Kodama, F., Schiuma, G., Park, H., Jeon, J., Park, K., Jung, K., et al. (2018). How to respond to the fourth industrial revolution, or the second information technology revolution? dynamic new combinations between technology, market, and society through open innovation. *Journal of Open Innovation: Technology, Market, and Complexity*, 4(3):21.

[Lenhart et al., 2015] Lenhart, A., Duggan, M., Perrin, A., Stepler, R., Rainie, H., and Parker, K. (2015). *Teens, social media & technology overview 2015*. Pew Research Center [Internet & American Life Project].

[Lerner and Tirole, 2002] Lerner, J. and Tirole, J. (2002). Some simple economics of open source. *The journal of industrial economics*, 50(2):197–234.

[Lüdeke-Freund and Dembek, 2017] Lüdeke-Freund, F. and Dembek, K. (2017). Sustainable business model research and practice: Emerging field or passing fancy? *Journal of Cleaner Production*, 168:1668–1678.

[Maloney et al., 2010] Maloney, J., Resnick, M., and Rusk, N. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education*, 10(4):1–15.

[Massa et al., 2018] Massa, L., Gianluigi, V., and Tucci, C. (2018). Business models and complexity. *Journal of Business Models*, 6(1):59–71.

[Massa and Tucci, 2013] Massa, L. and Tucci, C. L. (2013). Business model innovation. *The Oxford handbook of innovation management*, 20(18):420–441.

[Max-Neef et al., 1991] Max-Neef, M., Dlizalde, A., and Hopenhayn, M. (1991). *Human scale development: Conception, application and further reflections*. Apex Press.

[McManus, 2013] McManus, S. (2013). *Scratch programming in easy steps: covers Versions 2.0 and 1.4*. In Easy Steps.

[Michelini and Fiorentino, 2012] Michelini, L. and Fiorentino, D. (2012). New business models for creating shared value. *Social Responsibility Journal*, 8(4):561–577.

[Müller, 2018] Müller, M. (2018). Agile challenges and chances for open source: lessons learned from managing a floss project. In *2018 IEEE Conference on Open Systems (ICOS)*, pages 1–6. IEEE.

[Müller, 2019] Müller, M. (2019). Managing the open cathedral. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pages 1176–1179.

[Müller et al., 2018] Müller, M., Schindler, C., Luhana, K. K., and Slany, W. (2018). Enabling teenagers to create and share apps. In *2018 IEEE Conference on Open Systems (ICOS)*, pages 25–30. IEEE.

[Müller et al., 2019a] Müller, M., Schindler, C., and Slany, W. (2019a). Engaging students in open source: Establishing foss development at a university. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*, pages 7721–7730.

[Müller et al., 2019b] Müller, M., Schindler, C., and Slany, W. (2019b). Introducing agile product owners in a floss project. In *IFIP International Conference on Open Source Systems*, pages 38–43. Springer.

[Müller et al., 2019c] Müller, M., Schindler, C., and Slany, W. (2019c). Pocket code-a mobile visual programming framework for app development. In *2019 IEEE/ACM 6th International Conference on Mobile Software Engineering and Systems (MOBILESoft)*, pages 140–143. IEEE.

[Müller et al., 2019d] Müller, M., Vorraber, W., Herold, M., Schindler, C., Slany, W., and Tanaka, K. (2019d). Streamlining value in a foss project. In *Proceedings of the 13th European Conference on Software Architecture-Volume 2*, pages 231–234. ACM.

[Müller et al., 2019e] Müller, M., Vorraber, W., and Slany, W. (2019e). Open principles in new business models for information systems. *Journal of Open Innovation: Technology, Market, and Complexity*, 5(1):6.

[Mumford, 2006] Mumford, E. (2006). The story of socio-technical design: Reflections on its successes, failures and potential. *Information Systems Journal*, 16(4):317–342.

[Nair and Nair, 2014] Nair, P. R. and Nair, V. D. (2014). Organization of a research paper: The imrad format. In *Scientific writing and communication in agriculture and natural resources*, pages 13–25. Springer.

[Nakakoji et al., 2002] Nakakoji, K., Yamamoto, Y., Nishinaka, Y., Kishida, K., and Ye, Y. (2002). Evolution patterns of open-source software systems and communities. In *Proceedings of the international workshop on Principles of software evolution*, pages 76–85. ACM.

[Noone and Mooney, 2018] Noone, M. and Mooney, A. (2018). Visual and textual programming languages: a systematic review of the literature. *Journal of Computers in Education*, 5(2):149–174.

[O'Keeffe et al., 2011] O'Keeffe, G. S., Clarke-Pearson, K., et al. (2011). The impact of social media on children, adolescents, and families. *Pediatrics*, 127(4):800–804.

[Oreg and Nov, 2008] Oreg, S. and Nov, O. (2008). Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values. *Computers in human behavior*, 24(5):2055–2073.

[Osterwalder and Pigneur, 2002] Osterwalder, A. and Pigneur, Y. (2002). An ebusiness model ontology for modeling ebusiness. *BLED 2002 proceedings*, page 2.

[Osterwalder and Pigneur, 2010] Osterwalder, A. and Pigneur, Y. (2010). *Business model generation: a handbook for visionaries, game changers, and challengers*. John Wiley & Sons.

[Osterwalder et al., 2014] Osterwalder, A., Pigneur, Y., Bernarda, G., and Smith, A. (2014). *Value proposition design: how to create products and services customers want*. John Wiley & Sons.

[Papert and Harel, 1991] Papert, S. and Harel, I. (1991). *Situating Constructionism*, chapter 1. Ablex Publishing Corporation.

[Park, 2017] Park, H. S. (2017). Technology convergence, open innovation, and dynamic economy. *Journal of Open Innovation: Technology, Market, and Complexity*, 3(4):24.

[Partsch, 2010] Partsch, H. A. (2010). *Requirements-Engineering Systematisch*. Springer DE.

[Pavie et al., 2014] Pavie, X., Scholten, V., and Carthy, D. (2014). *Responsible Innovation - From Concept to practice*. World Scientific Publishing Company.

[Peppard and Rylander, 2006] Peppard, J. and Rylander, A. (2006). From value chain to value network:: Insights for mobile operators. *European Management Journal*, 24(2):128 – 141.

[Pichler, 2010] Pichler, R. (2010). *Agile product management with scrum: Creating products that customers love*. Addison-Wesley Professional.

[Pinto et al., 2019] Pinto, G., Ferreira, C., Souza, C., Steinmacher, I., and Meirelles, P. (2019). Training software engineers using open-source software: the students' perspective. In *Proceedings of the 41st International Conference on Software Engineering: Software Engineering Education and Training*, pages 147–157. IEEE Press.

[Pinto et al., 2017] Pinto, G. H. L., Figueira Filho, F., Steinmacher, I., and Gerosa, M. A. (2017). Training software engineers using open-source software: the professors' perspective. In *2017 IEEE 30th Conference on Software Engineering Education and Training (CSEE&T)*, pages 117–121. IEEE.

[Porter and Lawler, 1968] Porter, L. W. and Lawler, E. E. (1968). *Managerial Attitudes and Performance -*. Richard D. Irwin, Illinois.

[Porter and Kramer, 2011] Porter, M. E. and Kramer, M. R. (2011). Creating shared value. *Harvard Business Review*, January-February.

[Price and Barnes, 2015] Price, T. W. and Barnes, T. (2015). Comparing textual and block interfaces in a novice programming environment. In *Proceedings of the Eleventh Annual International Conference on International Computing Education Research*, ICER '15, pages 91–99, New York, NY, USA. ACM.

[Raymond, 1999] Raymond, E. (1999). The cathedral and the bazaar. *Knowledge, Technology & Policy*, 12(3):23–49.

[Lifelong Kindergarten Group at the MIT Media Lab, 2018] Lifelong Kindergarten Group at the MIT Media Lab (2018). Scratch statistics. accessed: 15th June, 2018.

[Resnick et al., 2009] Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., et al. (2009). Scratch: programming for all. *Communications of the ACM*, 52(11):60–67.

[Riehle et al., 2014] Riehle, D., Riemer, P., Kolassa, C., and Schmidt, M. (2014). Paid vs. volunteer work in open source. In *2014 47th Hawaii International Conference on System Sciences (HICSS)*, pages 3286–3295. IEEE.

[Robinson and Sharp, 2010] Robinson, H. and Sharp, H. (2010). Collaboration, communication and co-ordination in agile software development practice. In *Collaborative software engineering*, pages 93–108. Springer.

[Runkel and McGrath, 1972] Runkel, P. J. and McGrath, J. E. (1972). *Research on human behavior: A systematic guide to method*. Holt, Rinehart & Winston of Canada Ltd.

[Rupp, 2009] Rupp, C. (2009). *Requirements Engineering und -Management*. Carl Hanser Verlag München Wien.

[Sandoval-Reyes et al., 2011] Sandoval-Reyes, S., Galicia-Galicia, P., and Gutierrez-Sanchez, I. (2011). Visual learning environments for computer programming. In *2011 IEEE Electronics, Robotics and Automotive Mechanics Conference*, pages 439–444.

[Schranz et al., 2019] Schranz, T., Schindler, C., Müller, M., and Slany, W. (2019). Contributors' impact on a foss project's quality. In *Proceedings of the 2nd ACM SIGSOFT International Workshop on Software Qualities and Their Dependencies*, pages 35–38. ACM.

[Schreieck et al., 2018] Schreieck, M., Hein, A., Wiesche, M., and Krcmar, H. (2018). The challenge of governing digital platform ecosystems. In Linnhoff-Popien, C., Schneider, R., and Zaddach, M., editors, *Digital Marketplaces Unleashed*, pages 527–538. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Schwaber, 2004] Schwaber, K. (2004). *Agile project management with Scrum*. Microsoft press.

[Schwaber and Sutherland, 2017] Schwaber, K. and Sutherland, J. (2017). The scrum guide. *Scrum Alliance*.

[Sedano et al., 2016] Sedano, T., Ralph, P., and Péraire, C. (2016). Practice and perception of team code ownership. In *Proceedings of the 20th international conference on evaluation and assessment in software engineering*, page 36. ACM.

[Seely Brown and Adler, 2008] Seely Brown, J. and Adler, R. (2008). Open education, the long tail, and learning 2.0. *Educause review*, 43(1):16–20.

[Silva et al., 2017] Silva, J., Wiese, I., German, D., Steinmacher, I., and Gerosa, M. A. (2017). How long and how much: What to expect from summer of code participants? In *2017 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 69–79. IEEE.

[Slany, 2012] Slany, W. (2012). A mobile visual programming system for android smartphones and tablets. In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 265–266. IEEE.

[Slany, 2014a] Slany, W. (2014a). Pocket Code: A Scratch-like integrated development environment for your phone. In *Proceedings of the Companion Publication of the 2014 ACM SIGPLAN Conference on Systems, Programming, and Applications: Software for Humanity*, SPLASH '14, pages 35–36, New York, NY, USA. ACM.

[Slany, 2014b] Slany, W. (2014b). Tinkering with pocket code, a scratch-like programming app for your smartphone. *Proceedings of Constructionism*.

[Stallman, 2009] Stallman, R. (2009). Why open source misses the point of free software. *Communications of the ACM*, 52(6):31–33.

[Stallman et al., 2002] Stallman, R. M., Lessig, L., and Gay, J. (2002). *Free software, free society: selected essays of Richard M. Stallman*. GNU Press, 1st edition.

[Steinmacher et al., 2015] Steinmacher, I., Conte, T., Gerosa, M. A., and Redmiles, D. (2015). Social barriers faced by newcomers placing their first contribution in open source software projects. In *Proceedings of the 18th ACM conference on Computer supported cooperative work & social computing*, pages 1379–1392. ACM.

[Steinmacher et al., 2012] Steinmacher, I., Wiese, I. S., and Gerosa, M. A. (2012). Recommending mentors to software project newcomers. In *2012 Third International Workshop on Recommendation Systems for Software Engineering (RSSE)*, pages 63–67. IEEE.

[Stol and Fitzgerald, 2018] Stol, K.-J. and Fitzgerald, B. (2018). The abc of software engineering research. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 27(3):11.

[Sverrisdottir et al., 2014] Sverrisdottir, H. S., Ingason, H. T., and Jonasson, H. I. (2014). The role of the product owner in scrum-comparison between theory and practices. *Procedia-Social and Behavioral Sciences*, 119:257–267.

[Tafliovich et al., 2019] Tafliovich, A., Estrada, F., and Caswell, T. (2019). Teaching software engineering with free open source software development: An experience report. In *Proceedings of the 52nd Hawaii International Conference on System Sciences*.

[Tandemic, 2017] Tandemic (2017). Social business model canvas. Online Available: http://www.socialbusinessmodelcanvas.com (Last Accessed: 2019-01-27).

[Täuscher and Abdelkafi, 2017] Täuscher, K. and Abdelkafi, N. (2017). Visual tools for business model innovation: Recommendations from a cognitive perspective. *Creativity and Innovation Management*, 26(2):160–174.

[Terceiro et al., 2010] Terceiro, A., Rios, L. R., and Chavez, C. (2010). An empirical study on the structural complexity introduced by core and peripheral developers in free software projects. In *Software Engineering (SBES), 2010 Brazilian Symposium on*, pages 21–29. IEEE.

[Teufel et al., 2004] Teufel, S., Götte, S., and Steiner, M. (2004). *Managementmethoden für ICT-Unternehmen: aktuelles Wissen von Forschenden des iimt der Universite de Fribourg und Spezialisten aus der Praxis*. Verlag Industrielle Organisation Zürich.

[Tiemann and Fichter, 2016] Tiemann, I. and Fichter, K. (2016). Developing business models with the sustainable business canvas: manual for conducting workshops.

[Timmers, 1998] Timmers, P. (1998). Business models for electronic markets. *Electronic markets*, 8(2):3–8.

[Torres et al., 2011] Torres, M. M., Toral, S., Perales, M., and Barrero, F. (2011). Analysis of the core team role in open source communities. In *Complex, Intelligent and Software Intensive Systems (CISIS), 2011 International Conference on*, pages 109–114. IEEE.

[Trainer et al., 2014] Trainer, E., Chaihirunkarn, C., Kalyanasundaram, A., and Herbsleb, J. D. (2014). Community code engagements: summer of code & hackathons for community building in scientific software. In *Proceedings of the 18th International Conference on Supporting Group Work*, pages 111–121. ACM.

[Trist, 1981] Trist, E. (1981). The evolution of socio-technical systems. *Occasional paper*, (2):1–67.

[Tsirakidis et al., 2009] Tsirakidis, P., Kobler, F., and Krcmar, H. (2009). Identification of success and failure factors of two agile software development teams in an open source organization. In *Global Software Engineering, 2009. ICGSE 2009. Fourth IEEE International Conference on*, pages 295–296. IEEE.

[Turnu et al., 2006] Turnu, I., Melis, M., Cau, A., Setzu, A., Concas, G., and Mannaro, K. (2006). Modeling and simulation of open source development using an agile practice. *Journal of Systems Architecture*, 52(11):610–618.

[United Nations World Commission on Environment and Development, 1987] United Nations World Commission on Environment and Development (1987). Report of the world commission on environment and development: Our common future. Technical report, United Nations, New York, NY, USA.

[Upward and Jones, 2016] Upward, A. and Jones, P. (2016). An ontology for strongly sustainable business models: Defining an enterprise framework compatible with natural and social science. *Organization & Environment*, 29(1):97–123.

[van Rooij, 2011] van Rooij, S. W. (2011). Higher education sub-cultures and open source adoption. *Computers & Education*, 57(1):1171–1183.

[Von Hippel, 2001] Von Hippel, E. (2001). Learning from open-source software. *MIT Sloan management review*, 42(4):82–86.

[von Hippel, 2017] von Hippel, E. (2017). Free innovation by consumers—how producers can benefit. *Research-Technology Management*, 60(1):39–42.

[von Hippel and Jin, 2009] von Hippel, E. and Jin, C. (2009). The major shift towards user-centred innovation. *Journal of Knowledge-based Innovation in China*.

[Von Krogh et al., 2012] Von Krogh, G., Haefliger, S., Spaeth, S., and Wallin, M. W. (2012). Carrots and rainbows: Motivation and social practice in open source software development. *MIS quarterly*, pages 649–676.

[Von Krogh and Von Hippel, 2006] Von Krogh, G. and Von Hippel, E. (2006). The promise of research on open source software. *Management science*, 52(7):975–983.

[von Schomberg, 2013] von Schomberg, R. (2013). *A Vision of Responsible Research and Innovation*. John Wiley & Sons, Ltd.

[Vorraber et al., 2016] Vorraber, W., Lichtenegger, G., Brugger, J., Gojmerac, I., Egly, M., Panzenböck, K., Exner, E., Aschbacher, H., Christian, M., and Voessner, S. (2016). Designing information systems to facilitate civil-military cooperation in disaster management. *International Journal of Distributed Systems and Technologies (IJDST)*, 7(4):22–40.

[Vorraber et al., 2019a] Vorraber, W., Mueller, M., Voessner, S., and Slany, W. (2019a). Analyzing and managing complex software ecosystems: A framework to understand value in information systems. *IEEE Software*, 36(3):55–60.

[Vorraber and Müller, 2019] Vorraber, W. and Müller, M. (2019). A networked analysis and engineering framework for new business models. *Sustainability*, 11(21):6018.

[Vorraber et al., 2019b] Vorraber, W., Neubacher, D., Moesl, B., Brugger, J., Stadlmeier, S., and Voessner, S. (2019b). Uctm—an ambidextrous service innovation framework—a bottom-up approach to combine human-and technology-centered service design. *Systems*, 7(2):23.

[Vorraber and Vössner, 2011] Vorraber, W. and Vössner, S. (2011). Modeling endogenous motivation and exogenous influences in value networks of information service systems. *Journal of convergence information technology*, 6(8):356–363.

[Vroom, 1964] Vroom, V. H. (1964). *Work and motivation.* Wiley.

[Wagner, 2018] Wagner, M. (2018). The open source revolution is over – the revolutionaries won. *Light Reading*. Accessed: 12 July 2019.

[Warsta and Abrahamsson, 2003] Warsta, J. and Abrahamsson, P. (2003). Is open source software development essentially an agile method. In *Proceedings of the 3rd Workshop on Open Source Software Engineering*, pages 143–147.

[Weber, 2004] Weber, S. (2004). *The success of open source.* Harvard University Press.

[Webster and Watson, 2002] Webster, J. and Watson, R. T. (2002). Analyzing the past to prepare for the future: Writing a literature review. *MIS quarterly*, pages xiii–xxiii.

[Weintrop and Wilensky, 2015] Weintrop, D. and Wilensky, U. (2015). To block or not to block, that is the question: students' perceptions of blocks-based programming. In *Proceedings of the 14th International Conference on Interaction Design and Children*, pages 199–208. ACM.

[Weintrop and Wilensky, 2017] Weintrop, D. and Wilensky, U. (2017). Comparing block-based and text-based programming in high school computer science classrooms. *ACM Trans. Comput. Educ.*, 18(1):3:1–3:25.

[West and Lakhani, 2008] West, J. and Lakhani, K. R. (2008). Getting clear about communities in open innovation. *Industry and Innovation*, 15(2):223–231.

[West and O'Mahony, 2005] West, J. and O'Mahony, S. (2005). Contrasting community building in sponsored and community founded open source projects. In *Proceedings of the 38th Annual Hawaii International Conference on System Sciences*, pages 196c–196c. IEEE.

[West and O'mahony, 2008] West, J. and O'mahony, S. (2008). The role of participation architecture in growing sponsored open source communities. *Industry and innovation*, 15(2):145–168.

[West and Sims, 2018] West, J. and Sims, J. (2018). How firms leverage crowds and communities for open innovation. In Tucci, C. L., Afuah, A., and Viscusi, G., editors, *Creating and capturing value through crowdsourcing*, chapter 4, pages 58–96. Oxford University Press, Oxford.

[Willard et al., 2014] Willard, B., Kendall, G., Leung, P., Park, C., Rich, M., and Upward, A. (2014). Future fit business benchmark. *London, England: The Natural Step Canada and 3D Investment Foundation*.

[Wirtz et al., 2016] Wirtz, B. W., Pistoia, A., Ullrich, S., and Göttel, V. (2016). Business models: Origin, development and future research perspectives. *Long Range Planning*, 49(1):36 – 54.

[Wolber et al., 2015] Wolber, D., Abelson, H., and Friedman, M. (2015). Democratizing computing with app inventor. *GetMobile: Mobile Comp. and Comm.*, 18(4):53–58.

[Yamakami, 2011] Yamakami, T. (2011). The third generation of oss: A three-stage evolution from gift to commerce-economy. In *IFIP International Conference on Open Source Systems*, pages 368–378. Springer.

[Yamashita et al., 2015] Yamashita, K., McIntosh, S., Kamei, Y., Hassan, A. E., and Ubayashi, N. (2015). Revisiting the applicability of the pareto principle to core development teams in open source software projects. In *Proceedings of the 14th International Workshop on Principles of Software Evolution*, pages 46–55. ACM.

[Ye and Kishida, 2003] Ye, Y. and Kishida, K. (2003). Toward an understanding of the motivation of open source software developers. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 419–429, Washington, DC, USA. IEEE Computer Society.

[Yin, 2009] Yin, R. (2009). *Case Study Research: Design and Methods*. Applied Social Research Methods. SAGE Publications.

[Yin, 2017] Yin, R. K. (2017). *Case Study Research and Applications: Design and Methods*. SAGE Publications, Inc.

[Yin, 2018] Yin, R. K. (2018). *Case study research and applications: design and methods*. Sage, 6th edition.

[Yun et al., 2016a] Yun, J. J., Won, D., and Park, K. (2016a). Dynamics from open innovation to evolutionary change. *Journal of Open Innovation: Technology, Market, and Complexity*, 2(1):7.

[Yun et al., 2016b] Yun, J. J., Yang, J., and Park, K. (2016b). Open innovation to business model: New perspective to connect between technology and market. *Science, Technology and Society*, 21(3):324–348.

[Zolnowski et al., 2014] Zolnowski, A., Weiß, C., and Bohmann, T. (2014). Representing service business models with the service business model canvas–the case of a mobile payment service in the retail industry. In *system sciences (HICSS), 2014 47th Hawaii International Conference on*, pages 718–727. IEEE.

[Zott et al., 2011] Zott, C., Amit, R., and Massa, L. (2011). The business model: recent developments and future research. *Journal of management*, 37(4):1019–1042.

Appendix A.

Publications of the Author

Müller, M., Schindler, C., Luhana, K. K., & Slany, W. (2018, November). Enabling Teenagers to Create and Share Apps. In *2018 IEEE Conference on Open Systems (ICOS)* (pp. 25-30). IEEE.
Christian Schindler and I co-lead the writing of all parts of the paper. I focussed on the theory of visual programming and the analysis of the community (mobile users and uploaded projects to the community platform).

Müller, M. (2018, November). Agile challenges and chances for open source: lessons learned from managing a FLOSS project. In *2018 IEEE Conference on Open Systems (ICOS)* (pp. 1-6). IEEE.
Sole Author of the paper.

Slany, W., Luhana, K. K., Müller, M., Schindler, C., & Spieler, B. (2018). Rock Bottom, the World, the Sky: Catrobat, an Extremely Large-scale and Long-term Visual Coding Project Relying Purely on Smartphones. In V. Dagienė, & E. Jasutė (Eds.), *Constructionism 2018, Vilnius: Constructionism, Computational Thinking and Educational Innovation* (pp. 104-119).
I contributed minor parts to the paper (Sub-section about Catrobat's development and contributors).

Vorraber, W., & Müller, M. (2019). A Networked Analysis and Engineering Framework for New Business Models. *Sustainability*, 11(21), 6018.
I contributed the case study (incl. theory therefore) and parts of the general theory (definition of NBMs).

Vorraber, W., Müller, M., Voessner, S., & Slany, W. (2019). Analyzing and Managing Complex Software Ecosystems: A Framework to Understand Value in Information Systems. *IEEE Software*, 36(3), 55-60.
I contributed to the theory of Open Source Systems and the case study.

Müller, M., Vorraber, W., & Slany, W. (2019). Open principles in new business models for information systems. *Journal of Open Innovation: Technology, Market, and Complexity*, 5(1), 6.
I was lead-author of this paper, writing the theory of Open Innovation, Open Source Systems and their business models, as well as analyzing and writing the main parts of the conceptual approach of seeing OSS as a NBM.

Müller, M., Schindler, C., & Slany, W. (2019, January). Engaging Students in Open Source: Establishing FOSS Development at a University. In *Proceedings of the 52nd Annual Hawaii International Conference on System Sciences* (pp. 7721-7730).
I was lead-author of this paper, describing the project's setting (contribution model and roles) and also conducting and evaluating the survey and its implications (students' involvement and motivation).

Müller, M., Schindler, C., & Slany, W. (2019, May). Pocket Code-A Mobile Visual Programming Framework for App Development. In *2019 IEEE/ACM 6th International Conference on Mobile Software Engineering and Systems (MOBILESoft)* (pp. 140-143). IEEE.
Christian Schindler and I co-lead the writing of all parts of the paper. I focused on the use-cases and their possible application (extensions).

Müller, M., Schindler, C., & Slany, W. (2019, May). Introducing agile product owners in a floss project. In *IFIP International Conference on Open Source Systems* (pp. 38-43). Springer, Cham.
I was lead-author of this paper, drafting and describing the workflow, and the underlying motivation to introduce it.

Hirsch, T., Schindler, C., Müller, M., Schranz, T., & Slany, W. (2019, July). An Approach to Test Classification in Big Android Applications. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)* (pp. 300-308). IEEE.
I advised in software development methods and proof-read the paper.

Müller, M. (2019, August). Managing the open cathedral. In *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering* (pp. 1176-1179).
Sole Author of the paper.

Schranz, T., Schindler, C., Müller, M., & Slany, W. (2019, August). Contributors' impact on a FOSS project's quality. In *Proceedings of the 2nd ACM SIGSOFT International Workshop on Software Qualities and Their Dependencies* (pp. 35-38).
I advised in Open Source Software contribution and proof-read the paper.

Harty, J., & Müller, M. (2019, August). Better Android apps using Android vitals. In *Proceedings of the 3rd ACM SIGSOFT International Workshop on App Market Analytics* (pp. 26-32).
I contributed the case of "Pocket Paint" and its implications.

Schindler, C., & Müller, M. (2019, September). Gender gap? a snapshot of a bachelor computer science course at Graz University of Technology. In *Proceedings of the 13th European Conference on Software Architecture-Volume 2* (pp. 100-104).
I contributed to the data-analysis and proof-read the paper.

Müller, M., Vorraber, W., Herold, M., Schindler, C., Slany, W., & Tanaka, K. (2019, September). Streamlining value in a FOSS project. In *Proceedings of the 13th European Conference on Software Architecture-Volume 2* (pp. 231-234).
I was the lead-author of this paper, with a focus on the contribution and agile management sections.