How to Design a Crowdwork Platform

Johannes Schuster

Schaeffler Technologies AG & Co. KG, Industriestraße 1-3, 91074 Herzogenaurach, Germany
E-mail: johannes.schuster@schaeffler.com

Dominik Dellermann

University of Kassel, Pfannkuchstrasse 1, 34119 Kassel, Germany
E-mail: dellermann@uni-kassel.de

Nikolaus Lipusch

University of Kassel, Pfannkuchstrasse 1, 34119 Kassel, Germany
E-Mail: nikolaus.lipusch@uni-kassel.de

Thomas Kohler

Hawaii Pacific University, Honolulu, HI, USA
E-Mail: tkohler@hpu.edu

Introduction

The emergence of cloud platforms like Salesforce´s Force.com or Apple's iPhone operating system (iOS) substantially changed the way how value is created in the software industry. While the traditional approach of software engineering focused on the independent development of a monolithic product by an individual software vendor, new approaches of digital technology strongly rely on innovation from third-party developers (Ghazawneh & Henfridsson 2013). IT industry leaders like SAP or Salesforce provide a scalable cloud platform, i.e. an expandable resource base, which offers a large crowd of developers' important resources such as computing power and software databases to enable the development of applications that extend the basic functionality of the platform. Such cloud platforms create an ecosystem consisting of numerous participants like for instance the platform owner (e.g. Salesforce, SAP), third-party developers and end-users, which transact with each other to develop novel value propositions. Platform owners are increasingly engaging vibrant ecosystems around their platform to foster third-party innovation (Boudreau 2012). Digital technology therefore significantly challenges the logic of value creation and value capture in the software industry

(Bharadwaj et al. 2013). On such platforms, firms rely on external actors and their resources to drive an open approach to innovation (Chesbrough 2006).

One solution to manage this new and open logic of software engineering is crowdsourcing (Tsai et al. 2014). This approach is defined as the delegation of open tasks to “an undefined (…) network of people” (Howe 2006) using the “collective intelligence” to develop novel value propositions (Leimeister et al. 2009; Surowiecki 2005). Competitive designs like for instance programming contest (Lakhani et al. 2013) or the collaborative development of open source software (OSS) (Lerner & Tirole, 2002, 2005) are examples for the application of crowdsourcing in this domain.

Research in the field of R&D management (e.g. Afuah & Tucci 2012; Ebner et al. 2009) and practice have realized the growing importance of this approach for the future of software engineering. Crowdsourcing, open source development and other forms of open innovation received considerable research attention, which is mainly focusing on the design of competitions, supporting components or the motivation of participants and are frequently limited to the project level (e.g. Leimeister et al. 2009; Lakhani & Hippel 2003). However, one aspect that has not been considered by research is the design of crowdsourcing-based business models on a more holistic level that fosters software development by the crowd and creates mutual value for a platform owner and the crowd of developers. Therefore, there is a lack of understanding how an “organization is linked to external stakeholders, and how it engages in economic exchanges with them to create value for all exchange partners” (Zott and Amit 2007: 181). The business model concept consequently seems particularly suitable as a framework to strategically structure and analyze the novel approach of open and digital value creation (Osterwalder & Pigneur 2013).

To build vibrant platform ecosystems around their cloud platform, firm must shift their focus from developing and selling products to providing resources that support the crowd in the development of complementary applications (Boudreau 2012). Hence, only if the unique needs and motivations platform owners, app developers, and end-users must be aligned it is possible to create and capture mutual value. We therefore develop the overarching idea that the activities, costs, and rewards of the developer crowd should be integrated in the design of sustainable crowdsourcing-based business models for cloud platforms. Therefore, we utilize the concept of the business model (Osterwalder & Pigneur 2010) to identify design requirements for crowdsourcing-based business models that integrate both the perspective of the platform owner and the crowd.

For the exploratory purpose of our research, we applied a multiple case study research design and conducted 17 semi-structured interviews with platform owners and developers from eleven platforms to gain comprehensive insights. We therefore believe that the study provides a deeper understanding of how platform business models should be designed to foster crowdsourcing-based software development and create and capture mutual value for the platform owner and the crowd.

The paper is structured as follows: In the next section, we elaborate on third-party innovation on platforms as the new organizing logic of innovation and the concept of business models. In the next section, we provide a detailed explanation of the research methodology, the data collection, and data analysis processes. Next, we outline the findings of the multiple case studies. The paper concludes by highlighting the contributions of our study along with directions for future research.

2. Related Work

2.1 Crowdsourcing-based Software Development

Apart from open source projects like Linux or Mozilla, also software giants such as Microsoft apply crowdsourcing for software development. For instance, for Windows 8 development Microsoft launched blogs, crowdsourcing mobile devices for Windows 8, and rewarded security testing for up to US$100,000. From a technological perspective, the platform-as-a-service (PaaS) principle enables the establishment of a cloud environment, which facilitates the crowdsourcing process (Tsai et al. 2014). Platforms are a set of components, which functionality can be extended by third-party applications. Cloud platforms are therefore an expandable code base, which are provided by a focal platform owner and enable other actors to develop new software artefacts like applications that extend the basic functionality of the platform (Tiwana et al. 2010). Hence, the platform itself becomes the locus of innovation.

Platform owners are therefore increasingly engaged in facilitating vibrant ecosystems and fostering innovation on their digital platforms as the number of complementary applications increases not just the functionality but also the overall value of a platform (Ceccagnoli et al. 2012; Boudreau 2012). To shift design capability to third-party developers, platform owners offer resources that enable the crowd to join the ecosystem and to offer their products to end-users (Tiwana et al., 2010). Such platform boundary resources are technical resources like software development kit (SDK), APIs, repository of software development assets, as well as social boundary resources, such as monetary and non-monetary incentives or intellectual property (IP) rights (Ghazawneh & Henfridsson 2013). Apart from software development tools, platform owners offer support for community innovation by providing collaboration and communication tools, participant ranking and recommendation tools (Tsai et al. 2014). The combination of technological resources and social interaction infrastructure enables community based innovation (Fuller et al. 2008). Dependent on the intended organizational style, crowdsourcing for software development can be either competitive by the use of competition with rewards (e.g. Leimeister et al. 2009) or collaborative like in the case of open source communities (e.g. von Hippel & von Krogh 2003).

A crowdsourcing approach to software development enables firms to initiate new projects, identify talented developers, or conceptualize, create and certify novel products. On the other hand, the third-party developer is compensated by the sale of its applications offered on the platform owner´s marketplace, or various intrinsic and extrinsic rewards (Malone et al. 2009).

Therefore, it serves as an entirely novel blueprint for competition since the success no longer solely depends on the software artefact alone but rather on the focal company´s ability to orchestrate all heterogeneous parties involved in such ecosystems (Boudreau 2012). Firms can no longer solely rely on enhancing features and the quality of their products by focusing on their own innovation efforts, but have to design their business models to engage an active ecosystem around the platform to create value (Yoo et al. 2012).

2.2 The Business Model Concept

The concept of business model has gathered substantial attention from both academics and practitioners in recent years (Veit et al. 2014). In general, it describes the logic of a firm to create and capture value (Chesbrough 2007). Although there is no commonly accepted definition of the term business model, the idea of the business model has been examined through numerous frameworks (Morris et al., 2005). It can be generally viewed as a boundary-spanning concept providing a deep understanding of how an enterprise is embedded in interactions with its ecosystem (Zott et al. 2011) and how the strategy is interrelated with the business processes and the corresponding IT infrastructure to link value creation, delivery and the generation of revenue (Al-Debei & Avison 2010).

In particular, the business model concept provides a holistic approach toward describing how value is created for all engaged stakeholders, the allocation of activities among them and the role of information technology (Bharadwaj et al. 2013).

Although the business model as a unit of analysis is anchored on the level of one focal firm, it requires a design to enable this firm not only to enhance the creation of value but also to distribute a share of the value created among the participants (Veit et al. 2014).

As a the emergence of software platforms and the crowdsourcing approach trigger fundamental changes in the way business is carried out and revenues are generated, the concept of business models help to analyze and understand a firm´s current business logic in this context. However, although previous research emphasized the role of a value network and partnerships in designing business models, they did not consider the explicit alignment of the platform owner´s business model and the crowd´s activities, which is required to create mutual value.

2.3 Components of a Business Model

The concept of the business model are therefore a useful approach structure, analyze and design the novel organizing logic of crowdsourcing for software development (Osterwalder & Pigneur 2013), which is from a technological perspective enabled by cloud computing. Although, there is no commonly accepted business model framework, several studies attempt to identify attributes and components a generic business model consists of (e.g. Morris 2005). One of the maybe most prominent representatives of single business model components is Osterwalder and Pigneur's (2010) Business Model Canvas. Based on a business model ontology that includes value creation and value capture mechanisms, such a business model constitutes nine building blocks (see Figure 1). We therefore followed this meta-model to structure our research.

	Pillar	Business Model Component	Description
Value Creation	**Core Value**	***Value Proposition***	Bundle of products and services that are of value for the customer
	Customer Interface	***Customer Segments***	Segment of customers a company wants to offer the value proposition
		Channels	The way a company communicates interacts with its customers to deliver a value
		Relationship	The links a company establishes between itself and the customer
	Infrastructure Management	***Key Partners***	Cooperative agreements with other parties to create value for the customer
		Key Resources	Important assets that are required to make a business model work and create and capture value
		Key Activities	Activities that are necessary to create value
Value Capture	**Financial Model**	***Cost Structure***	Amount of costs employed to run the business model
		Revenue Model	Logic of how a company makes money through a variety of revenue streams

Figure 1: Business Model Building Blocks (adapted from Osterwalder & Pigneur 2013)

3. Methodology

As the field of business model research in general and on crowdsourcing-based business models in specific lacks theoretical development, we choose an inductive case, study-based approach as the most appropriate methodology for our study (Eisenhardt 1989). The case study is an empirical approach that enables to investigate a current phenomenon within its real-life context, especially when the boundaries between the observed phenomenon and its context are not clear (Yin 2013). This methodology is particularly useful for developing theoretical insights from analyzing complex elements and interactions in an organizational context (Eisenhardt & Graebner 2007). We selected a multiple case study approach as it enables the researcher to compare data. Multiple cases are most appropriate where each case is a separate experiment (Yin 2013). Thus, we provide a more valuable extension of theoretical perspectives and a higher level of generalizability. Evidence from multiple cases is typically considered to be more compelling (Strauss & Corbin 2008; Yin 2013).

3.1. Sampling approach

To analyze the business model and its components of open source platforms, we analyzed eleven cases. The selection of cases in our study relies on the concept of theoretical sampling rather than random sampling (Eisenhardt & Graebner 2007). To start our qualitative research, we investigated an initial set of more than 100 open source platforms. For each firm, we analyzed the value creation and value capture mechanisms by studying the platform and available publications. After that, we narrowed down this initial selection to 11 cases for an in-depth analysis. To ensure rigorous case research (Yin 2013) we applied several case selection criteria.

First, we chose cases that have open source platforms at their core and built as a standalone business model rather than time-limited and isolated open source projects. Secondly, platforms were selected which have some solid traction, that was identified by key metrics around the number or value that is created on the platform. We therefore focused on the number of third-party developers building applications on the platform, the number of app downloads and the total value generated on marketplaces. Thirdly, we are solely focusing on platforms with significant impact, which was measured by number of apps created by developers. Forth, we were solely focusing on open source platforms that enable third-party developers to create value added products having a commercial intention.

Nr	Name of Company	Description	Position Manager	# of Participants
1	*HabitatMap*	HabitatMap is a non-profit environmental health justice organization with an open source platform for quoting, sharing and mapping health data	Executive Director	-
2	*Kaltura*	Kaltura is an open source online video platform, offering enterprise level commercial software and services as well as free open source community supported solutions	Senior Director of Community & Eco-sytem Partnerships	2
3	*Keen IO*	Keen IO makes analytics APIs that allow developers to build analytics applications upon	Community Manager	-
4	*Leapmotion*	Leapmotion builds a small USB peripheral that can plug into a Mac, Pc, small development boards or can be embedded in other products	Director of Developer Programs	-
5	*Mozilla*	The Mozilla Foundation is a non-profit organization that has been founded to maintain and cooperatively steer the Mozilla open source projects	Program Manager of the Global Community Support	-
6	*Pimcore*	Pimcore is an open source web content management platform for generating and managing web applications and digital attendances	CEO	-
7	*Red Hat*	Red Hat is the world's leading provider of open source solutions, using a community-powered approach to provide reliable and high-performing solutions	Community Architecture & Leadership	-
8	*Salesforce*	Salesforce is a global cloud computing company. It offers a platform where community members can use tools to create applications and sell them via AppExchange	Developer Program Manager	1
9	*Software Company*	Software company supports the open source community via evangelists and developer support that actively participate in the open source projects	Senior Director, Open Source Solutions	2
10	*Twilio*	Twilio allows software developers to programmatically make and receive phone and send and receive text messages using its web service APIs	Developer Evangelist	1
11	*WSO2*	WSO2 is an open source middleware application development software company specialized on offering service-oriented architecture solutions for professional developers	Vice President of Technology Evangelism	-
Sum of conducted interviews			11	6

Figure 2: Sample Description

3.2. Data collection

The main source of data collection in this study was semi-structured in-depth interviews. During the period of November 2014 and February 2015 we conducted interviews with eleven key informants from all cases that are directly involved in the management of third-party developers. To gain holistic insights we complemented our data with interviews with six third-party developers. The interviews lasted between 30 and 60min and were conducted and recorded through video conferencing due to geographical limitations. All interviews were literally transcribed and reviewed by the interviewees to improve accuracy (Huber & Power 1985). Furthermore, triangulation of different data sources was used to reduce informant bias, subjective judgment of the researcher, and to increase construct validity (Gibbert et al. 2008). Therefore, we complemented our interviews through observations of the platform and publicly available articles and reports. All data were organized in a single research database using the Dedoose software for qualitative analysis.

3.3. Data Analysis

To take advantage of flexible data collection and the adjustments of our procedure during the data collection process (Eisenhardt 1989) we applied an overlapping data analysis and data collection. The entire data analysis was iterative by the application of new open codes for later interviews and re-coding with selective codes. We followed the three progressive degrees of analysis: open, selective and theoretical coding to gather advancing depth of analysis of the empirical data (Glaser & Strauss 1967). In the early stages of data collection our analysis was more open-ended as we were focusing on the development of concepts, properties, and the relations among them by open coding. The authors conducted the coding process together, which results in an initial sample of approximately 300 descriptive codes following an inductive approach. We compared all concepts by merging closely related codes to reduce overlap. In the comparison process, we identified common patterns and consolidated them step-by-step to more abstract concepts in form of thematic codes. These thematic codes were sufficiently generic to include all other concepts and appeared frequently in our interviews. In the next step, we applied selective coding to refine our conceptual constructs and identified interaction between the descriptive categories of step one. Finally, we used theoretical coding to explicitly determine relationships between the individual interpretive constructs. Therefore, we discussed the empirical findings of our study in the context of previous literature and integrated them with existing theories. The comparison of these results with literature is essential for theory building and allows a higher level of internal validity, wider generalization, and a higher conceptual level (Eisenhardt 1989).

4. Findings

4.1 Creating Value with the Crowd

Value Proposition

Our findings reveal that platform owners have to clarify their core value unit (CVU), and identify a clear value proposition to challenge the complexity of crowdsourcing platforms. The CVU of software platforms is typically the technological base (i.e. the cloud platform itself) that allows the development of complementary applications and services.

However, platform owners have to clearly distinguish their value proposition for third-party developers and the end user. To offer a convincing value proposition for developers, companies must provide a sophisticated technological functionality and solve various business problems for many firms and individual user simultaneously. Typically, software platforms allow a high degree of flexibility to be attractive to as many developers as possible. In addition, interacting on such platforms is beneficial for developers in terms of their businesses or careers. Being part of certain platforms implies certain advantage due to the brand name, which is providing business opportunities for participants that want to use the commercial capital of the platform owner´s brand for conducting business and reducing risk. The value of the overall ecosystem particularly fuels the value proposition. For instance, Leap Motion integrated its handset controller product into various digital devices and delights the developer crowd to build on top of the platform. Attracting the crowd to participate in the creation of value enables the platform owner to increase the overall value of the platform without making additional investments. Apart from that, one of the value propositions for the end user is that they can request products and services without possessing developer skills. Therefore, they just can use the services and available applications of the ecosystem on demand, like for instance on the AppExchange market place of global cloud computing companies like Salesforce. Additionally, clients obtain greater flexibility when using the products and services of open source platforms due to costs of switching from one certain technology to another.

Customer Segments

Identifying the involved actors on a software platform and their respective needs is one of the keys to designing successful business models. The customer segments of software platforms involve various heterogeneous parties with multifaceted interrelations. As the senior director of community and ecosystem partnerships at Kaltura, an open source video product platform highlights that, "[…] crowd, it can be defined in so many ways, right? We have end-user crowd, we have partners' crowd, we have um developers crowd […]". In essence, crowdsourcing platforms bring together three main actors: the focal firm, the creators (partners and third-party developers) and the end-users. The focal company takes the role of an interaction enabler among developers as creators and users as consumers of value. Third-party developers add value to the platform by developing complementary software artefacts. Such complementors range from individual developers to small companies to big agencies or corporations. Depending on the nature and domain of a software platform, the proportion of companies and independent developers can vary. For complex and highly commercial software platforms like WSO2, a vendor of open source middleware, mostly enterprises and their employees are involved in the development process. On the other hand, end users consume the created value by using the software artefacts and services, which are developed by the crowd. However, this separation of roles remains blurred as third-party developers that create applications might also be users of other applications on the platform, since companies and developers are using the platform for themselves and their end-customer correspondingly. Software platforms therefore provide a clear overlap of value creators and consumers.

Channels

Platform owners provide a digital infrastructure that serves as an interface for interaction between the focal company and the developer crowd. To maintain the relationships with both third-party developers and end-users, crowdsourcing-based business models use multiple interaction points of different online and offline channels to directly connect with developers and to link them with each other.

Scalable channels to engage the crowd are collaboration and communication tools that enable the exchange among participants. For instance, Mozilla uses online sub-communities for empowering developers to create small groups for topics or geographic regions. Platform owners should allow these communities to function independently and without deploying external control. Other important channels that facilitate communication among developers are forums, either internal or external ones like GitHub or Stack Overflow.

Our research suggests that also offline channels like competition events and hackathons are helpful to attract initial creators to resolve the typical cold start problem of platforms. Kaltura, for example, encourages developers, sharing the same motivation and goals, to develop collaboratively on top of their platform through hackathons and therefore fosters active communities of developers.

Customer Relationships

Crowdsourcing-based software platforms should clarify the types of relationships they want to establish with developers and end-users. A main part of customer relationships is based on designing mechanisms to attract and engage the crowd. The success of crowdsourcing-based business models is highly dependent on the capability of the platform to attract a large, diversified and highly capable developer community. Our results reveal that software platforms are characterized by collaborative competition among participating developers. Hence, on the one hand, the developers are competing against each other to develop the most promising services and applications on top of the platform. On the other hand, the developers interact and procure each other to mutually improve their own applications and the platform. In complex environments like platform ecosystems "[…] not everybody can be an expert in every layer of the stack and so I think to be able to have the interaction with people who are experts in the area that they don't know allows you to grow and actually be better at the piece that you are responsible for […]" as a third-party developer participating on a platform stated. The platform owner should therefore focus on fostering interaction among the developers and balance collaboration and competition to facilitate the crowd of developers.

After initially attracting a certain amount of developers, it is crucial to engage the crowd to work on the software platform on a continuous level focusing on the fact that "[…] the developer is their core demographic […]" as a Developer Evangelist at Twilio, a cloud communications company, highlights. Companies like Kaltura or Salesforce empower top creators of the community to successively increase their responsibilities and take advantage of their developers´ passion for teaching and helping their peers. Red Hat, for instance, has an award program, where they are trying to recognize and award the people who are contributing to the platform. Furthermore, companies like Mozilla provide a badged system where members can earn ranks, depending on their contributions. Pimcore, an open-source web content management platform, and Twilio, for instance, display certain developers on their website, where they "[…] can present their use cases and their showcases […]", as the CEO of Pimcore explained. This is crucial as the loss of attractiveness is one of the most severe risk for any crowdsourcing-based software platform.

Key Partners

For software platforms, collaboration with partners in the ecosystem like customers or firms that provide complementary products and services becomes a pivotal strategy to extend their business and create value. As a developer at Kaltura described, "[…] we are always looking at strategic partnership to work with parties that are also heavily involved with open source […]".

During our research, it became evident that the main driver of actively searching for partners is to strengthen the market position. Thus, the range of companies´ partners is quite broad. Partnerships can emerge with technology or solution partners, intermediaries providing distribution channels (e.g. marketplaces), hardware providers, integration partners or even consultancies.

For instance, system integrators are using the technological foundation to customize and integrate it for end-users. Furthermore, software consultancies provide trainings and other complementors offer additional services based on a certain software solution. By building partnerships with device manufactures, companies like Leap Motion extend the value of their core platform. The acquisition, development and cooperation with such partners are steadily dynamic. Hence, special attention is required during all stages of collaboration with the partner ecosystem and the key activities it conducts.

Company Key Activities

The key activities of the focal firm span the creation, curation and consumption of the core value unit provided by the software platform. Therefore, platform owners must connect companies, individual developers, and end-users to enable interactions on the platform. The activities of creating value are shared between the focal firm and the crowd. Thus, it is crucial for companies to decide upon the activities that they dedicate to the crowd. For software platforms, the key activity is to enable and support crowd development by orchestrating the activities of the community. The vice president of technology evangelism at WSO2 indicated that, "[…] you have to make it developer friendly, you have to make it architect friendly, you have to show that someone can create halo API or halo service in 15 minutes […]". On the technological level, the firm offers its developer community boundary resources like application programming interfaces (APIs) and/or a software development kit (SDK) to nurture external developers to build on top of the platform. The platform owner develops technological standards and provides the core technology respectively making the software platform itself the core value unit. The source code is open to enable contribution to the extension of the platform´s functionality. At the business level, platform owners orchestrate developers to create value for end-users. For instance, Salesforce made the shift from a single software solution to a cloud platform by interlinking other software applications on top of its own code base. Thus, the platform owner focuses its activities on enabling the development of complementary applications by the crowd. Therefore, they design the crowdsourcing process by the definition of tasks and support developers to create on the platform.

Crowd Key Activities

The crowd's key activities on software platforms can be distinguished between creating, curating, and consuming the value. A crucial activity of the crowd is the extension of the technological core. The crowd develops on top of the platform, helps to maintain the technological functionality of the platform, tests new features, and actively shapes development decisions. Consequently, the overall attractiveness of software platforms is increased by the various complementary applications that extend the functionality of the platform. To create such applications the third-party developers, interact with the platform owner, other developers, and end-users. Thereby, they use the software platform "[…] as its back-end technology and then building all of the very stripe and payment specific analytics on top of their APIs […]", as a Community Manager at Keen IO, a provider of analytics APIs, highlights.

Apart from developing complementary software artefacts, the crowd performs several activities to curate the overall ecosystem around the platform. First, the crowd governs itself depending on the degree of empowerment the platform leader is granting. Second, developers are promoting and maintaining the platform, contributing to the code base, and providing support to each other. We identified these social aspects among the participants as one of the crucial success factors of software platforms. Third, the community can participate in the business development of the crowdsourcing-based firm, like in the case of Kaltura, that fosters the integration with other frameworks and platforms to enable plug and play.

Finally, the crowd can also consume the created value. This means using the developed applications and services provided by independent developers or other businesses by occupying the role of end-users.

Key Resources

According to our research, the most valuable resource for all crowdsourcing-based business models are the developer crowd itself, the technological platform, and the commercial capital of the platform owner´s brand.

First, the community that the company builds around its platform as well as the knowledge accumulated during exchanges within the network provides valuable social capital for both the platform owner and developers. Our interview partners stated: "[…] the crowd is not an opportunity for outsourcing labor to an external entity but rather the most crucial resource and an integral component of the platform business […]". Leap Motion, for example, uses its crowd to translate and spread documentation of knowledge around the world. Second, a further resource that is crucial for the crowdsourcing-based firm is the technological platform and the related boundary resources (e.g. APIs & SDKs) that enable the crowd to develop applications. Third, the commercial capital of the platform owner constitutes a key resource to create and capture value. A strong brand value that creates trust creates traction among developers and nurtures the community. Furthermore, the access to distribution networks is an important resource for the platform owner, which can be offered to the crowd to overcome limited abilities to invest in marketing capabilities.

Consequently, in the context of crowdsourcing-based business model's key resources are not only assets that enable the platform owner to create and capture directly but also resources that the focal firm can offer the crowd for their activities in creating value.

4.2 Capturing value with the crowd

When the company and the community create value, the question arises how the platform owner can capture some of the value as revenue. The traditional logic of business models of charging the customer for the value created is less likely to be a viable strategy for product platforms. Instead, product platforms call for different revenue mechanisms. They are dependent on the position the company takes within the value network and the question of whom to charge the collaboratively created value to ensure fairness. Hence, platform owners must find feasible as well as equitable pathways of sharing the created value and in reverse they must reward the developer community for participating in the creation of value. The design of a justifiable value capture model is crucial to build sustainable and effective crowdsourcing-based business models.

Company Cost Structure

In general, the outsourcing of development activities to the crowd reduces the operating costs of firms. However, such crowdsourcing activities strongly reshape the cost structure of the platform owner.

The primarily cost components affecting companies which apply a crowdsourcing-based business model are the expenses of creating, enhancing, and maintaining the technological platform core. This task contains several aspects like product and technology development or marketing activities to attract the crowd. An employee at Schuberg Philis, a software company, explained "[…] you have to decide how much investment a commercial company is going to go to a particular set of platforms […]. The cost structure of software platform business models is therefore rather shaped by providing digital infrastructure which enables development of software applications and enables interactions among the crowd than traditional investments in software engineering.

Crowd Costs

While the crowd of developers reduces the costs for the platform owner, it also should contribute investments like giving up intellectual property, scarifying time, and effort, sharing resources and sometimes even money to participate in product platforms. The actual cost structure of third-party developers therefore represents a crucial variable that should be acknowledged for designing crowdsourcing-based business models.

Depending on the license model of the platforms, developers have limited intellectual property and usage rights of the crowdsourced applications. For instance, Kaltura utilizes a general public license (GPL), obligating developers to deallocate their modifications while Keen IO applies the apache license, which permits a higher degree of freedom for developers, because only modifications to the core source code should be shared back.

As a participant at the Kaltura platform explained, another, rather more obvious issue that community members should invest when participating in product platforms is time and effort: "Instead of paying monetary values for a product you actually pay with your time." Due to the complexity of software platforms one should invest high amounts of energy to be acquainted with the systems and open source projects that are requiring "[…] cross technology expertise […]".

Nevertheless, in some cases direct monetary investments by third-party developers are required. For instance, on the Salesforce market place called AppExchange, the crowd must pay for screening and selection procedures of their applications. For instance, security checks of their developed software artefacts are compulsory if they choose to build an app upon the platform.

Revenue Model

Platform owners can rely on several ways to generate revenue streams. The most prominent paths include product sales, licensing as well as API usage fees or selling complementary services. Apart from this, the emergence of cloud computing allows the offering of the whole platform as PaaS (platform as a service). For instance, Twilio builds on an on-demand model to make developers pay per use for software development tools, collaboration or even project management tools. Furthermore, software platforms can generate revenue streams by selling co-created products on their market places like in the case of Leap Motion that applies an "[…] app store model [..]" and charges "[…] customers for additional services on top which are not open source […]".

Many software platforms make users of protected intellectual property paying licensing fees for permission. For instance, HabitatMap, a provider of a health data platform or Red Hat, the world's leading provider of open source solutions, applies this concept to generate revenue for the company.

Furthermore, software platforms that assist the developer community with their APIs charge a usage fee for offering that service. A platform may for instance employ a free test version and charge for an extended professional account. Finally, firms capitalize on offering complementary services, like consulting or support.

Crowd Rewards

Implementing a reward model for the crowd in addition to the revenue streams of the firm is crucial for the platform's success. If companies are solely able to co-create but not to co-capture value with the crowd, they will struggle and eventually fail. The most prominent reasons for failure are losing trust and a sense of unfairness.

Platform owners have to understand the complex, multi-faceted, and context specific motives of developers participating in product platforms and the corresponding incentives.

Consistent with social exchange theory, (Blau, 1964) we assume that developers generally will participate in community development when their rewards outweigh their costs. Thus, platforms rely on different sources of value to incentivize and engage the crowd. We structure our findings from the cases on ways to reward the developer community according to the taxonomy of money, glory, and love suggested by Malone et al. (2009).

First, when the crowd significantly contributes to the creation of value and the platform owner financially benefits from such co-creation, third-party developers are likely to rely on money as compensation for their efforts. Therefore, revenue sharing is a valid approach for platforms to foster engagement in value creation. For instance, Leap Motion enables developers to sell apps in a store and shares the revenue by a predefined "[…] standard 70/30 model split […]". Contrary to software platforms that apply competitions with price money as reward, it is crucial to consider the awareness for unfair revenue sharing that can otherwise lead to termination of participation and constitutes a severe risk for crowdsourcing-based business models.

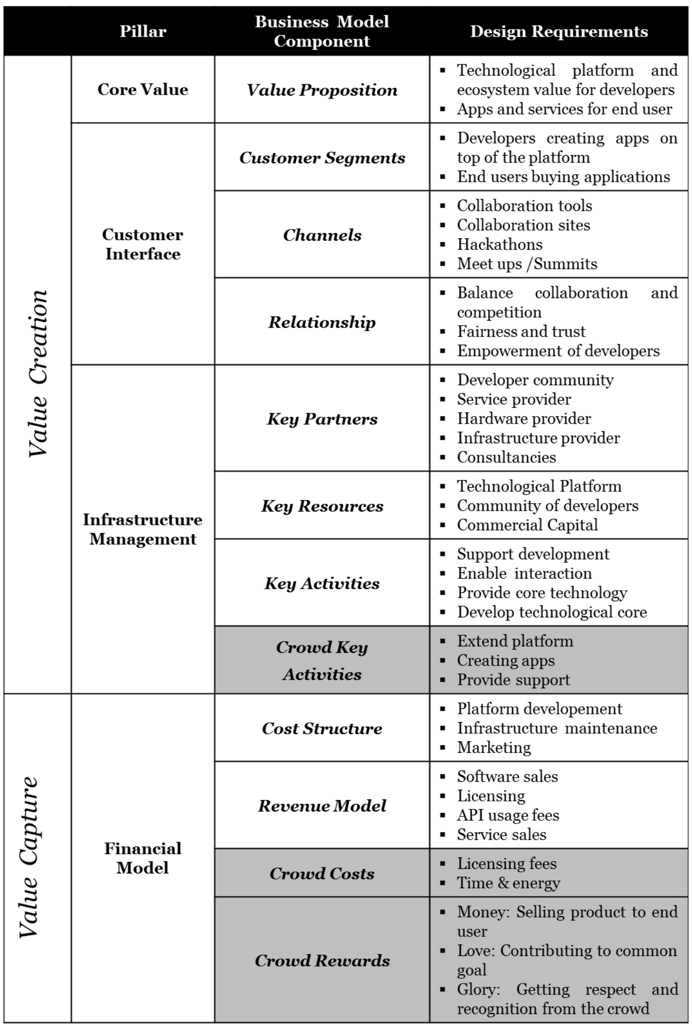
Second, glory is a key driver for many third-party developers creating software artefacts on platforms. In this case, the crowd seeks to get recognition for their individual efforts and skills. As a developer program manager at Salesforce stated, "[…] everyone that we see active in the community, I think what really drives them is getting that community recognition […]". Recognition needs to come from both the platform owner as well as from other members of the developer community. Successfully engaging developers in the creation of value is hence highly dependent on the user´s perception of glory and honor by the platform owner. Effective platforms treat the crowd not as an advantage for reducing operating costs but rather as a well-respected and equal partner. A reliable mechanism for providing appreciation of the communities´ effort is to enable visibility for their contributions such as performance-based memberships or top contributor lists.

Third, also on commercial software platforms a huge amount of developers is motivated by solely intrinsic factors that can be subsumed as love. Such incentives include for instance joy for an activity, contributing to a big picture or social aspects. In this case, the rewards for the crowd are having fun, learning and a sense of belonging. Platform owners therefore have to provide systems to connect the developers. As the community manager of Keen IO explains "[...] creating places for those developers to talk and making sure that we are supporting them and recognizing their efforts in the right ways [...]" is crucial for platforms to enable the exchange within the community as well as the required support for learning. Finally, our findings reveal that crowdsourcing-based business models have to be designed to ensure that benefits outweigh the costs of the crowd and translate their motives for participating into distinctive rewards. The platform owner therefore has to provide revenue streams in terms of rewards for third-party developers to design a sustainable business model.

5. Discussion

The results drawn from the cases reveal design requirements for integrating open innovation in general and crowdsourcing in particular approach in the business model perspective. What becomes obvious is the dual role of platform owners. Apart from traditionally offering software to an end user, platform provider´s role shifts towards enabling the development of complementary applications by third-party developers and the orchestration and engagement of the crowd. This fact results in a multiplicity of value proposition, heterogeneous customer segments and an increasing complexity of customer relationships and channels. To create successful business models, platforms have to gain solid traction among developers to be attractive. Furthermore, it is crucial to include the crowd not only in the creation but also in the capturing of value.

Apart from the contextual design requirements we are therefore able to identify three novel dimensions that extend the business model framework of Osterwalder and Pigneur's (2010): crowd activities, crowd costs and crowd rewards (see Figure 3). Our results show that the explicit interaction of the platform owner´s business model with the crowd is required to succeed. The activities of third-party developers have to be aligned and integrated into the platform owner´s business model. Furthermore, the crowd is only willing to engage if their rewards outweigh their costs. Therefore, platform owners should acknowledge the financial model of third-party developers to design sustainable business models.



	Pillar	Business Model Component	Design Requirements
Value Creation	**Core Value**	***Value Proposition***	▪ Technological platform and ecosystem value for developers ▪ Apps and services for end user
	Customer Interface	***Customer Segments***	▪ Developers creating apps on top of the platform ▪ End users buying applications
		Channels	▪ Collaboration tools ▪ Collaboration sites ▪ Hackathons ▪ Meet ups /Summits
		Relationship	▪ Balance collaboration and competition ▪ Fairness and trust ▪ Empowerment of developers
	Infrastructure Management	***Key Partners***	▪ Developer community ▪ Service provider ▪ Hardware provider ▪ Infrastructure provider ▪ Consultancies
		Key Resources	▪ Technological Platform ▪ Community of developers ▪ Commercial Capital
		Key Activities	▪ Support development ▪ Enable interaction ▪ Provide core technology ▪ Develop technological core
		Crowd Key Activities	▪ Extend platform ▪ Creating apps ▪ Provide support
Value Capture	**Financial Model**	***Cost Structure***	▪ Platform developement ▪ Infrastructure maintenance ▪ Marketing
		Revenue Model	▪ Software sales ▪ Licensing ▪ API usage fees ▪ Service sales
		Crowd Costs	▪ Licensing fees ▪ Time & energy
		Crowd Rewards	▪ Money: Selling product to end user ▪ Love: Contributing to common goal ▪ Glory: Getting respect and recognition from the crowd

Figure 3: Summary of Empirical Findings

6. Conclusion

The emerging relevance of open approaches to the development of software artefacts shifts platforms increasingly in the center of innovation management and research. Crowdsourcing turned out to be a useful and promising approach for designing business models for software platforms in the future of digital businesses. Our primary objective in this paper was therefore to analyze how platform business models should be designed to foster crowdsourcing-based software development and create and capture mutual value for the platform owner and the crowd. The results of our multiple case study analysis provide several interesting insights for both theory and practice.

First, our research contributes to the stream on open innovation (e.g. Chesbrough 2006) in general and crowdsourcing (e.g. Majchrzak & Malhotra 2013) by providing a business model perspective. Therefore, our study helps to view the design requirements for open innovation and crowdsourcing activities on a more holistic level and provides a detailed examination of value creation and value capture from such initiatives.

Second, we contribution to research on IT enabled and driven business models (e.g. Veit et al. 2014) by explicitly focusing on the dual role of platform owners in providing and maintaining technology as well as enabling interaction among the crowd. This study broadens the current state of research in crowdsourcing-based business models due to its uniqueness in the attempt to evaluate the degree of distributing a portion of the captured value back to the crowd. Furthermore, we shed light on integrating the exchange between the focal platform owner and the third-party developers. We therefore introduce three new components for a conceptual level meta model (e.g. Osterwalder et al. 2005): crowd activities, crowd rewards and crow costs. These new components ensure the sustainable design of business models that include the developer crowd in the creation and the capturing of value.

For managers, crowdsourcing raises a new set of strategic choices for mangers related to how to create and capture value. By studying successful platform strategies, managers can find ways that work to innovate, flourish and endure their business model when implementing the crowd in the creation and capture of value. The empirical study emphasizes numerous aspects, summarized in the following activity framework, which help managers to create better business models based on crowdsourcing:

- Technology asset management
- Orchestration of the platform ecosystem
- Providing social infrastructure
- Integration of the crowd in value capture mechanisms

Executives should solve the trade-off between empowerment and control, flexibility, and usability of the platform as well as the collaborative competition to create mutual value for both the platform owner and the crowd.

References

Afuah, A. and Tucci, C.L. (2012) Crowdsourcing as a solution to distant search. *Academy of Management Review,* **37**, 3, 355–375.

Al-Debei, M.M. and Avison, D. (2010) Developing a unified framework of the business model concept. *European Journal of Information Systems,* **19**, 3, 359–376.

Bharadwaj, A., El Sawy, O.A., Pavlou, P.A., and Venkatraman, N.V. (2013) Digital business strategy: toward a next generation of insights. *Mis Quarterly,* **37**, 2, 471–482.

Boudreau, K.J. (2012) Let a thousand flowers bloom? An early look at large numbers of software app developers and patterns of innovation. *Organization Science,* **23**, 5, 1409–1427.

Ceccagnoli, M., Forman, C., Huang, P., and Wu, D.J. (2012) COCREATION OF VALUE IN A PLATFORM ECOSYSTEM: THE CASE OF ENTERPRISE SOFTWARE. *Mis Quarterly,* **36**, 1, 263–290.

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

Chesbrough, H. (2007) Business model innovation: it's not just about technology anymore. *Strategy & leadership,* **35**, 6, 12–17.

Ebner, W., Leimeister, J.M., and Krcmar, H. (2009) Community engineering for innovations: the ideas competition as a method to nurture a virtual community for innovations. *R&D Management,* **39**, 4, 342–356.

Eisenhardt, K.M. (1989) Building theories from case study research. *Academy of Management Review,* **14**, 4, 532–550.

Eisenhardt, K.M. and Graebner, M.E. (2007) Theory building from cases: Opportunities and challenges. *Academy of management journal,* **50**, 1, 25–32.

Füller, J., Matzler, K., and Hoppe, M. (2008) Brand community members as a source of innovation. *Journal of Product Innovation Management,* **25**, 6, 608–619.

Ghazawneh, A. and Henfridsson, O. (2013) Balancing platform control and external contribution in third-party development: the boundary resources model. *Information Systems Journal,* **23**, 2, 173–192.

Gibbert, M., Ruigrok, W., and Wicki, B. (2008) What passes as a rigorous case study? *Strategic management journal,* **29**, 13, 1465–1474.

Glaser, B.S. and Strauss, A. (1971) A.(1967). The discovery of grounded theory. *New york.*

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

Howe, J. (2006) The rise of crowdsourcing. *Wired magazine,* **14**, 6, 1–4.

Huber, G.P. and Power, D.J. (1985) Retrospective reports of strategic-level managers: Guidelines for increasing their accuracy. *Strategic management journal,* **6**, 2, 171–180.

Lakhani, K.R., Boudreau, K.J., Loh, P.-R., Backstrom, L., Baldwin, C., Lonstein, E., Lydon, M., MacCormack, A., Arnaout, R.A., and Guinan, E.C. (2013) Prize-based contests can provide solutions to computational biology problems. *Nature biotechnology,* **31**, 2, 108–111.

Lakhani, K.R. and Hippel, E. von (2003) How open source software works:"free" user-to-user assistance. *Research policy,* **32**, 6, 923–943.

Leimeister, J.M., Huber, M., Bretschneider, U., and Krcmar, H. (2009) Leveraging crowdsourcing: activation-supporting components for IT-based ideas competition. *Journal of management information systems,* **26**, 1, 197–224.

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

Lerner, J. and Tirole, J. (2005) The economics of technology sharing: Open source and beyond. *The Journal of Economic Perspectives,* **19**, 2, 99–120.

Majchrzak, A. and Malhotra, A. (2013) Towards an information systems perspective and research agenda on crowdsourcing for innovation. *The Journal of Strategic Information Systems,* **22**, 4, 257–268.

Malone, T.W., Laubacher, R., and Dellarocas, C. (2009) Harnessing Crowds: Mapping the Genome of Collective Intelligence.

Morris, M., Schindehutte, M., and Allen, J. (2005) The entrepreneur's business model: toward a unified perspective. *Journal of business research,* **58**, 6, 726–735.

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

Osterwalder, A. and Pigneur, Y. (2013) Designing business models and similar strategic objects: the contribution of IS. *Journal of the Association for Information Systems,* **14**, 5, 237.

Osterwalder, A., Pigneur, Y., and Tucci, C.L. (2005) Clarifying business models: Origins, present, and future of the concept. *Communications of the association for Information Systems,* **16**, 1, 1.

Strauss, A. and Corbin, J. (2008) *Basics of qualitative research*: Newbury Park, CA: Sage.

Surowiecki, J. (2005) *The wisdom of crowds*: Anchor.

Tiwana, A., Konsynski, B., and Bush, A.A. (2010) Research commentary-Platform evolution: Coevolution of platform architecture, governance, and environmental dynamics. *Information Systems Research,* **21**, 4, 675–687.

Tsai, W.-T., Wu, W., and Huhns, M.N. (2014) Cloud-Based Software Crowdsourcing. *IEEE Internet Computing,* **18**, 3.

Veit, D., Clemons, E., Benlian, A., Buxmann, P., Hess, T., Kundisch, D., Leimeister, J.M., Loos, P., and Spann, M. (2014) Business models. *Business & Information Systems Engineering,* **6**, 1, 45–53.

Yin, R.K. (2013) *Case study research: Design and methods*: Sage publications.

Yoo, Y., Boland Jr, R.J., Lyytinen, K., and Majchrzak, A. (2012) Organizing for innovation in the digitized world. *Organization Science,* **23**, 5, 1398–1408.

Zott, C. and Amit, R. (2008) The fit between product market strategy and business model: implications for firm performance. *Strategic management journal,* **29**, 1, 1–26.

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