Journal of the Association for Information Systems (2020) **21**(6), 1402-1460
doi: 10.17705/1jais.00642

RESEARCH ARTICLE

ISSN 1536-9323

An Information Systems Design Theory for Service Network Effects

Christian Janiesch[1], Christoph Rosenkranz[2], Ulrich Scholten[3]

[1]University of Würzburg, Germany, christian.janiesch@uni-wuerzburg.de
[2]University of Cologne, Germany, rosenkranz@wiso.uni-koeln.de
[3]VentureSkies, France, ulrich.scholten@scholten.aero

Abstract

Service platforms make software applications available as a service to end users. Platforms enable noticeable economic benefits for scaling and transforming a business. Their long-term competitiveness is ensured in controlled cooperation with channel intermediaries and network partners. Hence, service platforms must be designed to harness self-enforcing effects of value generation, so-called network effects. In an exaptation of existing knowledge, we present an information systems design theory to inform the design of methods that analyze, describe, and guide the design of service platforms through the means of causal loops and control methods. We describe the theory's purpose and scope as well as the underlying justificatory knowledge behind the constructs and principles of form and function. The design theory covers the design of all service platform participants and activities as well as their transactions and influences in areas of staged platform authority, using enforcing and incentivizing control methods. We demonstrate the principles of implementation with an expository instantiation and apply it to the M-Engineering service platform, which offers surveillance, control, and data acquisition solutions. Furthermore, we present and discuss testable propositions and a study design to evaluate our design principles.

Keywords: Information Systems Design Theory, Design Principles, Service Platform Design, Network Effects, Control Methods

Jan Pries-Heje was the accepting senior editor. This research article was submitted on September 28, 2016 and underwent four revisions.

1 Introduction

The stature and structure of software development and distribution have changed dramatically over the past decade, as many traditional software products have become software services. Rather than being installed on-premises, software as a service (SaaS) is internet-based and runs in the cloud and is thus accessible to end users as on-demand services anywhere in the world (Mell & Grance, 2011). A *service platform* is software that makes such deployed applications available as a service to end users while taking care of elasticity, metering, and billing (Barros & Dumas, 2006; Mell & Grance, 2011; Rimal, Choi & Lumb, 2009). Service platforms cater to multisided markets, help build vibrant ecosystems, and enable others to innovate building on the platform-controlled resources such as software development kits or application programming interfaces (Yoo, Henfridsson & Lyytinen, 2010). Technical factors that are often associated with and that drive the rapid growth of service platforms include shortened deployment times, reduced upfront implementation, and minimized long-term overheads, compared to traditional on-premises software (Holt et al., 2011). Examples of contemporary service platforms include classic SaaS platforms, such as those used by Salesforce and Dropbox as well as

LinkedIn or Facebook, and computing platforms such as Android or iOS. While we do not explicitly cover services such as Airbnb or Uber, this research also applies to their service platforms.

Because of the unprecedented growth of the platform business, in mid-2019, the top four public companies by market capitalization were, for the third year in a row, platform operators offering digital services. [1] Business models based on platforms allow companies to scale external resources and innovation to match the soaring needs of customers, while the companies' internal capacities can only be considered as static in comparison. In platform-based businesses, third-party developers develop applications, services, or systems for satisfying end users of the platform, on behalf of someone else, namely the platform owner (Ghazawneh & Henfridsson, 2013). The platform owner does not directly compensate the third-party developer but offers a marketplace with greater reach than would otherwise be available. By doing so, the platform owner taps into multiple networks of developers, characterized by heterogeneous innovation capability and knowledge resources, and thus can set up a revenue-sharing business model in which a specific portion of the revenue is withheld as compensation for the distribution and support of applications (Ghazawneh & Henfridsson, 2013, Yoo et al., 2010). Platforms involving third-party service providers enable noticeable economic advantages abandoning the traditional linear pipeline value chain involving gatekeepers (Parker & Van Alstyne, 2018, Van Alstyne, Parker & Choudary, 2017).

Accordingly, the long-term competitiveness of services can only be ensured in cooperation with channel intermediaries and network partners[2] (Vargo & Lusch, 2004) with the platform operator in charge of a microeconomy (Parker & Van Alstyne, 2018). As platforms bring together multiple user groups and third-party developers, they create so-called *network effects*, or network externalities (de Reuver, Sørensen & Basole, 2018). This active participation of consumers and external suppliers in the value creation process can additionally accelerate a product's success (Chesbrough, 2012; West et al., 2014). Network effects are self-enforcing effects of value generation, created by causal loops of reciprocal interdependency between platform attractiveness and third-party value provisioning (Rohlfs, 1974; Shapiro & Varian, 1998). However, after opening the platform to third-party service provisioning, the platform operators must give up control over their service quality to some degree and must accept a certain level of self-organization by service providers (Lee et al., 2010; Parker, Van Alstyne & Choudary, 2016). For example, at first, Apple only reluctantly opened the AppStore and thus iOS to external developers, giving up some autonomy over the platform's content (Isaacson, 2011). Similarly, Salesforce and Dropbox have enabled service providers to integrate third-party services into their respective platforms.

To profit from self-organization, however, it is crucial that platform operators manage and constrain relationships carefully by inciting and exploiting network effects to steer the flows of service provisioning and consumption effectively (Parker & Van Alstyne, 2018). They can do so by using platform governance and control mechanisms (Thies, Wessel & Benlian, 2018)—i.e., their *platform authority*. Platform authority describes the exercisable degree of control and influence a platform operator has over an ecosystem participant or activity. Platform operators can use multiple *control methods* to exercise their platform authority (Kirsch, 1997; Ouchi, 1979). Consider again the example of Apple's AppStore: Apple employees review all apps before they appear on the AppStore and the company takes a share of the earnings, while at the same time preventing application installations outside of the AppStore.[3]

The design of a service platform that allows for balancing fine-grained degrees of control and influence with the freedom to incite network effects, appears to be challenging. At this point, few platform operators that offer selective cross-industry applications have successfully mastered the development of a robust and durable service platform harnessing network effects (Van Alstyne et al., 2017). On the one hand, the platform operator must attribute to the members of its ecosystem certain rights to act and react in a self-paced way—for example, by granting them the right to develop or deploy services on the service platform. On the other hand, the platform operator must ensure that these services are attractive to other participants and comply with published terms and conditions as well as the service platform's business model. Coordinating these simultaneous interactions is a complex task that is not always successful. For example, Google Health[4] was a personal health record service. Its failure provides an insightful example of the platform operator failing to incite network effects. Google did not engage with doctors as service providers and was not able to partner with insurances and, thus, did not provide a base value for consumers (patients) to reach self-sustaining levels. We still lack theoretical knowledge on how to design

[1] *Global Finance Magazine*, https://www.gfmag.com/global-data/economic-data/largest-companies

[2] A more detailed distinction of roles in service delivery can be found, for example, in Barros and Kylau (2011).

[3] https://developer.apple.com/app-store/review/

[4] https://www.google.com/health (deactivated as of 2013)

service platforms for network effects (Abdelkafi et al., 2018; de Reuver et al., 2018) and how to manage these service network effects using one's own platform authority (Ondrus, Gannamaneni & Lyytinen, 2015).

At the core of this paper is an information systems design theory (ISDT) for service network effects, which we describe as a "systematic specification of design knowledge" (Gregor & Jones, 2007, p. 314). The artifact type of our research is a design method for service platforms to guide the implementation of service platforms for network effects. As network effects cannot truly be designed themselves, we consider our design principles to be propositions for a design method intended to design service platforms in such a way that positive networks effects are likely to emerge. In doing so, we relate our approach to design goal-oriented approaches summarized as *design for X* (Pahl et al., 2007).

Our ISDT will be helpful in analyzing, explaining, and designing for the phenomenon of network effects on service platforms by identifying the relevant constructs and their relationships and by applying controls (Baskerville & Pries-Heje, 2010). Moreover, it will help guide the (re)design of service platforms through explicit prescriptions. Yet this does not provide a ready-to-use design method. Hence, we use this ISDT to devise an expository instantiation that assists in outlining service platforms, allowing us to plan for the desired effects and instantiations of parameters for a successful balance between control and self-organization.

In the following, we present the purpose and scope of our research as well as the methodical framework. First, we scope our ISDT and explain the research methodology and the process we followed. We then detail the underlying kernel theories. Based on these, we introduce the necessary constructs, formulate eleven distinct design principles, and present our design rationale. Further, we describe an expository instantiation and its application. Finally, we present testable propositions to evaluate our design principles. We conclude with a discussion of the implications and limitations for research and practice.

2 Methodical Framework and Design Theory

We follow a design science research process to develop our ISDT. As our research progressed, we iteratively developed the configuration of the artifacts (Baskerville, Pries-Heje & Venable, 2009). We make use of kernel theories to derive an ISDT with design requirements and a design rationale underlying our approach (Simon, 1996; Walls, Widmeyer & El Sawy, 1992). See Appendix A for a summary of crucial steps in the design process.

2.1 Purpose and Scope of the IS Design Theory

According to Gregor (2006), theories in IS research include, among others, theories for design and action. They provide prescriptive rather than descriptive knowledge to shape the phenomena at hand through artificial artifacts. Following Gregor and Jones (2007), an ISDT consists of at least the following components: a *purpose and scope*, which explains the design's goals; its *constructs* encompassing all relevant entities; its *principles of form and function* as an abstract blueprint; *artifact mutability* to assure the theories robustness and flexibility; testable *propositions* for evaluation; and *justificatory knowledge*, which can be other theories as well as practical knowledge (Gregor & Hevner, 2013). The *principles of implementation* to convey how the artifact can be implemented and *expository instantiations* as a proof of concept are optional components.

The primary artifact of our research is an ISDT to inform *design methods for service network effects*. Through explicit methodological prescriptions in the form of design principles, this ISDT can be used to engineer design methods that describe, analyze, explain, and improve network effects using control methods on service platforms. Further, we provide network effects patterns as selected principles of implementation (see Appendix D). An expository instantiation of the ISDT in the form of a conceptual modeling language provides an illustration of a working design method (see Section 5 and Appendix C). This language adds the ability to visualize abstract theoretical constructs and their relationships graphically.

It must be noted that our ISDT does not explain why or if a service platform is or will be commercially successful since this depends on other factors such as, for example, subsidization, revenue sharing, and alliance strategies (Casey & Töyli, 2012; Wanner, Bauer & Janiesch, 2019). However, it does assist in the analysis of commercialization options by making causal loops and network effects visible and explicit.

Following the knowledge contribution framework of Gregor and Hevner (2013), we consider our ISDT to be an exaptation of network effects theory and control theory to the design of service platforms. It has explanatory power and provides design practice theory for the design and improvement of further artifacts that aim to incite and harness service network effects using control methods. We have formulated eleven design principles following the suggestion of Chandra, Seidel, and Gregor (2015) to include materiality as well as action, and we document the underlying design rationale. Their general boundary conditions have already been covered above; specific limitations are mentioned in the detailed description of each principle.

2.2 Research Methodology and Design Search Process

The design search process was an iterative process involving building artifacts and theories, intervention and learning, and enhancement, as described in Meinel and Leifer's (2010) "Design Thinking." Our process aligned with suggestions proposed in Sein et al.'s (2011) "Action Design Research" and was similar to the approach used by Gregory and Muntermann (2014) in their "Heuristic Theorizing."

We compiled the first version of our artifacts through data elicitation, surveys, and field studies. They evolved over a period of five years through a cyclical research process within a large research project. The continuous application of the method led to adaptations and enhancements and resulted in redesigns of the approach, a common occurrence in design science research (Davison, Martinsons & Kock, 2004; Vaishnavi & Kuechler, 2008). Among other things, this cyclic research process consisted in an analysis of successful service platforms over a period of four years, a comparative longitudinal study of service intermediaries and field studies on three service platforms, as well as an e-market pilot that we developed. More details are given in Appendix A. The preliminary results have also been presented in prior publications (May, Scholten & Fischer, 2011; Scholten, 2013; Scholten et al., 2011; Scholten, Fischer & Zirpins, 2009; Scholten, Janiesch & Rosenkranz, 2013; Scholten, Schuster & Tai, 2012).

Conducting these tests solely on our own created the risk of biased results (Zelkowitz & Wallace, 1997). Hence, we enhanced the results through an iterative process of discussion and reflection with researchers as well as with business analysts and platform architects. Our research built on and benefited from these continuous exchanges with industry. We conducted tests through modeling (intervention/action taking) of sample cases to reveal whether all known control methods are addressed within our ISDT. We conducted an assessment and reflection of the derived ISDT with four platform operators (two at intermediate and prefinal stages and with two at the end of their respective design processes). Furthermore, we formulated testable propositions about the design principles and proposed evaluating them empirically in a survey. In prior publications, we presented an initial categorization of control methods (Scholten et al., 2011; Scholten et al., 2012).

3 Related Work and Underlying Theories

We draw justificatory knowledge (i.e., kernel theories) from different areas, as our ISDT is an exaptation of existing theory. Hence, it synthesizes and applies theories from several domains. In particular, we based our ISDT on the theories of *system dynamics*, *network effects*, and *control theory*.

3.1 System Dynamics and Network Effects

In network theory, the term *network effect* describes the phenomenon that products or services become more valuable when large numbers of people use them (Rohlfs, 1974; Shapiro & Varian, 1998). Many digital goods are subject to network effects. For example, once Facebook increased in popularity, Facebook accounts became more useful and valuable since they could be used to connect to more people.

A network effect is comparable to reinforcing feedback in *system dynamics* (Forrester, 1961). System dynamics describes macroscopic system behavior over time, built through the interaction of *sources* and *sinks*, *stocks*, *flows*, and *feedback loops*, as described in systems theory. It is a means to frame, understand, and discuss intricate issues and problems in the simulation and engineering of complex systems (Dutta, Roy & Seetharaman, 2008; Gleich, Mosig & Reinwald, 2011; Schneider, Gschwendtner & Matthes, 2015).

Sources and sinks represent the origin and destination of any supply. In system dynamics, they are assumed to have infinite capacity, meaning they can never be fully depleted or filled. As a source, it causes an inflow into a system; as a sink, it is the destination of an outflow. Stocks describe the storage capability of a system. A stock can accumulate or deplete over time and provides a system with memory and inertia. Flows change stock over certain periods of time. They can behave as inflows, filling the stock, or outflows, emptying it. Flows can be positive or negative. Feedback loops describe the reciprocity that a stock has on its own filling or depletion. These concepts describe the elementary constructs of systems designed for network effects.

In the context of service platforms, the provision of information about activities on a platform to participants who are active on a platform is called feedback. The participants' reaction to this information causes reciprocity, meaning modified activity on the platform leading to renewed feedback. We refer to the feedback loops of auxiliary variables, which are functions of stocks and which impact flows as *causal loops*.

Further, we refer to the value propositions that are expected to incite network effects as the *base value*. Base values need to exceed a minimal threshold called *critical mass*. Critical mass theory explores the conditions, or the tipping point (Schelling, 1971) under which reciprocal behavior begins and becomes self-sustaining (Oliver, Marwell & Teixeira, 1985). These conditions are defined by (1) the relationship between

individuals' contributions of resources and achievement of the common good (production function), and (2) the heterogeneity of resources and interest in the population (Markus, 1987), thus interrogating (1) the level at which a group action (network effect) is likely to begin, and (2) the difference in the interest of the participants. For example, for a network effect to occur, what is the minimal quantity of subscribed consumers on a social network, and how likely is it that some users will have more interest and thus invest more?

The base value may take the form of a contribution from the platform operator—for example, Salesforce's CRM software. In this case, the value would be a nearly static stock because the limited team of contributors and content only grows organically. Other base values depend on activities from ecosystem partners, such as Salesforce's third-party service providers. If successfully implemented, these base values grow through the activated network effect beyond the static behavior restricted by one's own resources and provide scalability otherwise unachievable through mere automation.

The simplest form of a network effect is a direct effect where an increase in use leads to an increase in value (Katz & Shapiro, 1986). These effects are always same-sided. The telephone is a common example. Sterman (2000) shows that network effects can be strengthened or weakened by complementary network effects. Indirect network effects or market mediated effects may also manifest. They occur when the use of a good spawns complementary products or services that in turn increase the value of the original good (Economides & Salop, 1992). The iPhone is a good example of a valuable product that became more valuable because of complementary software services.

Rochet and Tirole (2003) as well as Parker and Van Alstyne (2005) reveal that, in many cases where indirect network effects occur, two or more distinct participant groups benefit from each other. The increase in use by one group increases the value of a complementary good of another group and vice versa resulting in two- or multisided markets or platforms (Eisenmann, Parker & Van Alstyne, 2006; Parker & Van Alstyne, 2005; Rysman, 2009). Two-sided markets may include various alternatives for causal loops. Those loops may be same-sided—for example, demand-sided or supply-sided. They may also be cross-sided, involving both sides of the platform. Eisenmann et al. (2008, 2011, 2006) speak of demand-sided network effects, finding that platforms that are open to external supply can encompass cross-sided network effects. In these effects, the supply side and demand side are interdependent.

Service platforms with consumers on the one side and service providers on the other comply with the definition of two-sided markets. The Salesforce service platform illustrates this cross-sided effect. With more consumers looking for offers, the attractiveness rises for third-party service providers to offer services. Salesforce complemented their same-sided marketplace with the development and deployment environment enabling external services to create cross-sided network effects. The size of Salesforce's consumer base had a direct effect on the suppliers' motivation to offer their own services, and the increasing amount of services offered increased the service platform's attractiveness to consumers. Further examples can be found in the relationship between hardware and software vendors, such as Microsoft and Intel (Wintel)[5], as well as in the relationship between marketplaces and service providers (Barros & Dumas, 2006).

Hence, in our context, service network effects describe the reciprocal relationship between the value of a service platform and the quantity of involved service consumers and service providers. These network effects are driven by self-organization of the platform ecosystem based on participants' autonomy, adaptability, and sensitivity to change (De Wolf & Holvoet, 2005; Nicolis & Prigogine, 1977) because third-party participants are not under the full control of the platform operator.

Consequently, (service) network effects themselves cannot be truly designed. Successful service platforms must be designed *for* network effects. Unsuccessful service platforms tend to not accomplish self-enforcing network effects (Van Alstyne et al., 2017).

3.2 Control Theory

Many studies that investigate control in the context of open technical platforms and the enabling cooperation of distinct supplier and user groups include network effects in their reasoning and conceptualizing (e.g., Boudreau, 2010; Hagiu & Lee, 2011; Katz & Shapiro, 1986; Parker & Van Alstyne, 2018; Schilling, 2009). In this context, control is considered from perspectives of power through technology ownership, technical evolution decisions, and distribution rights. Being in control includes the rights to appropriate value from a technology.

Closer to the systems-theoretical consideration of causal loops is the theory of feedback loop control (Ashby, 1964; Conant & Ashby, 1970). It describes the concept of a (technical) system being regulated by a control device aligning a reference value with a fed-back system output (see Figure 1).

[5] http://www.intelalliance.com/microsoft/ (now defunct)

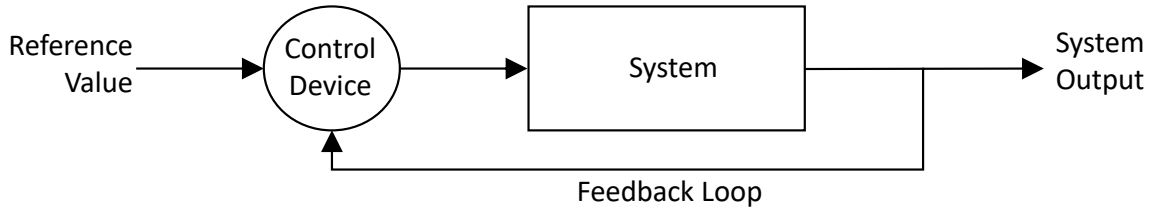


Figure 1. Feedback Controlled System

Table 1. Modes and Methods of Control based on Kirsch (1997)

Control mode	Key characteristics	Antecedent condition	Example mechanism	Group of method	Control method
Behavior (formal)	• Rules and procedures articulated • Rewards based on compliance with rules and procedures	Behavior observability	Development methodology, work assignments, rules and procedures	Enforcement	Prescriptive control, sanctional control
				Incentive	
Outcome (formal)	• Outcomes and goals articulated • Rewards based on producing outcome and goals	Outcome measurability	Comparison of outcome with the expected level of performance and successive rewards	Enforcement	Restrictive control
				Incentive	Motivational control
Self (informal)	• Individual sanctions him- or herself • Individual defines task goals or procedures, Individual monitors and rewards her or-himself; the rewards are based in parts on individuals' self-control skills	None	Individual empowerment, self-management and self-monitoring, and self-rewarding with respect to self-set goals	Enforcement	
				Incentive	Informative control, market regulative control
Community (informal)	• Identification and reinforcement of acceptable behaviors • Common values and beliefs and problem-solving philosophy	None	Coalitions of individuals with share ideologies, socialization, hiring and training practices, implemented rituals and ceremonies	Enforcement	
				Incentive	Market regulative control, informative control, motivational control

Transferred to our context of service platforms, *control* describes service management actions of the platform operator that change a set of parameters from the current status to a target status. Control in a platform context operates as a closed loop that uses feedback monitoring in the context of a regulatory process. The devices and methods, which are used to control such a process, can be attributed to different *control modes.*

Building on these insights, Kirsch (1997) developed a taxonomy of control modes. We have associated six abstract control methods with these four control modes. They can be either enforcing or incentivizing.

The set of Kirsch's formal control modes contains (1) *behavior control,* characterized through articulated rules and procedures, and (2) *outcome control*, defined by expressed project outcomes and goals. Formal control modes can be designed to be observable and are hence suitable in enforcement and reward-oriented approaches (Eisenhardt, 1985; Kirsch, 1997; Ouchi, 1979). Informal modes include (3) *self-control* and (4) *clan control*. Instead of clans, we use the term *community*, as this term is established in the service platform context. Self-control relies fully on an individual's ability and competence for self-control.

Community control modes are suitable where coalitions of individuals cluster around common values and beliefs (Kirsch, 1997). Informal modes lack observability and, hence, their successful implementation is difficult to observe. However, an organization can benefit from such interpersonal feedback-seeking dynamics in social structures that support the self-regulation of social processes (Ashford & Tsui, 1991). Table 1 summarizes Kirsch's four control modes, their key characteristics, antecedent conditions, as well as examples of individual control mechanisms. Building on this, we introduce six control methods for service platforms. We categorize the enforcing methods as *prescriptive control*, *sanctional control*, or *restrictive control*. Similarly, we categorize the incentivizing control methods as *market regulative control*, *informative control*, or *motivational control*. We develop these control methods further in the following section, as their discussion requires further constructs we have not yet introduced. Each method can make use of multiple mechanisms

4 Artifact Description

In this section, we present the principles of form and function incorporating the underlying constructs with the aim of developing an ISDT for a design method. For the sake of readability, we have not separated the description of these two principles. First, we introduce our fundamental assumptions: actors and process elements allow for the description, modeling, and analysis of causal loops and thus service network effects. Second, we introduce structural elements to embrace different areas of staged platform authority. Finally, we introduce control methods to support the implementation of service management. For all of the above, we formulate design principles as explicit prescriptions on how to improve service platform design for network effects. Each of the design principles is accompanied by design rationale argumentation explaining the reason and justification as well as alternatives and trade-offs considered (Lee, 1997). A tabular overview of the design rationale is available in Appendix B. The boundary condition for each design principle is "… given that it shall be used to design service platforms for network effects."

4.1 Constructs for Actors and Processes as well as their Form and Function

Following our argument from the previous section, we build our conceptualization of actor and process constructs on system dynamics theory (De Wolf & Holvoet, 2004; Forrester, 1961; Nicolis & Prigogine, 1977) and introduce the constructs *participants*, *participant groups*, *activities*, *influences*, *transactions*, and *causal loops.*

The construct *participant* describes specific entities with respect to the service platform inside and outside its ecosystem. From a system dynamics point of view, participants are sources of small stock, perceived to be static in capacity and accumulation (Forrester, 1961). A group of specific entities of a large but finite size is called a *participant group*. The construct of a participant can be equally applied for consumers and suppliers (such as third-party software vendors like Zynga) or internal actors such as development teams. For the sake of simplicity and since participants can have multiple roles, we have not introduced multiple constructs for consumers, suppliers, etc. In a trade-off, we chose to distinguish groups of participants from individuals to highlight their importance in terms of capacity in causal loops.

> **Design Principle 1:** Provide the method with a technique to conceptualize all specific and unspecific *participants* and *participant groups* relevant to the service platform for users to distinguish all sources that can have transactions with activities on the service platform or that can be influenced by or influence other participants or participant groups.

Activities correspond to stocks in system dynamics (Forrester, 1961). They can accumulate and deplete. Activities are any IT-enabled tasks of the platform operator (e.g., registration on a website, an app store, streaming music). They represent the location for the interaction of participants or participant groups with the service platform. They are the target of participants or participant groups addressed through transactions. Outgoing transactions to participants and other activities describe workflows. Any accumulation within an activity is considered an increase in value. Value denotes positive effects on the performance of actions, objects, and tasks. For example, the quantity of movies available on a video streaming platform may have value for service consumers. Likewise, the more participants subscribed, for example, to a social media network, the more the stock activity accumulates and vice versa.

> **Design Principle 2:** Provide the method with a technique to conceptualize all *activities* on the service platform for users to distinguish all stocks that can have transactions with other activities or participants or can influence participants.

Transactions are value flows (Forrester, 1961; Sterman, 2000). Participants, participant groups, and activities can be the source of a transaction. Value flows coming from the ecosystem participant can only target activities, as those are the only constructs inside the service platform that exhibit stock characteristics. Transactions may be purchases of services or data exchange such as registrations on a service platform.

Influences stimulate ecosystem participants or ecosystem participant groups to choose specific value

flows into the service platform. From a system dynamics perspective, influences are auxiliary variables that control the rate of flow of transactions through the stimulus of their sources (Forrester, 1961; Sterman, 2000). Influences therefore exclusively address participants or participant groups, that is, members of the platform ecosystem. Examples for influences include price change notifications, reviews, competitor offerings, or changes in group behavior (bandwagon effect).

> **Design Principle 3:** Provide the method with a technique to conceptualize *influences* and *transactions* on the service platform and the platform ecosystem for users to categorize interactions between participants, participant groups, and activities, respectively.

If we had considered both flows and auxiliary variables as the mere input and output of participants and activities, then we would have only considered their interactions as one construct rather than separate transactions and influences. This would have resulted in a simpler structure of the model, yet it would have reduced the options to place controls in a differentiated manner.

Further, we contemplated introducing *causal loops* as a distinct construct. Yet causal loops that incite *network effects* consist of a concatenation of the above constructs and therefore are not constructs of their own (De Wolf & Holvoet, 2004; Nicolis & Prigogine, 1977). Introducing them as their own construct would simulate independence from the existing circumstances, which is not accurate. We assume that any perceived gain in guidance would be thwarted by dependencies with other constructs.

> **Design Principle 4:** Provide the method with a technique to design *causal loops* on the service platform and the platform ecosystem for users to make explicit possible *network effects* involving participants, participant groups, and activities.

Figure 2 gives a schematic overview of the discussion thus far. It will be referenced again to explain further constructs. Participants with no relationship to the service platform cannot be the source of transactions. Nevertheless, as the origin of an endogenous variable, these participants must be able to influence ecosystem participants (Figure 2, a). If a participant is an ecosystem participant, the participant may have a defined relationship with the service platform and may thus be the source of a transaction (Figure 2, b), for example, as a customer subscribing (Activity 1) to the service platform. The participant may also be influenced by other ecosystem participants as a sink of an endogenous variable (Figure 2, c).

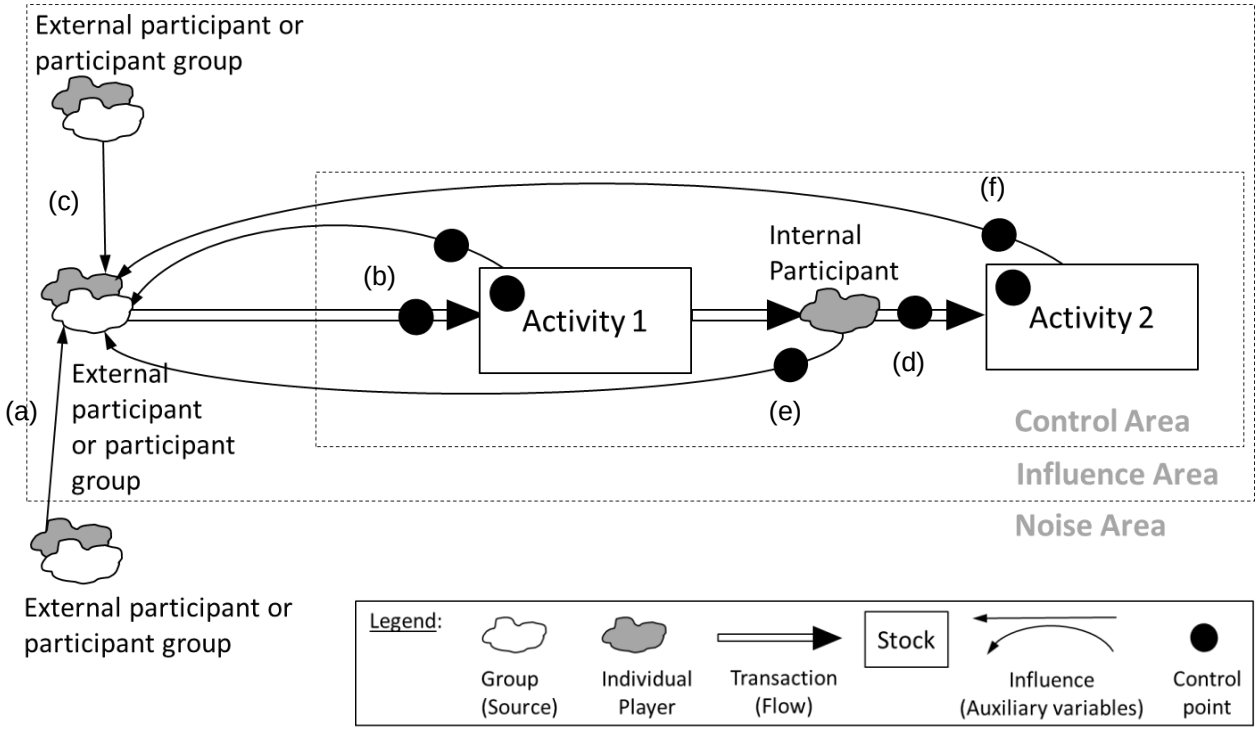


Figure 2. Simplified Stock and Flow Diagram (Norta, Hendrix & Grefen, 2006) with Adapted Terminology in the Context of Structural Allocation

Participants cannot influence activities or internal participants. From a design point of view, the platform operator should attempt to influence the external participant creating a causal loop. Otherwise, his or her influence on the ecosystem will not be controllable. An internal participant can be the origin of an auxiliary variable that stimulates a value flow into the service platform. For example, an internal department could provide an existing (considered static) stock of services for deployment into Activity 2 (Figure 2, d), for example, downloadable content for an entertainment or gaming services. This provisioning may serve as the base value. The influence pointing from the internal participant to the external participant (Figure 2, e) (e.g., blog posts) may stimulate a value flow to fill Activity 1 (i.e., subscriptions) as may the influence from Activity 2 (Figure 2, f) (e.g., downloads).

The design of causal loops, which is crucial to the success of service platforms, is inherently complex (Ondrus, Gannamaneni & Lyytinen, 2015). During our research project, we have gathered best-practice advice on how to design causal loops. However, since the design of a service platform is a situational and bespoke undertaking, a normative decomposition into further design principles is unfeasible. Consequently, the following should be taken as a recommendation only and not as a recipe.

We have observed that after specifying all known participants and activities, the platform designer should attempt to connect participants in simple one-sided direct causal loops. For example, if no direct contact to customers is present, one should consider introducing activities that foster community behavior such as chats or forums. Only then should the platform designer attempt to connect participants in two-sided or multisided indirect causal loops. These can augment direct causal loops or create new causal loops. For example, the designer could influence suppliers through the growth of the user base, and, vice versa, influence participants through new service offerings. This can be especially helpful if it is not possible to engage certain participants in a direct one-sided causal loop.

Since participants' interests in the service platform differ, multiple causal loops may be necessary to incite the desired network effect. Hence, the most important participants should be involved in multiple causal loops. For example, a social network may provide multiple activities for participants to interact with each other (e.g., direct communication through messengers, indirect communication through wall posts, or situated communication through games). Service platforms without a properly placed base value are likely to be nonstarters and, thus, unsuccessful. Closed service platforms that rely only on their base value may eventually run out of stock. For example, Steam was introduced as an updating and anti-piracy facility for Valve's own games. Its growth only started once further publishers made their games available on the service platform.

Activities can also represent this base value to incite service network effects (Oliver et al., 1985). We found that, because of their limited scalability, activities including their own base values should point to a causal loop rather than be part of a causal loop. Similarly, participants represent bottlenecks within a causal loop because of their nearly static stock behavior. Such loops rather require participant groups who can fill the accumulating activities on a service platform. Appendix D and E present and illustrate two archetypical causal loop patterns representing one-sided direct network effects and two-sided indirect network effects as well as three design patterns for service platforms.

4.2 Framing Constructs and their Form and Function: Areas of Staged Platform Authority

Based on control theory and, specifically, the taxonomy of control mode theory, we use the term *platform authority* to refer to the platform operator's ability to exert control (in any of the four modes) over the quality of offered services (Kirsch, 1997). However, different areas of staged authority exist for platform operators. These areas indicate that the provider has full, limited, or no authority over service consumers or service providers because of the ecosystem's inherent ability to self-organize (De Wolf & Holvoet, 2005; Nicolis & Prigogine, 1977). This can also be observed with different kinds of service intermediaries (Heinrich, Leist & Zellner, 2011; Legner, 2009; Scholten et al., 2009). We derive the following three areas based on these observations.

In the *control area*, the platform operator has full platform authority. Activities are exclusive to the control area. From a technical point of view, this means that the platform operator has the capability to enforce its technical infrastructure on all technically enabled activities that take place on the service platform (Heinrich et al., 2011). Examples include service consumption and service provisioning by third-party service providers. From an organizational point of view, this means that the platform operator can exert full platform authority over workforce participants (e.g., internal teams or external entities working on an assignment). Hence, the control area is the area in which the platform operator can fully exert control over activities, internal participants, and internal and incoming transactions. Moreover, the platform operator uses the control area to exert influence over the ecosystem participants. For example, Amazon's control area, with respect to their cloud offerings, encompasses all IT-enabled services surrounding Amazon Web Services and the responsible employees. It includes neither its customers (e.g., Netflix) nor its suppliers of

third-party services (e.g., Sophos) nor its competitors (e.g., Google, Deutsche Telekom).

In large models, it may be necessary to structure elements of the control area, for example, by grouping participants and activities because they are building a set of solutions or because they are in the same physical location. In areas where service network effects apply, the platform operator must be aware that these environments require scaling.

> **Design Principle 5:** Provide the method with a technique to conceptualize a *control area* for users to distinguish the section in which the platform operator can exert full platform authority through enforcement.

The *influence area* is the structural area of the ecosystem surrounding the control area. Ecosystem participants, who are in or may come into a value exchanging relationship with the service platform, are located in this area. The influence area does not allow for control through enforcement, as it is outside the reach of the platform operator's platform authority. It is a self-organizing system that features autonomy, adaptability, and sensitivity to change (De Wolf & Holvoet, 2005; Nicolis & Prigogine, 1977).

The influence area therefore requires indirectly operating methods of control that the platform operator exerts from his control area, so-called incentivizing methods. In the influence area, the ecosystem participants can also influence each other. Finally, they are subject to influences of entities external to the ecosystem (e.g., competitors). As an example, the influence area of the aforementioned Amazon Web Services would include their customers and suppliers as well as transactions with and influences on them, but it would not comprise Amazon's internal activities or their competitors.

> **Design Principle 6:** Provide the method with a technique to conceptualize an *influence area* for users to distinguish the subsection of a platform ecosystem where the platform operator can only exert limited platform authority through incentives.

The *noise area* is the structural area outside the platform's ecosystem. In this area, the platform operator cannot exert any influence on participants. It accommodates competitors and participants uninterested in becoming customers. Participants in the noise area are neither in a relationship of value exchange with the service platform nor can they exert influence on the platform operator. However, they may influence the ecosystem participants of the service platform and may, thus, cause a backflow of value (e.g., the rise of Facebook led to a decrease in unique MySpace visitors). No value flow happens between the noise area and control area, as they have no direct relationship with each other. Neither can any construct in this area be the target of influences from any other area. As mentioned earlier, a common example of participants in the noise area are competitors.

> **Design Principle 7:** Provide the method with a technique to conceptualize a *noise area* for users to distinguish the subsection outside the platform ecosystem where the platform operator has no platform authority and no form of control.

This distinction into three areas is in line with Legner (2009), who provides a rather functional, tripartite taxonomy consisting of (1) infomediaries, (2) e-hubs, and (3) e-markets to describe different levels of control and influence. Complementing this categorization with the concept of (d) integrators (Heinrich et al., 2011) allows us to exemplify these areas of staged platform authority (see Figure 3). The services, which are simply crawled by an infomediary such as IoT Directory that actively collects and provides information about product-service systems in the internet of things, lie outside its control or influence area (Figure 3, a). E-hubs such as the Open Bank Project do not have access to any data traffic while federating a service. They sell a standardized application programming interface and provide a free-of-charge software development kit to financial technology manufacturers and link a large ecosystem of compliant SaaS providers to potential SaaS users. However, both consumer and service provider actively choose cooperation with the e-hub. Therefore, both are located in the influence area (Figure 3, b). E-markets such as Advorto for recruitment services or Stripe for online payments represent a supply concept with limited enforcing authority. E-markets can control all traffic between the client and the service provider, as it is routed through the control area (Figure 3, c). E-markets are common in the intermediation of physical services (e.g., Uber, Airbnb).

Integrators such as the CRM vendors Salesforce and NetSuite are omniscient to all traffic coming from and going to the consumer. They have enforcing power over the service as it is deployed within their control area, i.e., on their servers. However, this omniscience and platform authority shrinks once the service is of a composite nature and uses services outside the service platform's control area (Figure 3, d). The consumer is always placed in the influence area.

The previously discussed, Figure 2 also depicts these three staged areas of platform authority and allows for an improved visualization of interaction options. It now becomes clear which of the participants can be influenced and which participants have or could have transactions with the service platform since those are located in the influence area. Not considering one of the three areas would remove the ability to distinguish controllable participants (internal) from noncontrollable participants (external/ ecosystem) or remove the ability to distinguish noninfluenceable (competitors) from influenceable participants (ecosystem).

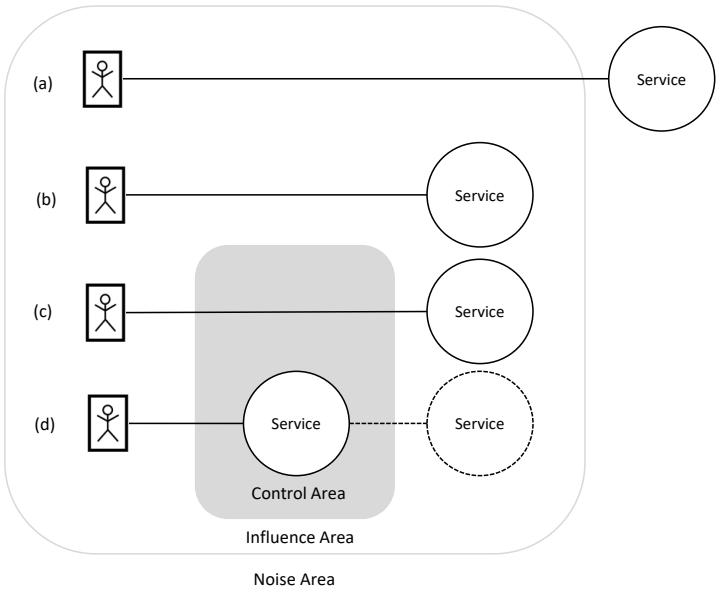


Figure 3. Areas of Staged Platform Authority

4.3 Constructs for Control Methods and Their Form and Function

Control methods with their respective mechanisms allow platform operators to intervene in managing services and service consumption by managed self-organization (Kirsch, 1997). Control methods can be used to steer causal loops surrounding the service platform (i.e., to generate and control network effects). Adding control methods to causal loops turns those loops into controlled feedback systems. However, since full platform authority is restricted to the control area, the positioning of control methods that point to exert service management is exclusive to this area.

In the following, we detail all six control methods introduced in Section 4.1. We explain their applicability to the different constructs before formulating further design principles on how to employ them. The enforcing methods comprise prescriptive control, sanctional control, and restrictive control.

An antecedent condition for *prescriptive control* is the observability of behavior (Kirsch, 1997). Platform operators can observe behavior of third-party activities and participants in the control area since they have submitted to their prescriptions. Having allocated the activities and participants within the control area, the platform operator can both observe and steer the activities of external participants and modify their outcomes (Ouchi, 1979). Hence, prescriptive control refers to the sequence of observing and steering a participant's set of actions within activities. Within causal loops, prescriptive control channels the behavior of internal participants and activities such that they provide a maximum of value for the service network effects by abiding with rules set by the platform operator. For example, a platform operator such as Valve enforces certain software designs to facilitate enhanced observability with respect to service quality features such as cheat detection, transactions, etc., on its Steam platform. Prescriptive control may further include subsequent corrective actions by the platform operator through sanctional control.

The platform operator can use *sanctional control* to either sanction deployed (third-party) services or subscribed participants through suitable activities. Sanctions are enforced in the moment of a policy breach (Henderson & Lee, 1992). The service platform gathers information on policy breaches (e.g., copyright infringement or SLA violations) by automatic verification, service support, or complaint management systems. After the discovery of such an infringement, an escalation routine is initiated. The escalation routine can vary from defined time for correction or statement requested from the participant to immediate un-deployments, depending on aspects of safety, security, or the importance of the policy breach. Within causal loops, sanctional control penalizes activities such that they return to a behavior that provides maximum value for the service network effects. In practice, sanctional control is typically active when operating a service platform. Salesforce, for example, has a two-staged escalation routine for infringed services. They proactively remove the service in question but may redeploy it upon request.

Only a court order or similar outside forces will lead to a final removal of the service.[6]

Restrictive control is a formal control method. It acts as a filter on transactions within the control area and verifies compliance with platform policies (e.g., Transaction 4 in Figure 2). As the platform operator does not observe the generation of a value, restrictive control relies on outcome measurability as an antecedent condition (Henderson & Lee, 1992). Within causal loops, restrictive control confines inflows into the service platform and value flows on the service platform to those of value for the network effect. For example, if a consumer is not considered credit-worthy, he or she will not be able to transact with certain activities on the service platform. The incentivizing control methods comprise market regulative control, informative control, and motivational control.

Market regulative control categorizes influencing control methods that are fully driven by the ecosystem and generated through explicit feedback (Ouchi, 1979). Market regulative control can address service consumers (e.g., through product rankings) as well as service providers (e.g., through recommendations). Its objective is to communicate information on service quality. This is a self-organizing process (De Wolf & Holvoet, 2005; Nicolis & Prigogine, 1977). The ecosystem self-organizes when, in reciprocity, consumers adapt their consumption behavior and service providers amend service quality. This represents a causal loop within the control area as well as into the ecosystem in the influence area. Hence, within causal loops, market regulative control provides a scalable means of independent and individualized quality management that goes beyond the ability of an internal business unit to improve the focus for value flows to incite service network effects. For example, collaborative feedback systems such as Google Play's reviews enable participants to recommend or to advise against a value contribution provided by another participant and, thus, provide structure and highlight the content of their app market. Market regulative control is located exclusively at influences and activities within the control area.

Informative control stimulates creativity in the ecosystem, targeting individuals or communities and providing them with preprocessed information (e.g., regarding service requirements, preferences, or feedback on a specific quality). It addresses the participants' intrinsic motivation (Frey & Oberholzer-Gee, 1997) and consequently highlights opportunities or invitations to participate in activities. In contrast to the contributions in market regulative control, which are community-based, the platform operator manages informative control and addresses existing or potential participants or participant groups. Within causal loops, the goal of informative control is to incite a self-organizing process of alignment and retention in favor of the service platform. It addresses external participants (e.g., customers) through influences—for example, through notifications on content views, as Google Maps does after submitting reviews for locations.[7]

Motivational control comprises methods that explicitly set incentives and potential rewards for participants. It works extrinsically (Frey & Oberholzer-Gee, 1997). The platform operator can incentivize activities in the control area. Moreover, motivational control can affect influences on participants or participant groups in the ecosystem (e.g., to motivate participants financially to produce services that are of strategic relevance to the service platform in a certain segment). Within causal loops, motivational control communicates explicit benefits for participants, which is often helpful in the early stages of establishing a network effect. For example, Dropbox motivates subscribed participants to invite new participants by offering extra storage as a reward.[8] Often programs such as these are discontinued when a sufficient number of users have subscribed. Table 2 gives an overview of the applicable control methods for each actor and process construct within the control area.

Table 2. Activities, Participants, Transactions, Influences, and their Respective Control Methods

	Enforcing methods			Incentivizing methods		
	Prescriptive control	**Sanctional control**	**Restrictive control**	**Market regulative control**	**Informative control**	**Motivational control**
Participant	x					
Activity	x	x		x	x	x
Transaction			x			
Influence				x	x	x

[6] https://www.salesforce.com/company/legal/
[7] https://maps.google.com/localguides
[8] https://help.dropbox.com/accounts-billing/space-storage/earn-space-referring-friends

Using these six control methods, the platform operator can structure its authority over participants and participant groups, activities, transactions, and influences to incite and harness service network effects on its service platform. Each conceptualized method needs to be activated to produce an effect. This results in four further design principles:

Design Principle 8: Provide the method with a technique to place and activate *prescriptive control methods on internal participants* who are in hierarchical subordination to the platform operator's authority for users to monitor and steer their sets of actions.

Design Principle 9: Provide the method with a technique to place and activate *all but restrictive control methods on activities* for users to regulate their interactions by giving prescriptions, monitoring, and possibly intervening or incentivizing and influencing.

Design Principle 10: Provide the method with a technique to place and activate *restrictive control methods on transactions* for users to regulate the inflow (resp. outflow) into activities and thus ensure compliance.

Design Principle 11: Provide the method with a technique to place and activate *incentivizing control methods on influences* for users to regulate the feedback into the platform ecosystem.

In terms of design rationale, we have defined one set of control methods per construct. For all four design principles, we have considered all control methods and only allow those that apply to the context of the respective construct. We have considered formulating six design principles as one per control method. Yet we found it more intuitive and better aligned with an ISDT for design methods to consider control methods in the service platform design based on preexisting actor and process constructs rather than independent thereof. For a more detailed discussion of the design decisions, also see Appendix B.

Figure 2 also illustrates the application of three of the above four design principles through black dots: activated control methods on activities (Activities 1 and 2), on transactions (b and d), and on influences (e and f). Appendix E provides three examples of best-practice patterns for common features of service platforms, including the respective controls.

Once more, we have gathered practical advice for the placement of control methods. Again, the following should be considered as a recommendation only and not as a "one-size-fits-all" prescription. Activities conceptualize a part of the service platform, where third-party applications can be executed on the service platform's (virtual) infrastructure. Consequently, it is important to ensure that actions can be taken for security breaches, performance problems, or legal issues. Hence, the platform designer should consider using both enforcing methods on activities in order to sanction third-party behavior on the service platform that does not conform to prescriptions.

As transactions represent value flows and can only flow into or within the control area, it is crucial to use restrictive methods to monitor and control the transactions entering the service platform, as the origin of the transaction cannot be observed and controlled. Transactions in the control area may not be as crucial. Influences are a group of stimuli on ecosystem participants. Attracting and retaining the right participants to the service platform with motivational control is often the most effective but also the costliest option. Hence, informational control and market regulative control can substitute (or support) activities and influences at later stages of platform development once the relevant data is available (e.g., download numbers or customer reviews).

Finally, we found that it is important not to overregulate third-party behavior on the service platform. All controls should be conceptualized and placed as deemed necessary. Yet, one must carefully consider which controls to activate at the same time. This also suggests that it may be useful to design a roadmap of multiple stages of service platform evolution, implementing and controlling further causal loops as the service platform progresses.

5 Expository Instantiation

5.1 Instantiation as a Conceptual Modeling Language

To facilitate the design process of a service platform, any development group must agree on some shared representational forms. Based on these, they can exchange and discuss ideas, thoughts, opinions, objectives, and beliefs about the object system (Hirschheim, Klein & Lyytinen, 1995) (e.g., about the design and parameters of a service platform). One suitable means of representation is considered to be *conceptual modeling* (Frank, 1999).

In this section, we present a graphical modeling language derived from our ISDT for the design of service network effects to instantiate its design principles. It is a means of documentation, of support for analysis and design, and of facilitating communication. It is an early prototype, a proof of concept that makes our theory actionable by providing a notation for a design method to model network effects for a service platform. It shows that our ISDT can be employed to design a suitable design method. We have collected evidence supporting the artifact's utility, quality, and efficacy through an initial assessment with two focus groups.

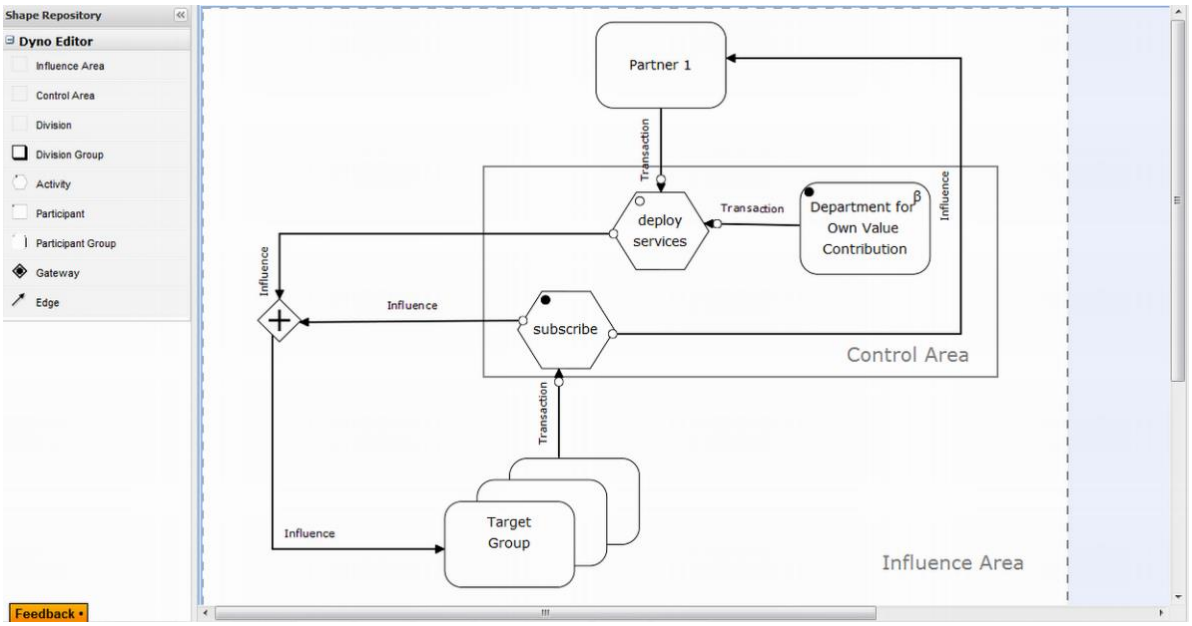


Figure 4. Screenshot of Cloud-Based Editor with a Sample Model

We implemented the notation's syntax and morphology through a stencil set, a plugged-in runtime constraint, and the layout processor within the Oryx framework (Decker, Overdick & Weske, 2008). The editor includes a shape repository, accommodating the language's structural and procedural elements, a modeling canvas, and a property configuration panel, allowing for configuring the control methods. Figure 4 presents a screenshot of the editor with a sample model. We have documented all major elements of the language in Appendix C using the OMG Unified Modeling Language (Object Management Group Inc., 2015) and the OMG Object Constraint Language (Object Management Group Inc., 2014). A full specification of the language can be found in Scholten (2013).

The sample model displays the control area (box with black line), influence area (box with dashed line) as well as the noise area (shaded canvas). There are two activities (hexagon) and one participant (box with rounded edges) in the control area and one participant and one participant group (three stacked boxes with rounded edges) in the influence area. All value flows between the contructs are either labeled as an influence or as a transaction. Whenever multiple transactions or influences converge, we use a merging gateway (diamond with a plus) that symbolizes that all interactions lead to a single value flow. All controls are marked as white or black dots on participants, activities, influences, and transactions. Black dots signify activated controls and white dots signify control points with the possibility to enable control methods. The base value of the service platform is displayed as the symbol *β* with the *Department for Own Value Contribution*. This department now deploys services on the service platform, which serves as a base value. This influences the *Target Group* to *subscribe*. The more subscribers the service platform has and the more services the service platform has, the more the Target Group subscribes, that is, the stock is filled. This creates a causal loop, a direct demand-sided network effect. The amount of subscriptions eventually also has an influence on *Partner 1* to start deploying services on the service platform. This activity eventually influences more participants of the Target Group to subscribe and so forth. This second causal loop is an indirect cross-sided network effect through service consumption and third-party supply.

5.2 Exemplary Application: The Case of M-Engineering

We have modeled a set of existing service platforms that were part of our initial explorative analysis. Service platforms comprised Salesforce, NetSuite, Dropbox, and Google+. This helped us explore the conceptual modeling language's expressiveness (i.e., its capability to represent all relevant processes and control methods encountered in the real world in a semantically and syntactically correct way).

In the following, we exemplarily present the results achieved with M-Engineering, a company offering surveillance, control, and data acquisition solutions (SCADA) in automated processes. The company offered on-premises solutions and is a new entrant into the service platform business. Figure 5 presents the model, developed by one of two key evaluation users, the company's solution manager and platform architect. To start with, Design Principles 5 to 7 were automated using the software and placed the boundaries of the control area and influence area on the noise area as a modeling canvas according to the restrictions introduced above.

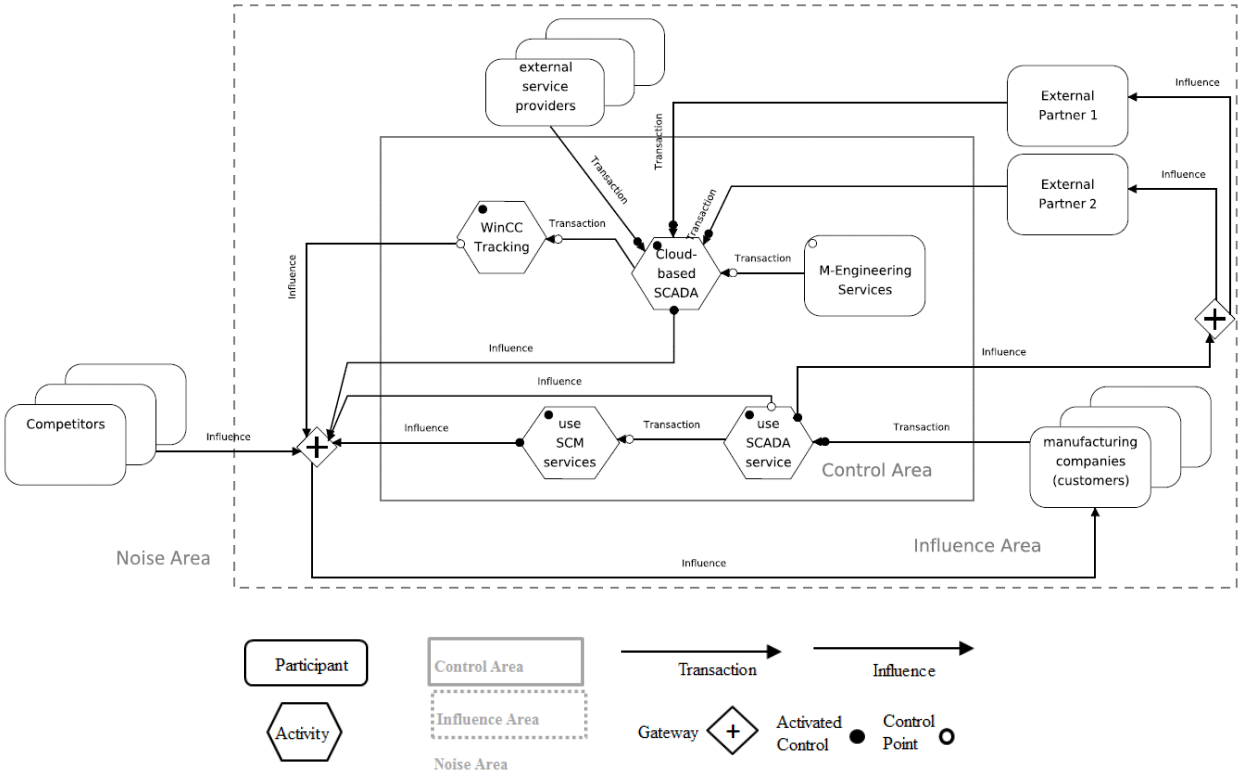


Figure 5. Model of the M-Engineering Service Platform

Following Design Principle 1, M-Engineering decided to distinguish the internal participant *M-Engineering Services* as the operator of their key service. Relevant participants groups are *manufacturing companies (customers)* and *external service providers*. Some of the latter are addressed as an individual participant (*External Partner 1* and *2*) since they will receive individualized treatment by M-Engineering. They expect competitors to influence their service platform as well. *External service providers* were consciously placed in the influence area while *competitors* remain noninfluenceable in the noise area. Following Design Principle 2, the modelers centered their value proposition on the deployment of a *cloud-based SCADA* solution (modeled as an activity). Further activities include their native app *WinCC Tracking*, the *use of SCADA services* as well as the use of their just-in-sequence supply-chain integration (*use SCM services*).

Embracing Design Principles 3 and 4, the modelers have connected all participants and activities with transactions and influences to conceptualize flows of information and services as well as causal loops. The *cloud-based SCADA* solution is operated by *M-Engineering Services* and is used by the *WinCC Tracking* native app (modeled as transactions).

They expect this offer to influence and attract *manufacturing companies* to subscribe and use their services (modeled as a transaction) despite *competitor* influence. They saw the potential to design consumer-sided network effects through the offer of *SCM services*. Transferring detailed process and quality data from one manufacturer in the chain becomes simple through the cloud-based aggregation of SCADA data. M-Engineering is aware that its customer-base it is too small to create strong network effects on the supply side. However, the consumption of services promises to influence specific hand-selected partners (*External Partner 1* and *2*) to provide additional services to support an emerging indirect two-sided network effect.

Following Design Principles 8 to 11, M-Engineering chose not to control their internal participant. However, they chose to activate restrictive control methods on all transactions from the influence area. Furthermore, they modeled the inclusion of several control methods on activities and influences. For example, they chose to use market regulative control, where one provider can invite co-producers from the same supply chain. Additional free data storage for the inviting manufacturing company is part of this motivational control. Also, the offer of a free-of-charge service bottom line to the addressee is motivational control, reducing the addressee's switching costs and representing a suitable way to attain critical mass with respect to network effects (Shapiro & Varian, 1998).

This application confirmed our assumption that the design method can be used to model existing service platforms to the extent necessary to visualize, analyze, and explain existing networks effects. It furthermore confirmed that our conclusion was not primarily based on our possibly biased perception of the design method but was also confirmed by selected platform operators.

5.3 User Feedback

To assess the effectiveness of our first stable configuration of the artifact, we conducted a user survey (Henderson et al. 1995). The survey is based on the two workshops with the aforementioned users from M-Engineering as well as business professionals from an IT company with service platform products. The survey used their respective service platforms as a modeling case. We constructed the survey instrument based on existing questionnaires (Kitchenham & Pfleeger, 2002; Wittern & Zirpins, 2016) to measure how actionable our approach is in supporting the design of service platforms (see also Chandra et al., 2015). First, we modified the wording to fit the situation. Second, we developed some novel items relating to our design requirements. Each question was measured on a 5-point Likert scale, ranging from *-2: strongly disagree* to *2: strongly agree*. The questions were answered by all ten participants of the two workshops. Table 3 includes an excerpt of the questions asked and their results. Appendix F contains the full questionnaire.

As a result of this preliminary assessment, the modelers reaffirmed the usefulness of our expository instantiation. Moreover, the design method was found to be suitable for the design of service platforms (utility) and not only shortened the design time (efficacy) but also improved the understanding of the effects at work (quality). We received less favorable results for the current usability of the implementation. We assume that this is mainly because we limited ourselves to web-based open source modeling frameworks and we have not yet focused on a wizard-like interface to guide the modelers but gave them full access to all constructs of the design method.

The approach could be extended by following Tremblay, Hevner, and Berndt (2010), who suggest using confirmatory focus groups for the refinement of a proposed artifact and the evaluation of its utility. In this case, participants would be asked to use a traditional design method, and subsequently the proposed approach, and asses the usefulness. Their assessment method, however, would not offer evidence to judge individual design principles as it would also propose a summative appraisal of the design method.

Table 3. Selected Scores from the Questionnaire (Excerpt)

Question	Aggregate score (*n*=10)
The design time was shorter when using the software as compared to a design without the design method and software	2
I was quickly able to understand both design method and software	1.2
The design method gives me a better understanding of network effects and control possibilities in platform ecosystems	2
The design method and the software helped me to produce a better solution than without them	2
All relevant elements from the real-world scenario find application in the model representation	2
The graphical user interface (GUI) was intuitive and easy to follow	-0.6
Average of all 45 questions asked	1.60

6 Evaluation of Design Principles

To specify our design principles, we drew from several underlying theories (see Section 3). We derived the design principles conceptually, deducing their necessity argumentatively from literature and experience and explained our rationale (see Section 4). To make our design principles actionable, we instantiated them in the form of a conceptual modeling language (see Section 5). While Gregor and Hevner (2013, p. 351) argue that "a proof-of-concept may be sufficient" when judging design science research contributions, we use this expository instantiation also in the design of an evaluation for our design principles to support textual description of service network effects with diagrams.

However, the evaluation of digital platforms in design science studies presents methodological challenges because typical evaluation criteria for IS design such as user acceptance or system quality do not necessarily suffice for platforms. Furthermore, evaluation approaches for platforms are difficult to develop since platforms in and of themselves offer little value for end users without the services being provided by them (de Reuver et al., 2018).

We propose an ex post analysis that can either be naturalistic (focus group, survey) or artificial (laboratory experiment) to evaluate our design principles (Venable, Pries-Heje & Baskerville, 2012; Venable, Pries-Heje & Baskerville, 2016). Consistent with Venable et al. (2012), we have already argued a logical proof for the necessity of some design principles when discussing our design rationale. For example, we have argued that removing an area of staged platform authority would remove the ability to differentiate options for control or influence. A logical proof, however, does not shed light on the actual ability of our design principles to guide service platform designs for network effects.

When describing his *theory of action*, Norman (1986, p. 55) states that it is "an important point to realize that approximate methods suffice" when trying to guide design construction and use of systems. At the beginning of a design project, a person forms a mental model and expresses his or her goals in these terms. The resulting system design has features expressed relative to its physical state. His theory describes the *gulf of execution* as the discrepancy between psychological and physical variables when creating a system design and the *gulf of evaluation* as the degree to which the artifact provides representations that can be directly perceived and interpreted in terms of the expectations and intentions of the user (Norman, 1986, 1988). These gulfs are theoretical constructs we deem suitable for framing our evaluation.

Consequently, the evaluation of our design principles is a triple-edged sword. On the one hand, we need to test if the design principles bridge the gulf of execution and indeed support the generative task of designing. On the other hand, we need to check whether the designs according to our design principles improve the understanding of service network effects by users and conform to their expectations and, thus, bridge the gulf of evaluation. Further, we must assess the perceived expressiveness of the design principles to understand if they seem to be complete to the user. In the following, we propose a survey to perform an artificial ex post evaluation.

6.1 Survey Design

Norman's gulf of execution bridges several segments relevant to our evaluation: intention formation, specifying the action sequence, and executing the action. While the former two are mental actions relating to design principles and guidelines in terms of language and procedure, the latter is the physical action of creating a service platform design that can be observed and whose results can be analyzed. For analyzing the generative task of design, we borrow from Bowen, O'Farrell, and Rohde (2009) and Gassen, et al. (2016), who argue that complexity limits our ability to execute because of information overload on our limited working memory. Similar to Bowen et al.'s evaluation of query design, we propose evaluating whether following our design principles reduces inherent complexity when designing a service platform for network effects rather than increasing it. In doing so, we evaluate the specific modeling process as well as the result of that modeling process—that is, the conceptual model—rather than evaluating the metalayer by having subjects create a design method using our design principles.

Proposition: Service platform descriptions and diagrams in a notation for a design method based on all of our design principles will better bridge the gulf of execution by reducing complexity than a subset or superset thereof. This will lead to:

H1: Higher assessment scores in the qualitative assessment of the newly created model by model users.

H2: Greater satisfaction with and confidence in the resulting model by the model creator.

H3: Shorter or similar design time of the model.

We consider the cognitive theory of multimedia learning (Mayer, 2005; Mayer, 2009) to be an appropriate foundation for the evaluation of understanding, bridging the gulf of evaluation. In the past, this theory has been used to evaluate data models, object models, and process models (Burton-Jones & Meso, 2006; Gemino & Wand 2005; Mendling, Reijers & Recker, 2010). The theory provides broad variables for the comprehension of audio or text and visual information when multimedia messages are organized into mental models in one's working memory. It assumes that the human brain processes information in these dual channels, has a limited capacity (see also cognitive load theory, Sweller, 1988), and learns actively. We argue that the mechanisms at work can be used to evaluate the understanding of service network effects designed for by following our design principles. To judge the level of understanding, we apply Mayer's (1989) model for understanding, which has been adapted to the IS domain by Recker, Reijers, and van de Wouw (2010): understanding can be measured based on the knowledge construction from learning material (i.e., the content), instructional method (i.e., the representation), and learner or user characteristics. While knowledge construction as a learning process cannot be directly observed (Gemino, 1999), understanding can be measured based on the knowledge a user acquires as a result of the learning process (Recker et al., 2010).

We propose testing this across three measures of understanding (a search-recognition and inference test, a problem-solving test, and a "fill-in-the-blank" test). This leads to the following testable proposition and hypotheses:

Proposition: Service platform descriptions and diagrams in a notation for a design method based on all of our design principles will increase users' understanding of the domain. This improved understanding will lead to:

H4: Higher retention scores in search-recognition and inference questions about the case.

H5: Higher transfer scores in problem-solving questions about the case.

H6: Similar recall scores in a cloze test about the case.

To assess the expressiveness of our design principles, we borrow from Recker et al.'s (2011) evaluation of modeling grammars. They base their work on the idea that the grammar of a modeling language determines the outcomes of the modeling processes (Wand & Weber, 1993). Similarly, we assume that the configuration of design principles determines the outcome of the design process. Hence, we state that our design principles should be free of principle deficit, principle redundancy, principle overload, and principle excess. That is, we do not miss principles to describe real-world phenomena, we do not provide more principles than required for a single phenomenon, we do not provide principles that can be used to describe more than one phenomenon, and we do not provide principles that are not relevant to describe phenomena. Henceforth, any other configuration of design principles should lead to lower scores.

Proposition: A design method based on all of our design principles together will be regarded as more ontologically expressive than a design method based on a subset thereof. This will lead to:

H7: Lower scores in principle deficiency, redundancy, overload, and excess.

6.2 Survey Excerpt

For the questionnaire, we propose describing the case using text as well as the aforementioned conceptual modeling language as an instructional method to devise an A/B test. Each design principle belongs to one of three categories: actors and processes, areas of staged platform authority, or control methods. To evaluate our design principles, we suggest removing one design principle from each of these groups for one of the two test cases: Case B neither contains a control area, nor does it differentiate between influences and transactions, nor does it allow for controls on internal participants. All participants should be provided with the relevant design principles and guidelines for their set. User characteristics have been considered in terms of the user's experience with service platforms and their specification as text and model.

To test for generative aspects of our design theory, we suggest asking participants to employ the modeling language based on our design principles and guidelines to design a service platform. We recommend using an open question about a specific type of a service platform generally known to the participants, such as:

> *Please conceptualize a service platform for the distribution of online games using our design principles. Focus on a design that makes network effects more likely to appear and be controlled so that they support the business of the service platform.*

The aim of this task is to assess whether our design principles enable creating a meaningful design rather than seeking the correct solution to a question. By abstaining from a detailed textual description, we avoid spelling out the details of the service platform design since this would result in a transfer test rather than an assessment of generative aspects. We suggest taking the time to create a model to evaluate if using our design principles results in similar or better design times. Once the design is completed, we propose scoring the model using two independent reviewers and a 7-point Likert scale to evaluate how plausibly the service platform design is able to incite network effects and whether the model contains any defects. Further, participants should be asked if they are satisfied with this model and confident that it meets expectations:

> *The model I created is an exact representation of my design goals.*
>
> *The model I created will meet the expectations of the model user.*

Additionally, we propose an evaluation of the user's understanding by testing for retention (i.e., comprehension), transfer (i.e., the ability to use knowledge), and recall (i.e., the ability to retrieve knowledge).

We employ the cross-sided indirect network effect pattern of Appendix D as an example scenario using Dropbox's DBX service platform as a demonstration case. Participants would receive one diagram and case description, which they would keep for the duration of the retention and transfer test. These would be removed for the recall test.

Retention should be measured by the participants' number of correct answers to search-recognition and inference questions about the case. Each question would ask for a multiple-choice selection. The answer would be scored as correct or incorrect. The following is a sample question for search-recognition: "Is the Dropbox internal development currently actively controlled by the platform operator?" The following is a sample question for inference: "Does the deployment of services lead to more deployment of services?"

Transfer performance should be measured by the participants' number of acceptable answers to

problem-solving questions about the case. Each question would ask for a qualitative explanation. The answer would be scored as correct, borderline acceptable, or incorrect. The following is a sample question for transfer: "Assume another kind of third-party partner: they would not add services to Dropbox but use Dropbox in their products to store files as a service. How could they be influenced to add Dropbox to more of their products? How much authority would the platform operator have over them?"

Recall performance should be measured by the participant's capability to complete the narrative of the case in a cloze test. Participants would be assessed by the number of blanks they filled with a correct word or synonym. The following is an sample extract from the recall test: "The case shows how Dropbox Business customers positively _____ other Dropbox Business customers as well as DBX service platform partners through the application of several _____ methods in the context of causal loops."

Toward the end of the questionnaire, the participants would be asked questions regarding the expressiveness of the design method, using a question about each design principle group. For example, concerning the omission of the control area, we propose asking:

> *Have you ever had the need to distinguish concepts (such as participants) you can control from those you cannot?*
>
> *Does the design theory provide sufficient design principles to distinguish concepts you can control from those you cannot?*

Similarly, we propose asking questions about interactions and controls regarding principle deficiencies. We would then recommend using an analogous structure to test for redundancy and overload:

> *I often have to choose between equal concepts to represent one kind of interaction on a service platform.*
>
> *I often have to provide additional information to clarify the context in which I want to use* [*transactions or influences / interactions*] *on a service platform.*

For those in Treatment group A, we added further questions regarding principle excess. For example: "Prescriptive control on participants does not have a real-world meaning on a service platform." See Appendix G for a description of both scenarios and the full questionnaire.

We assume that participants would need at least 60 minutes to formulate meaningful answers and some prior experience with the subject domain. To improve validity, the survey has been reviewed by independent experts (academic and PhD level) for coherence and by students for understandability. Based on their feedback, we revised the cases and questions substantially for additional clarity. In particular, we chose a reasonably sized model, thus reducing the number of constructs to a manageable amount for a survey.

Given his evaluation, we propose testing for the effect of what we believe are underspecified texts and models to show that our design principles improve the understanding of the domain for users and do not contain unnecessary or confusing elements. We acknowledge that we neither evaluate the superiority of the chosen visual representation over text or other notations nor do we gather information on potentially missing design principles in a structured way. Further, we acknowledge that this test does not shed light on the ability of our design principles to create a meaningful design method and/or modeling language and notation, but rather substantiates the ability and utility of one instantiation of our design theory to create new IT artifacts. In doing so, we consider conceptual models to be a suitable multimedia support for generating and understanding. However, we do not test for the effectiveness of different notations. In addition, structured surveys cannot cater to all types of user context factors. We seek to differentiate users based on novice/ expert and field dependence/ independence behaviors. In summary, the questions we ask focus on the omission of design principles.

As regards a suitable sample for the evaluation, we propose collecting data from two populations: (1) IT architects and experts engaging in service platform design, and (2) students of information systems or computer science. This will also allow us to make comparisons with regard to domain knowledge and generalizability.

7 Discussion

7.1 Contributions

Design science research seeks to develop prescriptive design knowledge through building and evaluating innovative IT artifacts that are intended to solve an identified class of problems (Hevner et al., 2004). Our core contribution in terms of this goal is an ISDT that offers explicit prescriptions and further principles of implementation for engineering a method to design service platforms for network effects using control methods.

There are further artifact types in design research beyond methods (Offermann et al., 2010). While our ISDT provides prescriptions on how to design a method to build service platforms for network effects, some of the constructs and design principles may also apply to other artifact types. In particular, their essence can be used to analyze and improve system designs or

instantiations of service platforms for their use of network effects.

As suggested by Gregor and Hevner (2013), we provide new knowledge in the form of a theory-grounded ISDT using elven design principles. Methodically, we combine both improvement research and exaptation research (Gregor & Hevner, 2013). We propose a new solution to known problems—in our case, a new way of designing service platforms—and we apply known solutions extended to new problems, i.e., providing a theoretical rationale for harnessing network effects on service platforms, applying critical mass theory to service platforms, and using control methods to guide the resulting causal loops.

Regarding improvement research, initial observations indicate that our ISDT for service network effects offers an improvement over current ad hoc development approaches because we explicate constructs (e.g., areas of staged platform authority) that are necessary to consider and address the problem of placing and balancing mechanisms for inciting network effects through control methods, which have not been explicitly and consistently considered in service platform design previously. The justification for our approach is grounded in the knowledge base—specifically, in system dynamics, network theory, and control theory as our kernel theories. We thereby deviate from and improve existing service platform development practice. We used our approach to map and analyze multiple real-world scenarios (see also Scholten et al., 2009) and provide evidence that this configuration of design principles is indeed useful.

Similar to Chen (1976), providing the entity relationship model as a design method for Codd's relational model (Codd, 1970), we propose instantiating the design method, our ISDT artifact, in the form of a conceptual modeling language capable of supporting any written specification with diagrammatic models. We agree with Gregor and Jones (2007) that building an expository instantiation and merely demonstrating that it works is not enough. It remains essential to include justificatory knowledge that provides an explanation of why it was constructed as it is and why it works, as well as testable propositions. Therefore, our results include not only an innovative instantiation but also knowledge about creating other instantiations that belong to the same class (e.g., other approaches building on the prescriptions of our ISDT to create methods to build service platforms) (Sein et al., 2011) and testable propositions for an empirical ex post evaluation.

Our instantiation builds on the prescriptions of our ISDT. We show how the design of the instantiation and the use of specific components rests on the principles of these theories. This entails that practitioners have more, and deeper, knowledge to rely on when interpreting service platform designs or using our prescriptions as well as our guidelines in particular circumstances (Gregor & Jones, 2007). In sum, our design science research project makes a contribution to the knowledge base in the form of both the core artifact of an ISDT with corresponding design principles and guidelines (Level 3 contribution) as well as a situated instantiation of a conceptual modeling language (Level 2 contribution) (Gregor & Hevner, 2013).

7.2 Artifact Mutability

Artifact mutability considers "changes in state of the artifact anticipated in the theory" (Gregor & Jones, 2007, p. 322). It is concerned with the mutability of the output, the instantiation of the design theory. Similarly, while the Oryx-based implementation presented above is an instantiation based on our ISDT, the artifact whose mutability needs to be addressed is the conceptual model.

This understanding of mutability is consistent with Gregor's (2006) understanding of Codd's (1970, 1982) presentation of mutability. He proposes views to counter the changes in the base tables (i.e., our models). He does not consider mutability of the relational model/ database management system itself. Hence, the ISDT, as well as its software-based instantiation, needs to be robust and continue to exhibit the traits ascribed to it in changing circumstances. Yet the artifacts produced, i.e., the models or service platforms, need to be flexible and adapt to change.

Conceptual models are naturally mutable. They can be edited and updated at any stage by any actor with access to the software. Saving conceptual models does not render them ineditable with the editor (such as an export into an image format). Furthermore, versioning of the models allows for different variants of the models for different purposes to coexist (e.g., "as-is" vs. "to-be" models). Our decision to model causal loops as a concatenation of constructs further supports this, as the causal loops can be adapted over time rather than being replaced as a whole. Thus, if a new form of network effect is discovered, it is rather likely that our method can describe it in the design process of a service platform using our constructs. In addition, while the categorization of controls methods is extensible, we are confident that it is comprehensive enough to represent further methods of enforcement or incentive.

The mutability of actual service platforms relies on software engineering principles and architecture paradigms. Workflow technology can enable flexibility in the adaptation of transactions and influences; object orientation and web service technology can enable flexibility concerning the adaptation of activities.

7.3 Limitations

As mentioned earlier, our ISDT applies to any service platform design. It is a broad theory. Its merit is that it is applicable to multiple types of service platforms, ranging from social networks to mobile application distribution platforms. As a consequence of this breadth, our theory may lack precision in some domains. There may be factors that influence networks effects in one case but not in another. We do not cover these domain-specific factors. This is a limitation and a trade-off. In addition, our ISDT is not meant to be used to assess whether a service platform design is better or worse than another service platform design, or whether the inclusion of a particular causal loop is good or bad, or whether a certain base value will reach a critical mass. Our ISDT for service network effects may benefit from the inclusion of further domain-specific variables when applied to a particular context. For example, factors from the 6C (Jaffe, 2010) could be used when trying to establish a community first rather than focusing on the speedy return of investment.

We have presented testable propositions for our research based on the omission of design principles for actors and processes, areas of staged platform authority, and control methods. We hypothesize that their omission will reduce the generative aspects, understanding of the resulting service platform descriptions, and the ISDT's expressiveness. We are aware that this evaluation is not all-embracing. It is however not feasible to test any combination of design principles with a suitable group of participants in sufficient numbers. Through the application of the theory in academia and practice, time will tell which configuration of design principles proves to be most beneficial.

Even then, it is practically impossible to relate the design principles to the economic and financial success of a service platform a priori. As mentioned earlier, whether a service platform is successful in the market and reaches a critical mass also depends on additional, at times political factors that cannot be expressed through our ISDT. Moreover, our current transactions and influences represent a generic flow of value and do not attribute concrete valuations to allow for simulation or prediction.

Furthermore, service platform design is rarely a greenfield activity. Hence, it is always necessary to answer to several issues that make the application of feedback theory as applied here in a sociotechnical context different from its rather technical application in system dynamics. For example, characteristics of the (eco)system might impose inherent limitations (e.g., jurisdiction on data privacy), inherent constraints and trade-off may exist (e.g., straightforward service implementation for the service provider vs. comprehensive control over the service's capabilities by the platform operator), and there may be performance limits (e.g., bandwidth); these issues need to be quantified for meaningful design decisions (Chen, 2014). However, these are strategic business questions that our ISDT does not answer.

We are aware of some limitations of the expository instantiation. First, the graphical modeling language is targeted at system architects and decision makers. Therefore, it is rather abstract and it is not possible to (semi)automatically derive any deployment architecture or even program code. So far, it is designed to be a means of communication to support the early stages of system design. Second, despite the language's rather high level of abstraction, our evaluations so far have shown that not all language constructs are self-explanatory. In addition, the current implementation does not offer any form of automated guidance for the creation of new models. In some respect, the same limitations also apply to the understanding of models by users. In contrast to typical process models, there is no single point of entry, and models (i.e., the represented service platforms and their ecosystems) must be understood as a whole.

Third, we have not yet performed a comprehensive side-by-side comparison with other service modeling languages. For example, Eisenmann et al. (2008, 2011, 2006) do not consider technical service management. The open semantic service relationship (OSSR) approach (Cardoso, 2013) and the open semantic service networks (OSSN) approach (Cardoso, Pedrinaci & Leenheer, 2013; Cardoso et al., 2012) focus on graphical models of service networks in the context of service management. The service network notation (SNN) (Bitsaki et al., 2008; Bitsaki et al., 2009), the service network modeling notation (SNMN) (Danylevych, Karastoyanova & Leymann, 2010), and e3* (e3value/e3services/e3controls) (Kartseva et al., 2010) model service networks as a set of nodes and edges in a to-be approach. These notations consider explicit relationships of value exchange. Yet, none of them consider control methods nor the more implicit approaches of ecosystem influence as immanent in network effects. Because of shortcomings associated with our design principles, we have elected to create our own expository instantiation rather than modifying existing languages and dealing with the inevitable repercussions. Ultimately, their extendibility should also be explored.

8 Conclusion

The emergence of service platforms and their openness to contributions of third parties as well as to self-organizing behavior on the consumer side, both give rise to new challenges for existing and future platform operators. The increased autonomy of suppliers and consumers and the network effects resulting from their

behavior increase the complexity of managing such service platforms. Platform operators are aware of the opportunities resulting from service network effects. However, challenges also arise because of the loss of influence on service quality, on the one hand, and the possibility of unachieved growth or rapid collapse of the consumer base because of negative network effects on the other.

We have used, adapted, and structured existing knowledge originating from systems dynamics, network effect theory, and control theory and applied it to service platform design to find a balance between control and self-organization. We have used this to present an ISDT that provides prescriptive knowledge for design and action. It explains how the impact of platform authority is different depending on the level of control available to the platform operator. We have formulated eleven design principles to create a method that guides and improves the design of service platforms for network effects. They guide the development for causal loops and placement of control methods on participants, activities, influences, and transactions, and thus explain how service network effects work. Based on these insights, we have created a conceptual modeling language as an expository instantiation.

For practitioners and researchers, this design method allows for the modeling and the identification of service network effects. It allows platform operators to conceptualize and manage the flows of service provisioning and consumption by fine-tuning their methods of platform authority. It enables researchers to hypothesize more comprehensively about the effects of causal loops and the effect of control methods. Consequently, the proposed method can act as a tool for supporting shared modeling, discussions, analyses, and decisions.

Future research could focus on adding further modes than just control methods. Attributes with other foci and aspects might be useful (e.g., for monitoring or for security). At this stage, our instantiation provides a high-level view of interactions of service platforms and their ecosystems with the focus on harnessing service network effects and placing control methods. To be used in later stages of the systems development process, elements of the language also must be passed on to subordinate layers of modeling such as process modeling languages.

In terms of generalizing the results for the broader topic of (composite) service engineering, a next step would be a comparison and evaluation in a controlled environment against other approaches such as OSSR, SNN, or SNMN, all of which do not consider either service management or ecosystem influence. As additional research builds on our foundation, formal comparison with alternative approaches in a variety of contexts becomes crucial to enable claims of generalizability (Hevner et al., 2004).

References

Abdelkafi, N., Raasch, C., Roth, A. & Srinivasan, R. (2018). Research advances in multi-sided platforms [CfP special issue]. *Electronic Markets*. http://www.electronicmarkets.org/call-for-papers/single-view-for-cfp/datum/2018/03/31/cfp-special-issue-on-research-advances-in-multi-sided-platforms/.

Ashby, W. R. (1964). *An introduction to cybernetics University*. Paperbacks.

Ashford, S. J. & Tsui, A. S. (1991). Self-regulation for managerial effectiveness: The role of active feedback seeking. *Academy of Management Journal*, *34*(2), 251-280.

Barros, A. & Dumas, M. (2006). The rise of web service ecosystems. *IT Professional*, *8*(5), 31-37.

Barros, A. & Kylau, U. (2011). Service delivery framework: an architectural strategy for next-generation service delivery in business network *Proceedings of the 2011 Annual SRII Global Conference*.

Baskerville, R. & Pries-Heje, J. (2010). Explanatory design theory. *Business & Information System Engineering*, *2*(5), 271-282.

Baskerville, R., Pries-Heje, J. & Venable, J. (2009). Soft design science methodology. *Proceedings of the 4th International Conference on Design Science Research in Information Systems and Technology*.

Bitsaki, M., Danylevych, O., van den Heuvel, W., Koutras, G., Leymann, F., Mancioppi, M., Nikolaou, C. & Papazoglou, M. (2008). An architecture for managing the lifecycle of business goals for partners in a service network. *Proceedings of the 1st European Conference ServiceWave.*

Bitsaki, M., Danylevych, O., van den Heuvel, W. J. A. M., Koutras, G. D., Leymann, F., Mancioppi, M., Nikolaou, C. N. & Papazoglou, M. P. (2009). Model Transformations to Leverage Service Networks. *Proceedings of the 4th International Workshop on Engineering Service-Oriented Applications.*

Boudreau, K. (2010). Open platform strategies and innovation: granting access vs. devolving control. *Management Science*, *56*(10), 1849-1872.

Bowen, P. L., O'Farrell, R. A. & Rohde, F. (2009). An empirical investigation of end-user query development: The effects of improved model expressiveness vs. complexity. *Information Systems Research*, *20*(4), 565-584.

Burton-Jones, A. & Meso, P. N. (2006). Conceptualizing systems for understanding: An empirical test of decomposition principles in object-oriented analysis. *Information Systems Research*, *17*(1), 38-60.

Cardoso, J. (2013). Modeling service relationships for service networks. *Proceedings of the 3rd International Conference on Exploring Services Sciences.*

Cardoso, J., Pedrinaci, C. & Leenheer, P. (2013). Open semantic service networks: modeling and analysis. *Proceedings of the 3rd International Conference on Exploring Services Sciences.*

Cardoso, J., Pedrinaci, C., Leidig, T., Rupino, P. & De Leenheer, P. (2012). Open semantic service networks. *Proceedings of the 4th International Symposium on Services Science.*

Casey, T. R. & Töyli, J. (2012). Dynamics of Two-sided platform success and failure: An analysis of public wireless local area access. *Technovation*, *32*(12), 703-716.

Chandra, L., Seidel, S. & Gregor, S. (2015). Prescriptive knowledge in is research: conceptualizing design principles in terms of materiality, action, and boundary conditions. *Proceedings of the 48th Hawai'i International Conference on System Sciences.*

Chen, J. (2014). Fundamental Limitation of Feedback Control. In J. Baillieul & T. Samad (Eds.), *Encyclopedia of systems and control* (pp. 1-10). Springer.

Chen, P. P.-S. (1976). The entity relationship model: toward a unified view of data. *ACM Transactions on Database Systems*, *1*(1), 9-36.

Chesbrough, H. (2012). Open innovation: Where we've been and where we're going. *Research-Technology Management*, *55*(4), 20-27.

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13*(6), 377-387.

Codd, E. F. (1982). Relational database: A practical foundation for productivity. *Communications of the ACM*, *25*(2), 109-117.

Conant, R. C. & Ashby, W. R. (1970). Every good regulator of a system must be a model of that system. *International Journal of Systems Science*, *1*(2), 89-97.

Danylevych, O., Karastoyanova, D. & Leymann, F. (2010). Service networks modelling: An SOA

& BPM standpoint. *Journal of Universal Computer Science*, *16*(13), 1668-1693.

Davison, R. M., Martinsons, M. G. & Kock, N. (2004). Principles of canonical action research. *Information Systems Journal*, *14*(1), 65-86.

de Reuver, M., Sørensen, C. & Basole, R. C. (2018). The digital platform: A research agenda. *Journal of Information Technology*, *33*(2), 124-135.

De Wolf, T. & Holvoet, T. (2004). Emergence versus self-organisation: Different concepts but promising when combined. *Proceedings of the 3rd International Workshop on Engineering Self-Organising Systems*.

De Wolf, T. & Holvoet, T. (2005). Towards a methodology for engineering self-organising emergent systems. In H. Czap, R. Unland, C. Branki & H. Tianfield (Eds.), *Self-Organization and Autonomic Informatics (I)* (pp. 18-34). IOS Press.

Decker, G., Overdick, H. & Weske, M. (2008). Oryx: sharing conceptual models on the web. *Proceedings of the 27th International Conference on Conceptual Modeling.*

Dutta, A., Roy, R. & Seetharaman, P. (2008). Wikipedia usage patterns: The dynamics of growth. *Proceedings of the 29th International Conference on Information Systems.*

Economides, N. & Salop, S. C. (1992). Competition and integration among complements, and network market structure. *Journal of Industrial Economics*, *40*(1), 105-123.

Eisenhardt, K. M. (1985). Control: Organizational and economic approaches. *Management Science*, *31*(2), 134-149.

Eisenmann, T., Parker, G. & Van Alstyne, M. (2008). *Opening platforms: how, when and why?* (Harvard Business School Working Paper). Harvard Business School, Boston, MA.

Eisenmann, T., Parker, G. & Van Alstyne, M. (2011). Platform envelopment. *Strategic Management Journal*, *32*(12), 1270-1285.

Eisenmann, T., Parker, G. & Van Alstyne, M. W. (2006). Strategies for two-sided markets. *Harvard Business Review*, *84*(10), 92-101.

Forrester, J. W. (1961). *Industrial dynamics*. MIT Press.

Frank, U. (1999). Conceptual modelling as the core of the information systems discipline: Perspectives and epistemological challenges. In *Proceedings of the 5th Americas Conference on Information Systems.*

Frey, B. S. & Oberholzer-Gee, F. (1997). The cost of price incentives: An empirical analysis of motivation crowding-out. *The American Economic Review*, *87*(4), 746-755.

Gassen, J. B., Mendling, J., Bouzeghoub, A., Thom, L. H. & de Oliveira, J. P. M. (2016). An experiment on an ontology-based support approach for process modeling. *Information and Software Technology*, *83*, 94-115.

Gemino, A. & Wand, Y. (2005). Complexity and clarity in conceptual modeling: Comparison of mandatory and optional properties. *Data Knowledge Engineering*, *55*(3), 301-326.

Gemino, A. C. (1999). *Empirical comparison of system analysis modeling techniques* (Doctoral dissertation). University of British Columbia, Vancouver, Canada.

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

Gleich, B., Mosig, B. & Reinwald, D. (2011). Contributing to knowledge-based decision support: A system dynamics model regarding the use of non-renewable resources. *Proceedings of the 19th European Conference on Information Systems*.

Gregor, S. (2006). The nature of theory in information systems. *MIS Quarterly*, *30*(3), 611-642.

Gregor, S. & Hevner, A. R. (2013). Positioning and presenting design science research for maximum impact. *MIS Quarterly*, *37*(2), 337-355.

Gregor, S. & Jones, D. (2007). The anatomy of a design theory. *Journal of the Association for Information Systems*, *8*(5), 312-355.

Gregory, R. W. & Muntermann, J. (2014). Heuristic theorizing: Proactiely generating design theories. *Information Systems Research*, *25*(3), 639-653.

Hagiu, A. & Lee, R. S. (2011). Exclusivity and control. *Journal of Economics & Management Strategy*, *20*(3), 679-708.

Heinrich, B., Leist, S. & Zellner, G. (2011). Service integrators in business networks: the importance of relationship values. *Electronic Markets*, *21*(4), 215-235.

Henderson, J. C. & Lee, S. (1992). Managing I/S design teams: A control theories perspective. *Management Science*, *38*(6), 757-777.

Henderson, R., Podd, J., Smith, M. & Varela-Alvarez, H. (1995). An examination of four user-based software evaluation methods. *Interacting with Computers*, *7*(4), 412-432.

Hevner, A. R., March, S. T., Park, J. & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), 75-105.

Hirschheim, R., Klein, H. K. & Lyytinen, K. (1995). *Information systems development and data modeling: Conceptual and philosophical foundations*. Cambridge University Press.

Holt, A., Weiss, K., Huberty, K., Gelblum, E., Flanner, S., Devgan, S., Malik, A., Rozof, N., Wood, A., Standaert, P., Meunier, F., Lu, J., Chen, G., Lu, B., Han, K., Khare, V. & Miyachi, M. (2011). Cloud computing takes off: market set to boom as migration accelerates (Morgan Stanley Blue Paper). Available at https://www.dabcc.com/documentlibrary/file/cloud_computing.pdf

Isaacson, W. (2011). *Steve Jobs*. Simon & Schuster.

Jaffe, J. (2010). *Flip the funnel: How to use existing customers to gain new ones*. Wiley.

Kartseva, V., Hulstijn, J., Gordijn, J. & Tan, Y.-H. (2010). Control patterns in a health-care network. *European Journal of Information Systems*, *19*(3), 320-343.

Katz, M. L. & Shapiro, C. (1986). Technology adoption in the presence of network externalities. *The Journal of Political Economy*, *94*(4), 822-841.

Kirsch, L. S. (1997). Portfolios of control modes and IS project management. *Information Systems Research*, *8*(3), 215-239.

Kitchenham, B. A. & Pfleeger, S. L. (2002). Principles of survey research: Part 3: Constructing a survey instrument. *ACM SIGSOFT Software Engineering Notes*, *27*(2), 20-24.

Lee, J. (1997). Design rationale systems: understanding the issues. *IEEE Expert*, *12*(3), 78-85.

Lee, S. M., Kim, T., Noh, Y. & Lee, B. (2010). Success factors of platform leadership in Web 2.0 service business. *Service Business*, *4*(2), 89-103.

Legner, C. (2009). Do web services foster specialization? An analysis of commercial web service directories. *Proceedings of the 10th International Conference on Wirtschaftsinformatik.*

Markus, M. L. (1987). Toward a "critical mass" theory of interactive media. *Communication Research*, *14*(5), 491-511.

May, N., Scholten, U. & Fischer, R. (2011). Towards an automated gap analysis for e-service portfolios. *Proceedings of the 8th International Conference on Services Computing*.

Mayer, R. E. (2005). Cognitive Theory of Multimedia Learning. In R. E. Mayer (Ed.) *The Cambridge handbook of multimedia learning* (pp. 31-48). Cambridge University Press.

Mayer, R. E. (2009). *Multimedia learning* (2nd ed). Cambridge University Press.

Mayer, R. J. (1989). Models for understanding. *Review of Educational Research*, *59*(1), 43-64.

Meinel, C. & Leifer, L. (2010). *Design thinking: Understand—Improve—Apply*. Springer.

Mell, P. & Grance, T. (2011). *The NIST definition of cloud computing*. https://doi.org/10.6028/NIST.SP.800-145.

Mendling, J., Reijers, H. A. & Recker, J. (2010). Activity labeling in process modeling: Empirical insights and recommendations. *Information Systems*, *35*(4), 467-482.

Nicolis, G. & Prigogine, I. (1977). *Self-organization in nonequilibrium systems: From dissipative structures to order through fluctuations*. Wiley.

Norman, D. A. (1986). Cognitive engineering. In D. A. Norman & S. W. Draper (Eds.), *User Centered System Design: New Perspectives on Human-computer Interaction* (pp. 31-61). Lawrence Erlbaum.

Norman, D. A. (1988). *The design of everyday things*. MIT Press.

Norta, A., Hendrix, M. & Grefen, P. (2006). A Pattern-Knowledge base supported establishment of inter-organizational business processes. *Proceedings of the International OTM Workshop on Modeling Inter-Organizational Systems focusing on Collaboration and Interoperability, Architectures and Ontologies.*

Object Management Group Inc. (2014). *Object Constraint Language. Version 2.4*. http://www.omg.org/spec/OCL/2.4/PDF.

Object Management Group Inc. (2015). *Unified Modeling Language. Version 2.5*. http://www.omg.org/spec/UML/2.5/PDF.

Offermann, P., Blom, S., Schönherr, M. & Bub, U. (2010). Artifact types in information systems design science: A literature review. *Proceedings of the 5th International Conference on Design Science Research in Information Systems and Technology*.

Oliver, P., Marwell, G. & Teixeira, R. (1985). A Theory of Critical Mass: I. Interdependence, group heterogenity, and the production of collective action. *American Journal of Sociology*, *91*(3), 522-556.

Ondrus, J., Gannamaneni, A. & Lyytinen, K. (2015). The impact of openness on the market potential of multi-sided platforms: A case study of mobile payment platforms. *Journal of Information Technology*, *30*(3), 260-275.

Ouchi, W. G. (1979). A conceptual framework for the design of organizational control mechanisms. *Management Science*, *25*(9), 833-848.

Pahl, G., Beitz, W., Feldhusen, J. & Grote, K.-H. (2007). *Engineering design: A systematic approach*. Springer.

Parker, G. & Van Alstyne, M. W. (2005). Two-sided network effects: A theory of information product design. *Management Science*, *51*(10), 1494-1504.

Parker, G. & Van Alstyne, M. W. (2018). Innovation, openness, and platform control. *Management Science*, *64*(7), 3015-3032.

Parker, G., van Alstyne, M. W. & Choudary, S. P. (2016). *Platform revolution: How networked markets are transforming the economy and how to make them work for you*. Norton & Company.

Recker, J., Reijers, H. A. & van de Wouw, S. G. (2010). An integrative framework of the factors affecting process model understanding: A learning perspective. *Proceedings of the 16th Americas Conference on Information Systems*.

Recker, J., Rosemann, M., Green, P. & Indulska, M. (2011). Do ontological deficiencies in modeling grammars matter? *MIS Quarterly*, *35*(1), 57-79.

Rimal, B. P., Choi, E. & Lumb, I. (2009). A taxonomy and survey of cloud computing systems. *Proceedings of the 5th International Joint Conference on INC, IMS, and IDC.*

Rochet, J. C. & Tirole, J. (2003). Platform competition in two-sided markets. *Journal of the European Economic Association*, *1*(4), 990-1029.

Rohlfs, J. (1974). A theory of interdependent demand for a communications service. *The Bell Journal of Economics and Management Science*, *5*(1), 16-37.

Rysman, M. (2009). The economics of two-sided markets. *The Journal of Economic Perspectives*, *23*(3), 125-143.

Schelling, T. C. (1971). Dynamic models of segregation. *Journal of Mathematical Sociology*, *1*(2), 143-186.

Schilling, M. A. (2009). Protecting or diffusing a technology platform: Tradeoffs in appropriability, network externalities, and architectural control. In A. Gawer (Ed.) *Platforms, markets and innovation* (pp. 192-218). Edward Elgar.

Schneider, A., Gschwendtner, A. & Matthes, F. (2015). Using system dynamics models to understand and improve application landscape design. *Proceedings of the 12th International Conference on Wirtschaftsinformatik*.

Scholten, U. (2013). *Dynamic network notation: A Graphical modeling language to support the visualization and management of network effects in service platforms* (Phd dissertation). Karlsruhe Institute of Technology, Karlsruhe, Germany.

Scholten, U., Fischer, R., Bojkov, D. & May, N. (2011). Supply chain control building on emergent self-organizing effects. In R. Bogaschewsky, M. Eßig, R. Lasch & W. Stölzle (Eds.), *Supply Management Research* (pp. 311-335). Gabler.

Scholten, U., Fischer, R. & Zirpins, C. (2009). Perspectives for web service intermediaries: How influence on quality makes the difference. *Proceedings of the 10th International Conference on Electronic Commerce and Web Technologies.*

Scholten, U., Janiesch, C. & Rosenkranz, C. (2013). Inciting networks effects through platform authority: A design theory for service platforms. *Proceedings of the 24th International Conference on Information Systems*.

Scholten, U., Schuster, N. & Tai, S. (2012). A pattern language and repository for service network management. *Proceedings of the 5th IEEE International Conference on Service-Oriented Computing and Applications*.

Sein, M., Henfridsson, O., Purao, S., Rossi, M. & Lindgren, R. (2011). Action design research. *MIS Quarterly*, *35*(1), 37-56.

Shapiro, C. & Varian, H. R. (1998). Versioning: The smart way to sell information. *Harvard Business Review*, *76*(6), 106-114.

Simon, H. A. (1996). *The sciences of the artificial* (3rd ed). MIT Press.

Sterman, J. D. (2000). *Business dynamics: Systems thinking and modeling for a complex world*. McGraw Hill Higher Education.

Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, *12*(2), 257-285.

Thies, F., Wessel, M. & Benlian, A. (2018). Network effects on crowdfunding platforms: exploring the implications of relaxing input control. *Information Systems Journal*, *28*(6), 1239-1262.

Tremblay, M. C., Hevner, A. R. & Berndt, D. J. (2010). Focus groups for artifact refinement and evaluation in design research. *Communications of the Association for Information Systems*, *26*(27), 599-618.

Vaishnavi, V. K. & Kuechler, W. (2008). *Design science research methods and patterns: Innovating information and communication technology*. Auerbach.

Van Alstyne, M. W., Parker, G. G. & Choudary, S. P. (2017). 6 reasons platforms fail. *Harvard Business Review*. https://hbr.org/2016/03/6-reasons-platforms-fail.

Vargo, S. L. & Lusch, R. F. (2004). Evolving to a new dominant logic for marketing. *Journal of Marketing*, *68*(1), 1-17.

Venable, J., Pries-Heje, J. & Baskerville, R. (2012). A comprehensive framework for evaluation in design science research. *Proceedings of the 7th International Conference on Design Science Research in Information Systems*.

Venable, J., Pries-Heje, J. & Baskerville, R. (2016). FEDS: A framework for evaluation in design science research. *European Journal of Information Systems*, *25*(1), 77-89.

Walls, J., Widmeyer, G. & El Sawy, O. (1992). Building an information system design theory for vigilant EIS. *Information Systems Research*, *3*(1), 36-59.

Wand, Y. & Weber, R. (1993). On the ontological expressiveness of information systems analysis and design grammars. *Journal of Information Systems*, *3*(4), 217-237.

Wanner, J., Bauer, C. & Janiesch, C. (2019). Two-sided digital markets: Disruptive chance meets chicken or egg causality dilemma. *Proceedings of the 21st IEEE Conference on Business Informatics*.

West, J., Salter, A., Vanhaverbeke, W. & Chesbrough, H. (2014). Open innovation: The next decade. *Research Policy*, *43*(5), 805-811.

Wittern, E. & Zirpins, C. (2016). Service feature modeling: Modeling and participatory ranking of service design alternatives. *Software & Systems Modeling*, *15*(2), 553-578.

Yoo, Y., Henfridsson, O. & Lyytinen, K. (2010). Research commentary—The new organizing logic of digital innovation: An agenda for information systems research. *Information Systems Research*, *21*(4), 724-735.

Zelkowitz, M. V. & Wallace, D. (1997). Experimental validation in software engineering. *Information and Software Technology*, *39*(11), 735-743.

Appendix A. Prior Research

Prior steps in the research process consisted of the following activities:

- *Analysis of successful service platforms over a period of four years*. We judged service platforms as "successful" according to two alternative criteria: First, if they were successful in terms of financial success in the service platform domain, substantiated through prior published investigations. Second, if they provided methods or structures that successfully support network effects (e.g., the service platform Trello applies specific structures and methods to achieve network effects in the case of small numbers of users). In particular, we considered Appirio, BOINC, Dropbox, Google, Facebook, Intensify, Intuit, LongJump, NetSuite, Salesforce, SAP, S.Chand Edutech, and Trello. With our choice of companies, we tried to include leading consumer service platforms, strong professional service platforms, and social clouds, as well as service platforms using public cloud applications. We also conducted several field studies with the platform operators CAS Software, SAP, S.Chand Edutech, and M-Engineering and evaluated the outcomes using self-control surveys of participants. The results progressively indicated that our conceptualization of network effect for service platforms matured. Several conceptualizations were discussed and later integrated or abandoned. Examples abandoned include indicators for positive or negative network effects and a quantifiable rather than binary base value.

  Design Science Iteration No. 1: First conceptualization of network effects on service platforms as a means to explain and analyze the effects.

- *Execution of a comparative longitudinal study*. We analyzed service intermediaries, assessing and evaluating their methods, service quality, and service success (Scholten, Fischer & Zirpins, 2009). Specifically, the study compared the intermediary operators SeekDa, WebServiceList, Xmethods, RemoteMethods, eSigma, and StrikeIron Marketplace. This was extended by a longitudinal comparison of service quality for SeekDa and StrikeIron Marketplace.

  Design Science Iteration No. 2: Distinction of different configurations of service platforms using areas of authority.

- *Execution of field studies on three selected service platforms*. We analyzed in detail the service platforms Force.com by Salesforce, SuiteApp by NetSuite, and Facebook Platform by Facebook. In the study design, we deployed self-developed sample services on the service platforms to gain a deeper insight into the control and release methods as well as on the service platforms' configurations. We complemented the field studies' findings with an analysis of the service platforms' terms and conditions (Scholten et al., 2011).

  Design Science Iteration No. 3: Distinction of different modes of enforcing and incentivizing control as a means to steer network effect behavior.

- *Execution of field studies on the research service platform and e-market pilot* "AGORA." Within the AGORA project, we analyzed the value and effect of explicit and implicit feedback methods to improve service quality (May, Scholten & Fischer, 2011). The monitoring of experimental test consumer behavior allowed retrieving necessary feedback on consumer self-organization that was incited by feedback.

  Design Science Iteration No. 4: Intervention design and evaluation of selected feedback mechanisms.

- *Refinement and situated implementation*. The critical discussion of an evolved solution led to a connection with the control modes suggested by Kirsch (1997), which we have described in a situated implementation of the artifact (Scholten, Schuster & Tai, 2012). We have also identified the underlying constructs as well as a method (Scholten, Janiesch & Rosenkranz, 2013) and proposed a comprehensive specification for a conceptual modeling language (Scholten, 2013).

  Design Science Iteration No 5: Refinement and consolidation of results toward a graphical design method; generalization toward an ISDT.

Appendix B. Detailed Design Rationale for Design Principles

Table B1. Detailed Design Rationale for Design Principles to Design Service Platforms for Network Effects

No.	Design principle	Construct	Reason	Justification	Alternatives	Trade-offs	Decision
1	Provide the method with a technique to conceptualize all specific and unspecific participants and participant groups relevant to the service platform for users to distinguish all sources that can have transactions with activities on the service platform or that can be influenced by or influence other participants or participant groups.	Participant / participant groups	Ability to distinguish sources with small and infinite capacity	Any service platform will require sources to incite network effects	(1) Consider only participants, (2) consider multiple participant (group) roles such as consumer, supplier, etc.	(1) Simpler structures vs. loss of information regarding the ability to sustain network effects (as a group), (2) more guidance about available participant constructs vs. increased complexity and new issues, e.g. when a participant is a consumer and supplier at the same time	Distinguish participants and participant groups
2	Provide the method with a technique to conceptualize all activities on the service platform for users to distinguish all stocks that can have transactions with other activities or participants or can influence participants.	Activities	Ability to distinguish stocks that deplete and replenish	Any service platform will require stocks to incite and maintain network effects	None	None	Include activities
3	Provide the method with a technique to conceptualize influences and transactions on the service platform and the platform ecosystem for users to distinguish interactions between participants, participant groups, and activities, respectively.	Influences / transactions	Ability to distinguish flows and auxiliary variables	Any service platform will require flows to incite and maintain network effects, influence of auxiliary variables should not be neglected	Consider flows and auxiliary variables as one kind of input or output of participants and activities	Simpler structure vs. loss of information and less differentiated means to place control methods	Distinguish flows as transactions from auxiliary variables as influences
4	Provide the method with a technique to design causal loops on the service platform and the platform ecosystem for users to make explicit possible network effects involving participants, participant groups, and activities.	n/a	Ability to design causal loops consisting of the above constructs	Causal loops are a concatenation of the above constructs. Hence, we did not introduce another construct	Explicitly include direct and indirect one- and multisided causal loops	More guidance vs. providing constructs that may not be necessary for all cases and hard to integrate with transactions and influences	Causal loops are a design decision and should be built-on and constructed from the existing circumstances rather than being a construct of their own. Nevertheless, their design requires guidelines as it is a complex problem

5	Provide the method with a technique to conceptualize a control area for users to distinguish the section in which the platform operator can exert full platform authority through enforcement.	Control area	Ability to distinguish extent of full stakeholding power	Demarcate the boundaries of the service platform, distinguish internal participants under full control from those under limited stakeholding power that can only be influenced	Do not distinguish control area and influence area	Lack of ability to distinguish internal participants from external participants and loss of clear demarcation of own platform boundaries (i.e., opportunities to exert full stakeholding power)	Distinguish a control area from other areas of less or no control
6	Provide the method with a technique to conceptualize an influence area for users to distinguish the subsection of a platform ecosystem where the platform operator can only exert limited platform authority through incentives.	Influence area	Ability to distinguish extent of limited stakeholding power due to self-organization of ecosystem	See DP5	(1) See DP5, (2) do not distinguish influence area and noise area	(1) See DP5, (2) lack of ability to distinguish which participants can be influenced in the future even though they are not influenced today	Distinguish an influence area from the control area and the noise area
7	Provide the method with a technique to conceptualize a noise area for users to distinguish the subsection outside the platform ecosystem where the platform operator has no platform authority and no form of control.	Noise area	Ability to distinguish an area where the platform operator has no stakeholding power	Necessity of an area where all other constructs can be displayed	None, as the noise area is essentially the empty canvas of a drawing board	None	Consider the canvas as noise area
8	Provide the method with a technique to place and activate prescriptive control methods on internal participants, who are in hierarchical subordination to the platform operator's authority for users to monitor and steer their sets of actions.	Controls on participants	Ability to prescribe behavior	The platform operator needs to be able to instruct internal participants	Include (1) additional or (2) fewer control methods	(1) Does not apply since restrictive control is a filter to verify compliance with service platform provisions, this is true by definitions for activities of the platform operator, also sanctional control does not apply to internal participants but only to activities, the same holds true for incentivizing controls methods, (2) would negate the ability to use stakeholding power at all	Apply prescriptive control methods to participants

9	Provide the method with a technique to place and activate all but restrictive control methods on activities for users to regulate their interactions by giving prescriptions, monitoring, and possibly intervening or incentivizing and influencing.	Controls on activities	Ability to enforce and incentivize behavior on stocks	The platform operator needs to take control of his stocks and harness them	See DP8	(1) See DP8 on restrictive control, (2) would decrease the ability to differentiate controls methods	Apply all but restrictive control methods to activities
10	Provide the method with a technique to place and activate restrictive control methods on transactions for users to regulate the inflow (resp. outflow) into activities and thus ensure compliance.	Controls on transactions	Ability to restrict flows	The platform operator needs to have control over transactions on its service platform	See DP8	(1) Sanctional and prescriptive control do not apply to transactions, the same holds true for incentivizing control methods (2) see DP8	Apply restrictive control methods to transactions
11	Provide the method with a technique to place and activate incentivizing control methods on influences for users to regulate the feedback into the platform ecosystem.	Controls on influences	Ability to actively incentivize behavior	The platform operator needs to take control of auxiliary variables and harness them	See DP8	(1) Enforcing control methods cannot be applied since the stakeholding power is limited to influences, (2) see DP9	Apply all incentivizing control methods to influences

Appendix C. Metamodel and Language Concepts

The language specification as presented here references the following normative, dated documents: OMG Unified Modeling Language (UML) for metamodeling (Object Management Group Inc., 2015) and the OMG Object Constraint Language (OCL) (Object Management Group Inc., 2014) for metamodeling. A full specification is available at Scholten (2013).

Scope

The conceptual modeling language focuses on essential aspects of the service platform design process. It provides a language and process support to platform architects and solution managers to model a service platform's surrounding business ecosystem (e.g., service providers, consumers, competitors) and to model suitable structures and control methods to harness the service platform's network effects. It further allows the evaluation of design alternatives. It explicitly includes the management of service provisioning and consumption. The scope is purely at the executive level and does not include board-related governance task such as corporate strategy formulation.

Abstract Syntax (Metamodel)

The abstract syntax gives a high-level definition of syntax, leaving out particularities to technical implementation, but it is precise enough to describe representation of and production rules for actual utterances (i.e., of graphical models). It builds on functional design specifications, derived from the design principles and guidelines discussed in Section 4. It complements those with specifications from theoretical design concepts as described in Section 3.

The present work uses the UML as a metalanguage to display the abstract syntax' assembly rules. Syntax needs to additionally prescribe the adaptation of graphic representation in function of specific conditions—for example, the case-dependent representation of the control points on nodes and edges. Therefore, OCL complements UML to prescribe adaptations of the graphical representation in design time and in function of the context-specific syntactical requirements.

In the UML metamodel (Figure C1), classes give the metaview on the element nodes and edges, related and conjugated through production rules. The classes encapsulate characteristics, immanent to those elements as attributes. Some attributes define basic properties immanent to every element (i.e., identification number (id), name, and documentation). There are further attributes that are only carried by specific elements (i.e., base-value, controllable, controllableSource, controllableTarget, scalability, location, and provisions). An important property to a subset of nodes and edges in the context of service management is the control methods (i.e., prescriptive control, sanctional control, restrictive control, informative control, market-regulative control, and motivational control). The metamodel depicts the control methods not as properties but as classes. First, this expresses that the control methods belong to the control center ProtagonistControl in the metamodel. ProtagonistControl is a generalization to all control methods. Second, this approach emphasizes the fact that specific control methods can be aggregated by several elements. Third, it gives the language the possibility to mature over the next releases. The control methods are key. The isolated modeling allows language engineers to evolve them over time, for example, through specific new properties, dependencies, or association relationships (e.g., aggregations or compositions). For instance, a subsequent release of the specification could equip market-regulative control with the options reputation system and recommender system, potentially regulated through constraints, specified in OCL. Another option would be to add and update all emerging elements of recommender and reputation systems in the market to the modeling language.

The present work begins class names with an upper-case letter and attributes with a smaller lower-case letter. OCL conditions under a class name describe conditions for a class instantiation. For example, `Activity {{OCL} self.location = controlArea}` describes that it is a necessary condition for an activity to be located within the control area. OCL conditions stated behind an attribute describe conditions when an attribute is applicable. For example, `controllable: Boolean {{OCL} self.location = controlArea}` defines that an element only carries this attribute if it is located within the control area. Table C1 details the metamodel and the language concepts.

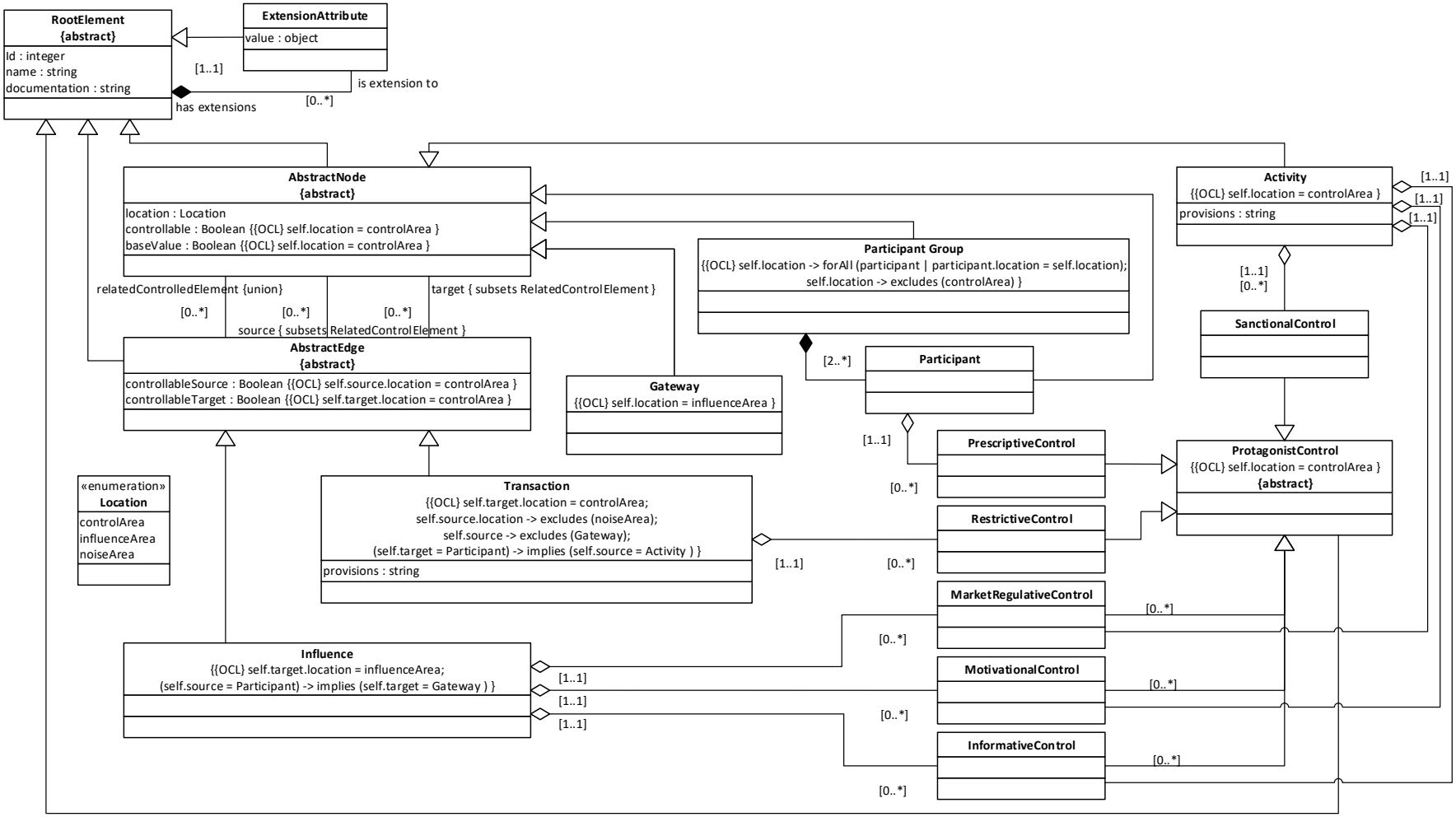
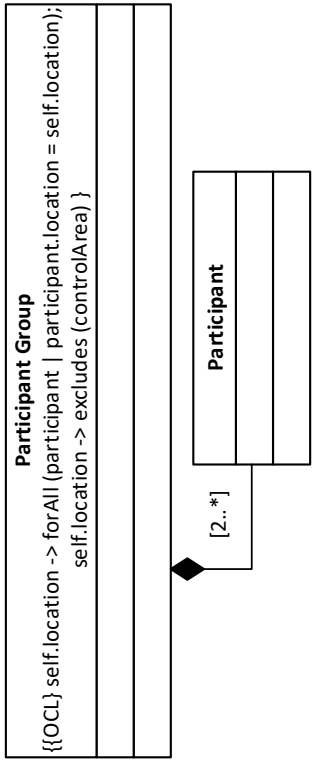
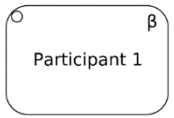
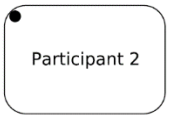
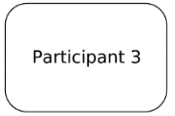
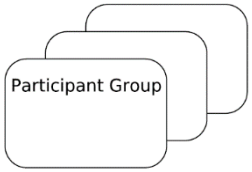
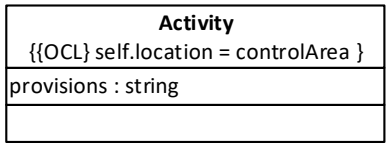
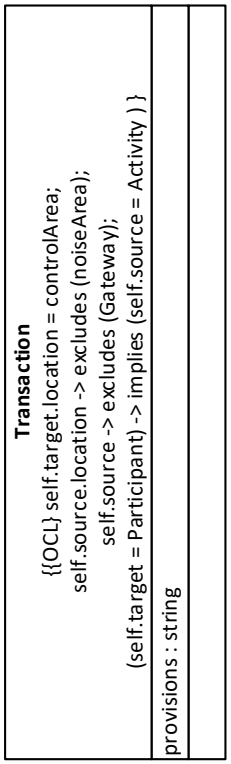
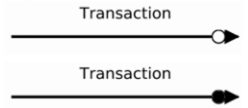


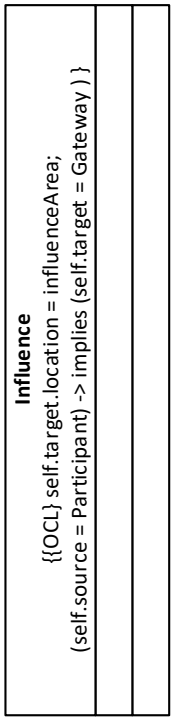
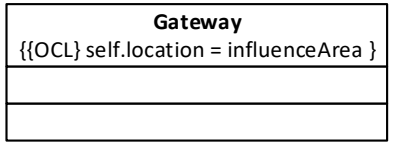
Figure C1. Metamodel

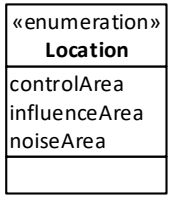
Table C1. Metamodel and Language Concepts

Conceptual language aspect	Linguistic action and statement (semantics and pragmatics)	Metamodel component (syntax, cf. Figure C1)	Notation element (representation symbol), concrete syntax
Participant	A participant is an individual or entity with small capacity (small stock or without stock behavior) within a control, influence, or noise area. Participants are considered static in the short-term view. They may not have any relationship at all with the service platform. However, in that case, they may want to influence those ecosystem participants who are or might be in a relationship with the service platform. A participant representing a group of ecosystem participants or internal participants may influence other ecosystem participants. Ecosystem participants may be the source of transactions into the service platform. A specific ecosystem participant may be in a transactional relationship with the service platform. Some workflows may flow from points of interaction to specific internal participants and then to an activity. An internal participant can be the origin of an endogenous variable, starting off a network effect. Participants are considered static in the short-term view. This does not rule out linear evolution (e.g., the entity service development hires new personnel and/or develops new services). Service platforms include the attribute base value. A base value is placed on and can be set true for participants within the control area. Base values of a platform ecosystem are those values that the modeler considers valuable enough to incite a network effect. Base values may vary during different modeling stages and depend on the goal to be accomplished. A participant further carries the attribute controllable if it is located in the control area. It is symbolized by a circle, depicted in the element's representation. In that case, it can carry one or more control mechanisms of prescriptive control. The controllability of internal participants originates from their subordination to the platform operator. A participant within the control area can be internal entities such as departments or workgroups, but also external suppliers that work on contractual assignment. Details may be textually formulated or referred to in the form of a document identification number or hyperlink. A base value on a participant may be required to start a causal loop.		 Participant 1 (controllable, no control mechanism placed, base value activated)  Participant 2 (controllable, one or more control mechanisms placed, no base value)  Participant 3 (uncontrollable, therefore no option for base value)

Participant group	A participant group is a group of individuals or of entities of finite large size within influence or noise areas. Participant groups may not have any relationship at all with the service platform. However, in that case, they may want to influence those ecosystem participants who are or might be in a relationship with the service platform. A participant group representing a group of ecosystem participants may influence other ecosystem participants and/or be in a transactional relationship with the service platform. Groups of ecosystem participants may be the source of transactions into the service platform.		 Participant group (uncontrollable, therefore no option for base value)
Activity	An activity is a variable stock, describing the magnitude and kind of interaction of participants and participant groups within the control area. Activities can represent a base value to incite network effects. They are points of interaction in the service platform that accumulate value. Value flows may exist from ecosystem participants and participant groups as well as from participants and activities to the service platform. An activity can be part of a causal loop, starting off a network effect. Positive modeling objectives means that only accumulating, but not depleting activities (stocks) of value are in the focus of the modelers' interest. Activities (e.g., service development, service consumption, or service deployment) represent a stock that has accumulated in the past and which might increase, stagnate, or decrease in quantity in the future. Since, by definition, an activity can only take place in the control area, the symbol mandatorily carries the circle in the upper left corner, as it is always controllable. An activity can carry a finite number of control mechanisms of prescriptive control, sanctional control, informative control, and market regulative control. If one or more of them are activated, the circle is filled. Activities also carry the attribute provisions to describe or refer to applicable terms and conditions. This attribute however is not visible and requires a modeling environment with the option to visualize it in a configuration panel or drop-down menu.		Activity 1 (no control mechanism, no base value)  Activity 2 (one or more control mechanisms placed, base value activated) 

Transaction	A transaction describes a value flow into, within, or from the control area. Their source is either in the influence or in the control area. Transactions represent flows of value into points of interaction in the service platform as well as flows within the service platform. They have no interaction with participants or participant groups that have no relation to the service platform. Transactions from such external participants or participant groups exclusively flow into places of interaction. Cooperation with a participant takes place in those places of interaction. Transactions may describe workflows within a control area. Positive modeling objectives means only accumulating, but not depleting activities (stocks) of value are in the focus of the modelers' interest. Transactions mandatorily carry the attribute of controllability, which becomes true if one or more restrictive control mechanisms are activated. Transactions also carry the attribute of provisions to describe or refer to applicable terms and conditions. This attribute, however, is not visible and requires a modeling environment with the option to visualize it in a configuration panel or drop-down menu. Transactions are displayed as arrows with a continuous line. The direction is indicated at the target side through a circle. A transaction must be depicted with a label "Transaction," first letter capital, all other letters small. Similar to the controlled elements, a circle is placed at the target side on the symbol.		Transaction without activated control mechanism and transaction with activated control mechanism 

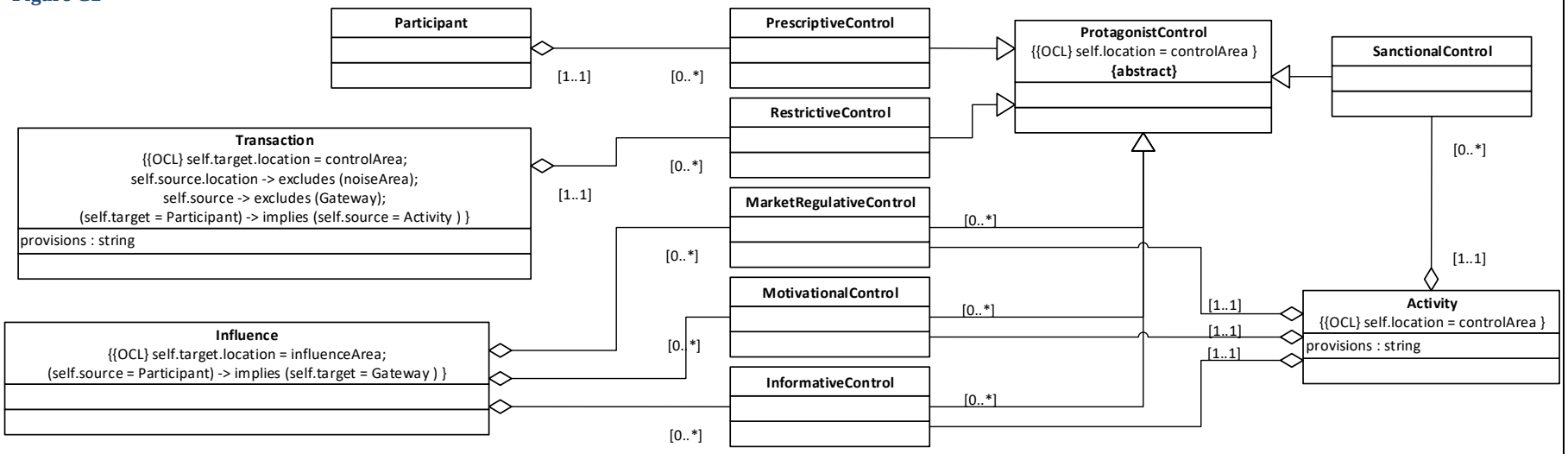
Influence	An influence is a means to stimulate the rate of value flows at their sources. Influences impact ecosystem participants or participant groups through stimulation toward a changed value flow into the service platform (increased or reduced). One source can address several targets, and influences may be the aggregation of influences from several sources. The focus of consideration is the initiation of network effects. Causal loops without any activity or without any participant group included respectively cannot cause network effects. Influences between participants with sources in noise and influence are beyond the platform operator's control. Influences carry a controllability attribute to show whether they can be controlled. The relevant Boolean status is defined by the location of the influences' source. If the source of an influence is located in the control area, the controllability attribute is true and the element carries the circle depicting controllability. In such cases, the influence can carry a finite number of control mechanisms of motivational control, informative control, and market regulative control. If one or more of them are activated, the circle is filled. Influences are displayed as arrows with a continuous line. The direction is indicated at the target-side through a filled arrow head. An influence must be depicted by the label "Influence," first letter capital, all other letters small. Similarly, to the controlled elements, a circle is placed at the source side of the edge whenever the attribute controllability is set to the value of "true."		An uncontrollable influence, a controllable Influence without activated control mechanism, and a controllable influence with activated control mechanism 
Gateway	A gateway is a merging gate, consolidating the impact all incoming edges before triggering one or more outgoing edges. It operates on transactions and influences.		The merging gateway symbol is also used in a similar fashion in process modeling. 
Control area	Control area is the area where the platform operator can exert control over activities and internal participants, over all their own infrastructure and services, and over third-party service in the frame of contractually agreed legal frame set. It is also the area from which it influences ecosystem participants that are placed outside the control area. The control area needs to be modeled within the influence area.		

Influence area	The influence area is the area where participants are located that are in or may come into value-exchanging relationship with the service platform. Ecosystem participants within this area may be influenced by the platform operator, but also by other participants within the ecosystem or outside. This area is out of scope for control activities. The influence area needs to be modeled within the noise area.		
Noise area	The noise area embraces all areas outside the platform ecosystem. Whereas the platform operator cannot exert any stakeholding power, the participants in this area can influence the ecosystem participants in the influence area. No value flow happens between noise area and control area.		The noise area does not have an explicit representation. The basic canvas of the modeling environment should be considered the noise area.
Protagonist control	Control points carry control mechanisms on participants, activities, transactions, and influences to manipulate their progression. Activities carry the enforcing mechanisms of prescriptive and sanctional control. They further accommodate market-regulative control, informative control, and motivational control. Participants may just carry prescriptive control; transactions are limited to restrictive control. Influences can work with market-regulative control, informative control, and motivational control.		We model from a specific protagonist's point of view. A complete model with participants, activities, and influences is interpreted as a control center for the protagonist. For the sake of clarity, we represent control mechanisms as aggregated control points, displayed only by a circle on specific instances of participants, activities, or influences. In an editor, configuration panels or drop-down menus display this as configurable attributes.
Prescriptive control method	Prescriptive control is the sequence of observing and steering a participant's set of actions within activities as well as of internal participants. For actions in activities, it may further include subsequent corrective measures on their results through the platform operator. For example, the platform operator can limit freedom of service providers by prescribing specific development languages and tools or adherence to other standards. As such, prescriptive control is the mechanism used to control participants and activities inside a control area. Prescriptive control means that the platform operator can fully prescribe the steps to take in activities and participants on the service platform. Details and related provisions may be textually described or referred to in the form of a document identification number or hyperlink.		

Sanctional control method	Sanctional control describes the enforcing action of the platform operator on policy breaches in activities. This happens through an escalation routine, including discovery processes, scope, and time of reaction for the participant, and range of enforcements through the platform operator. For example, the platform operator can specify potential restriction of service providers' and services' access to service platform functionality, termination of agreements, or any other necessary action in case of infringement. As such, sanctional control incites an escalation routine at any time an activity exhibits a certain level of incompliance with policy or regulations of the platform operator. Details and related provisions may be textually described or referred to in the form of a document identification number or hyperlink.	*See Figure C2 at the bottom of table	
Restrictive control method	Restrictive control is a filter mechanism on transactions placed within the control area and verifying compliance with service platform provisions. For example, when service providers supply a service to the service platform, then the service platform compares the service with the articulated provisions. In case of noncompliance, the service platform refuses access and potentially asks for amendment. As such, restrictive control is the mechanism to control inbound transactions from third parties (e.g., customers, suppliers). For example, restriction may be based on the compliance level of an inbound transaction to the service platform provisions. Details and related provisions may be textually described or referred to in the form of a document identification number or hyperlink.		
Market-regulative control method	Market regulative control is driven by participants. It gives explicit feedback to consumers or service providers in the service platform and/ or in the ecosystem on value, offered in activities or through participants. This incites a self-regulatory process. Optionally, it may provide a second layer sub-categorization into two types of collaborative feedback systems: recommender systems (collaborative sanctioning) and reputation systems (collaborative filtering). As such, market-regulative control uses feedback from consumers to exert control (i.e., through reputation and recommender systems). Details may be textually described or referred to in the form of a document identification number or hyperlink.		

Informative control method	In informative control, the platform operator preprocesses information and addresses it to existing or potential participants or participant groups. The analyses are customized on the addressed participants or participant groups and have the goal to incite a self-regulatory process among them. For example, potential customers can be intrinsically motivated through the available information on existing services. As such, informative control gives the suppliers (customized) information on consumer preferences, requirements, etc., and aims at supporting the platform operator toward an optimization of their services and eventually of the whole service portfolio. Details may be textually described or referred to in the form of a document identification number or hyperlink.		
Motivational control method	Motivational control aims at steering ecosystem participants toward the accomplishment of specific outcomes through rewards. Motivational control can be triggered through monetary or nonmonetary rewards. For instance, an example of monetary rewards would be seed funding for specific participants. As such, motivational control groups all incentivizing activities toward participants within a control area. Details may be textually described or referred to in the form of a document identification number or hyperlink.		

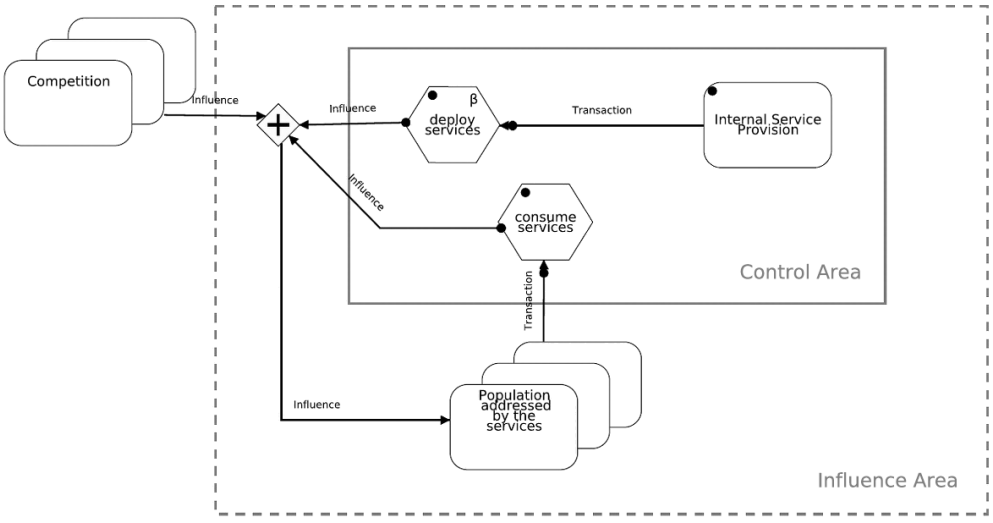
***Figure C2**



Appendix D. Causal Loop Design Patterns

The design for causal loops is not as simple as using one construct such as putting competitors in the noise area but it requires the interplay of multiple constructs. In the following, we present two typical network effects patterns: a direct single-sided network effect similar to customers using the telephone and an indirect two-sided network effect typical for service platforms such as the one of M-Engineering, which considers the interdependency of customer and supplier behavior.

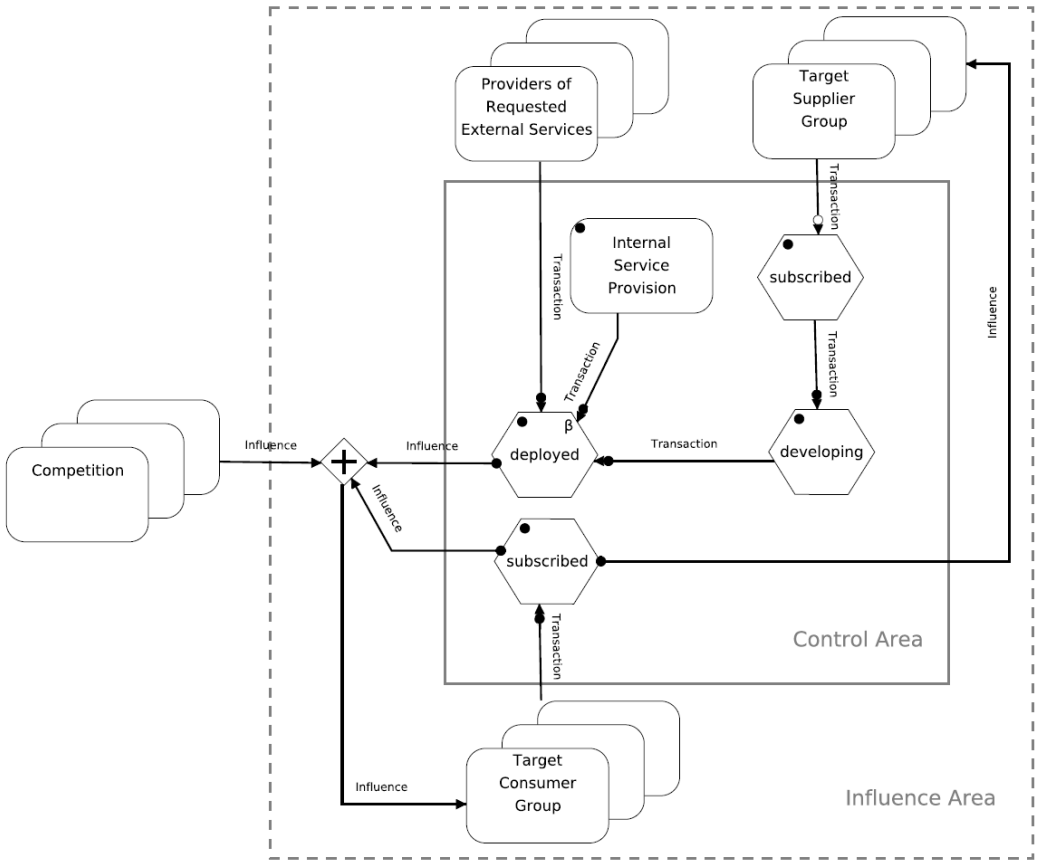
Table D1. Direct Single-Sided Network Effect Pattern

Pattern name	Demand-sided direct network effect
Intent	This pattern provides a solution for network effects that shall be exploited to grow the consumer base.
Applicability	The solution can be applied in single-sided and multisided service platforms. The precondition is a deployment environment, scalable enough to cater to the potentially accomplished dynamic growth of service consumption through a demand-sided network effect. Applying it through collaborative scenarios created cases, requiring small critical masses, starting at a magnitude of two.
Solution	Members of a potential user group subscribe and consume services that were deployed by the platform operator. Regarding it as a dynamic system, the activity consume services has the function of a stock. The more subscribed users consume, the more this stock is filled. The activity deploy services also acts as stock. This activity represents the base value, which initially sets off the system (indicated by the β-symbol). The quantity of users can motivate—together with a quantity and quality of deployed services—new potential users to subscribe to the service platform. This motivation is weakened by competitive offers. The pattern channels explicit effort on positive user influence on the addressed population through the application of several control mechanisms in the context of a causal loop: On the influence, linking the consume activity and the gateway, platform operators apply informative and motivational control. Informative control amplifies the impact of the size of the user group. The nature of communicated information may vary. However, service platforms commonly communicate the subscribed number of users. Platform operators may use motivational control in various shapes. The transaction pointing from the user group to the subscribe activity applies restrictive control. That is, the user shall be required to accept the service platforms' terms and conditions. Within the consume activity, the platform operator can exert sanctional control; that is, it may make use of its right and the power to exclude users. The deployment of services in the described pattern is an internal activity, therefore limited to prescriptive control and sanctional control. Exerting prescriptive control means to manage the deployment environment. Sanctional control stands for the power to un-deploy a service. The platform operator exerts prescriptive control on the participant internal service provision (e.g., in response to results from reputation systems) and may exert restrictive control on the transaction leading from internal service provision to deploy services. The influence, pointing from the deploy activity to the gateway includes the control mechanism of informative control. Informative control implies clear and targeted information on the product. The activity consumes needs to be placed in a scalable environment, to be able to respond to rapidly growing consumption. Strong growth behavior is realistic, as the activity is placed within a loop (demand-sided network effect).
Diagram	

Frequent features	In many cases, the influence, pointing from the deploy activity to the gateway also includes motivational control and market regulative control. Market regulative control is applied through reputation mechanisms to reduce the entry barrier and to give decision support. The activity deploy services can be replenished by specific participants, representing the service development departments on the service platform. Linking edges are transactions.
Consequences	The pattern is reduced to the core features of a service deployment. It does not consider any complex feature, for example, replication or synchronization. Those need to be added in a contextualized approach.
Examples	Core business of social networks such as Facebook, Google+, Dropbox for consumers

Table D2. Indirect Two-Sided Network Effect Pattern

Pattern name	Cross-sided indirect network effect through service consumption and third-party supply
Intent	This pattern provides a solution for network effects that shall be exploited to grow the user base and service base.
Applicability	The solution can be applied in multisided service platforms. The precondition is a deployment environment, scalable enough to cater to the potentially accomplished dynamic growth of service consumption through a demand-sided network effect and service deployment through supply-sided network effects.
Solution	Members of a potential user group subscribe and consume services that originate from the platform operator and service providers. Regarding it as a dynamic system, the activity consume has the function of a stock. The more subscribed users consume, the more this stock is replenished. The activity deploy services also acts as stock. This filled stock of services represents the base value, which initially sets off the system (indicated by the β-symbol). The quantity of users can motivate—together with a quantity and quality of deployed services—new potential users to subscribe to the service platform. This motivation is weakened by competitive offers. The pattern channels explicit effort on positive user influence toward the addressed population through the application of several control mechanisms in the context of a causal loop. Members of the targeted consumer group subscribe and consume services that were deployed by the platform operator. Regarding it as a dynamic system, the activity subscribed has the function of a stock. The more subscribed users consume, the more this stock is filled. The transaction pointing from the user group to the subscribe-activity applies restrictive control. That is, the user is required to agree to the service platform's terms and conditions. Within the subscribed activity, the service platform shall exert sanctional control; that is, it may make use of its right and the power to exclude users. On the influence, linking the subscribed-activity and the gateway, the service platform applies informative and motivational control. Informative control amplifies the impact of the size of the user group. The nature of communicated information may vary. However, service platforms commonly communicate the subscribed number of users. Platform operators may use motivational control in various shapes. The influence point from subscribed services to the target supplier group includes two control mechanisms: informative control and market regulative control. Deployed services are of internal and external origin. Services of internal origin are limited to prescriptive control and sanctional control. Exerting prescriptive control means to manage the services. Sanctional control stands for the power to un-deploy a service. The platform operator exerts prescriptive control within the participant internal service provision (e.g., in response to results from reputation systems) and may exert restrictive control on the transaction leading from internal service provision to deploy services. The influence, pointing from the deployed activity to the gateway includes the control mechanism of informative control. Informative control implies clear and targeted information about the services. The activities subscribed and deployed need to be placed in a scalable environment, to be able to respond to rapidly growing consumption. Strong growth behavior is realistic, as the activity is placed within a loop (demand-sided network effect).

Diagram	
Frequent features	In many cases, the influence, pointing from the deploy services activity to the gateway also includes motivational control and market regulative control. Often, suggestions are used—for example, other users applying this service also applied XYZ. Market regulative control in this case stands for the use of reputation mechanisms to reduce the entry barrier and to give decision support.
Consequences	This pattern couples two causal loops and enforces a cross-sided network effect. Quality control functions need to be properly in place to harness positive effects but also to avoid the negative scenario of negatively accelerating loops, for example, initiated through the provision of low-quality third-party services.
Examples	Salesforce, NetSuite

Appendix E. Service Platform Design Patterns

The design of or for any ability of a service platform requires the use of several design principles, combined with applying best-practice guidelines. As the design of service platforms typically is a bespoke activity, it is unreasonable to assume a finite list of patterns. As an illustration of best-practice knowledge, we present three core requirements for most service platforms: user subscription, service development, and service deployment. They illustrate the combined use of actors, processes, and controls to form causal loops. These patterns can be combined with those from Appendix D.

Table E1. Service Platform Subscription Pattern

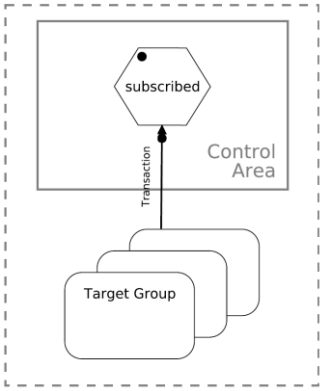
Pattern Name	Service platform subscription
Intent	This pattern provides a basic approach for managing subscription to a service platform.
Applicability	The solution is suitable for most service platforms. It helps onboarding and maintaining subscribers selectively. The subscribers can be service consumers, service providers, or both.
Solution	Members of an addressed user group subscribe and consume services. The transaction pointing from the addressed target group to the activity subscribe is equipped with restrictive control. That is, the user has to subscribe to the services and to accept the service platform's service provisions (or a subset thereof). In the subscribed activity, the service platform exerts prescriptive control. The activity allocates a limited amount of freedom to consumers. Within the activity subscribed, the service platform also exerts sanctional control. That is, it retains the right and the power to exclude users.
Diagram	
Frequent features	The provisions in transaction and activity often include the following: contractual agreement (acceptance of service platform's terms and conditions) and request for consumer details (address, payment details).
Consequences	The pattern is reduced to the core features of subscription. It excludes any application-specific feature, for example, geographical or market segment-specific distinctions due to national law or mentality. Those need to be added when applying the pattern.
Examples	Facebook, Salesforce, NetSuite

Table E2. External Service Deployment Pattern

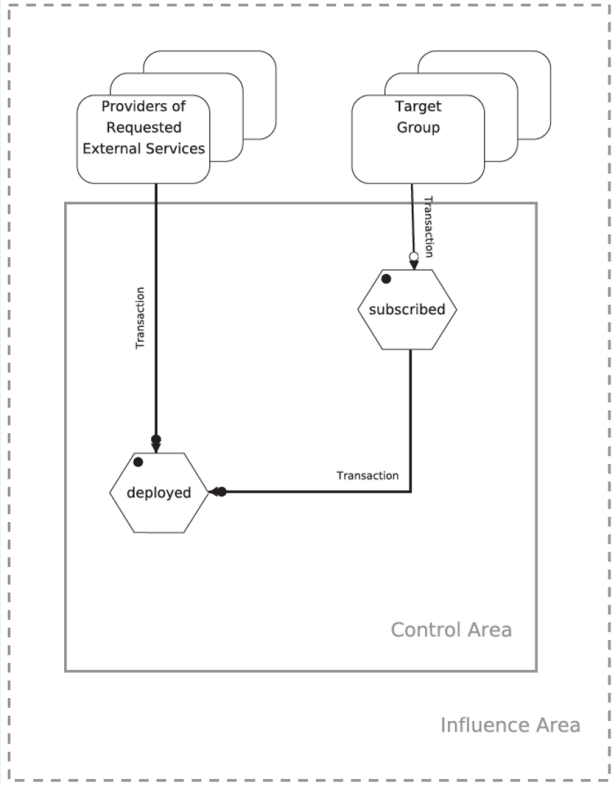
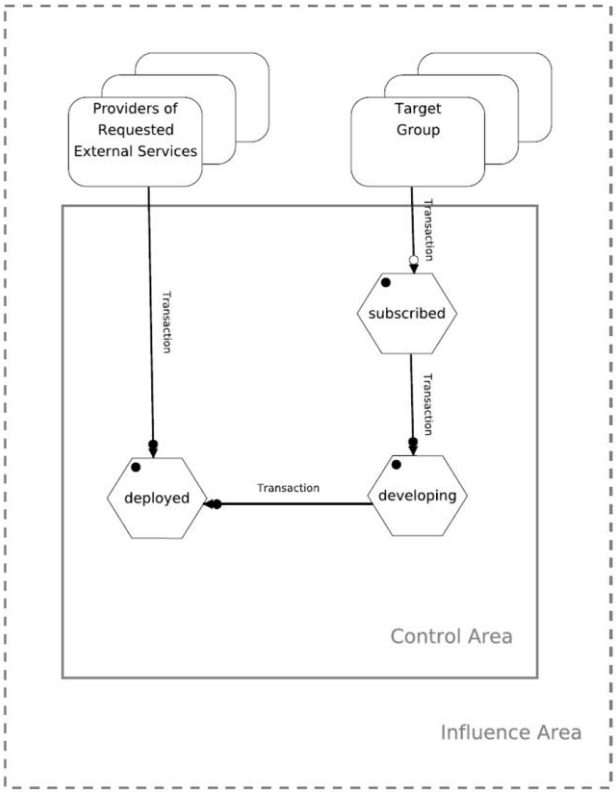
Pattern name	Basic external service deployment pattern based on programming specification
Intent	This pattern provides an approach for managing third-party service deployment into a service platform.
Applicability	The solution is suitable for all service platforms. It helps onboarding and managing services.
Solution	Subscribed participants of a service provider group (participant target group) provide services for deployment on a service platform. Restrictive control on the transaction pointing from the activity subscribed to the activity deployed filters the infeed of services in the function of compliance with the service platform service provisions—that is, service design in compliance with the programming specification but also to legal requirements. The activity deployed includes sanctional control, which regularly verifies compliance with the service platform service provisions and initiates escalation routines. In addition, it includes prescriptive control to manage the deployed services. External services outside the control area requested from services deployed in the control area are subject to restrictive control, as their execution cannot be observed directly.
Diagram	
Frequent features	Some platform operators include a sandbox into the activity deployed to test a service before deploying. The inclusion of such a sandbox as a construct is useful if it is exploited in causal loops—for example, to attract early adopters in a testing phase.
Consequences	The programming specification limits the service providers' scope of freedom with respect to reuse of services on other service platforms. Consequently, service providers could be tempted to produce simple requestors, programmed based on the programming specification, calling up external services. Many challenges with respect to service quality could migrate from an upfront filtering through restrictive control to an ongoing verification through sanctional control.
Examples	It corresponds to the approach chosen by Salesforce in their initial stage of expansion of the AppExchange/ Force.com service platform

Table E3. Service Development and Deployment Pattern

Pattern name	Service development with programming environment
Intent	This pattern provides an approach for managing third-party service development and deployment on a service platform.
Applicability	The solution is suitable for multisided service platforms. It helps with onboarding, testing, and maintaining services and service providers selectively.
Solution	Members of a target group provide services for deployment onto a service platform. A programming environment is prepended to the deployment environment, ensuring compliance with related service platform provisions. Apart from legal and administrative aspects, the programming environment on the activity developing ensures service manageability in the subsequent deployed activity through prescriptive control. The activity deployed also includes sanctional control, which regularly verifies compliance with the company policy and initiates escalation routines. Services, requested from outside the control area subject to restrictive control, as their execution cannot be observed directly.
Diagram	
Frequent features	Some platform operators include a sandbox into the activity deployed to test a service before deploying. The inclusion of such a sandbox as a construct is useful if it is exploited in causal loops, for example, to attract early adopters in a testing phase.
Consequences	The level of manageability is higher than in the case of a pure programming specification (see Table E2).
Examples	Prepended programming environments, for example, those used by Salesforce or NetSuite

Appendix F. User Feedback Questionnaire

For the design of a user evaluation questionnaire, Kitchenham and Pfleeger (2002) suggested prepended research of comparable studies to build on past experience. As the design method and software are new artifacts, no previously acquired data are available. Hence, we reverted to the questionnaire design of Wittern and Zirpins (2016), who evaluated modeling and engineering tools for service composition. Their experience with assessing tools contributed to the questionnaire's structure.

This is an accepted methodology for software or design evaluation (Henderson et al., 1995). We note that this approach is not to be confused with the standards of quantitative survey research for behavioral studies. Because of the very limited sample size, a questionnaire-based survey for software or design evaluation can seldom reach the statistical power required in more behavioral research settings.

Table F1. Full User Feedback Questionnaire

Questions
Please rate your experience with the software
I found the features in the software purposeful to model platforms ecosystems and the respective network effects with the software
The graphical user interface (GUI) was intuitive and easy to follow
The GUI was sufficiently comfortable to work with
The software did not restrict me in my modeling efforts
When designing, I benefitted from the software's guidance based on the design method's grammar and its resulting content-related suggestions and restrictions with respect to … … transactions and influences … controllable/ uncontrollable activities and participants … case-specific control mechanisms … areas (control, influence, noise) … getting a plausibility check of my modeling intentions
Please rate your experience with the design method
It was easy to understand the design method to model dynamic networks around service platforms
The design method gives me a better understanding of network effects and control possibilities in platform ecosystems
The design method helps to locate or create dynamic loops
The design method helps to upgrade one-sided loops to two- or multisided loops, meaning to create interconnected loops to structures of two or more loops that directly impact on each other
The context-specific suggestion of control mechanisms ... … always provides me with the most suitable control mechanisms … is unnecessarily limiting my scope of decision making … is difficult to grasp …. helps to design a service platform that influences growth of consumer and provider ecosystem …. helps to design a service platform that focuses on third-party services that fit into the corporate strategy …. allowed me to completely model my corporate platform
The design method allowed me ... … to select the optimal nodes for placing base values … to evaluate whether our company already possesses strong base values … to evaluate whether we still need to create suitable base values … to see—in view of our base value situation—whether it makes sense to enter the platform business
I was satisfied with the ordinal metrics for "Base Value" suggested by the design method
The design method allowed me to think about scalability requirements per division

Please rate your perception on the achievable quality of the models designed with the design method and the software
Correctness
… I was able to produce one or more models where the structure and behavior of the models are consistent with the real world
Relevance
… All relevant elements from the real-world scenario find application in the model representation
… All elements and relationships, applied in and required in the modeling language are relevant to accomplishing its design goal
Economic efficiency
… The model is robust, meaning it remains relevant over time, even when quantities of users, services, etc. evolve.
… The model is adaptable, meaning it can be adapted without big effort to a modified constellation (e.g. to implement strategic change or to respond to environmental change).
Clarity
… The produced model was clear, understandable, and not overloaded
Comparability
… When modeling design alternatives on the same business case, I can compare them as they follow the same grammar.
Systematic design
… The system description from the Analyzer (Analysis view) and the Modeler are consistent
General perception
The design method and the software helped me to produce a better solution than without them
The design-time was shorter when using the software as compared to a design without the design method and software
I was quickly able to understand both design method and software
The concept of control area, influence area and noise area … … allowed me to better position my modeling elements
… supported my understanding of the authority for service management I have, depending on my design decision
The concept of divisions helped me to better structure my model
The concept of division groups helped me to model repeated areas of similar behavior
The following elements helped me to express the interplay of relationships within a service platform … … activity … participant … participant group … transaction … influence

Appendix G. Evaluation Case Descriptions and Survey

During the survey, the participants would have access to the case description relevant to their task and to a notation sheet (i.e., a simplified version of Table C.1) at all times except for during the cloze test. The expressiveness test does not require a case description either. Depending on the technical background of the participants, it may be useful to start with a training exercise to explain network effects and the design method.

Introduction to Network Effects for Students

A network effect is the (positive) effect that an additional user of a good or service has on the value of that product to others. When a network effect is present, the value of a product or service increases according to the number of others using it.

The classic example is the telephone. A greater number of users increases the value to each user. Even though connecting a new telephone user does not intend to create value for others, it does so by joining the network. Service platforms such as online social networks work similarly, with sites Facebook and LinkedIn increasing in value to each member as more users join.

Network effects can create a so-called bandwagon effect once a threshold, the critical mass, has been reached. This suggests that more users want to join the network creating a causal loop. The effect does not have to be same-sided (customer to customer) but can also be cross-sided, meaning that the value for other sides increases as well. For example, more users in the AppStore make the store more valuable for developers. Conversely, more developed apps make the store more valuable for users.

Pre-test Questions: Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Have you ever used text to document a service platform?

   [Yes or No]

2. Have you ever used a diagram to document a service platform?

   [Yes or No]

3. Overall, how familiar are you with the methods to document a service platform.

   [1=not at all familiar ------ 7=Very familiar]

4. Estimate how INTENSIVELY you have worked with relevant methods in the last 4 (four) years.

   [1=not at all intensive ------ 7=Very intensive]

5. Estimate the level of COMPETENCE that you have attained in using methods for service platform documentation.

   [1=not at all competent ------ 7=Very competent]

6. Estimate the level of CONFIDENCE that you have attained in understanding service platform documentation.

   [1=not at all confident ------ 7=Very confident]

Generative Case: Case A/B

Please conceptualize a service platform for the distribution of online games using our design method. Focus on a design that makes network effects more likely to appear and control so that they support the business of the service platform.

(Note that in this case, we asked for a service platform for online games. The type of the service platform should be chosen based on the participant's knowledge about a domain since participants need to construct a model based on their own mental models without specific instructions.)

Generative Assessment: Case A/B

Thank you for completing the task. Now, we have some questions about your recent modeling experience. Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. The model I created is an exact representation of my design goals.

[1=not at all confident ------ 7=Very confident]

2. The model I created will meet the expectations of the model user.

[1=not at all confident ------ 7=Very confident]

Furthermore, we have recorded the time it took you to model the case. In addition, two experts will score the model based on the following two questions:

3. The model is plausible and its design for network effects supports the incitement and management of the aforementioned effects through causal loops and controls.

[1=not at all confident ------ 7=Very confident]

4. The model is free of defects.

[1=not at all confident ------ 7=Very confident]

(If the opinions of the two experts deviate, the model will be discussed and a joint assessment will be made.)

Understanding Case A

This is an example of direct and indirect network effects based on Dropbox's DBX Platform for business users. The DBX Platform provides third-party business services on top of the regular Dropbox storage service, for example, data loss prevention services or document management lifecycle support.

The case shows how Dropbox Business customers positively influence other Dropbox Business customers as well as DBX Platform partners through the application of several control methods in the context of causal loops.

Potential Dropbox Business customers subscribe and consume services that were deployed by Dropbox internal development and third-party service providers (henceforth, DBX Platform partners). The quantity of customers can motivate—together with a quantity and quality of deployed services—new potential customers and DBX Platform partners in the influence area to subscribe to the DBX Platform. This motivation is weakened by competitive offers of other SaaS storage providers in the noise area.

The transactions pointing from the customers and partners to the subscribe activities in the control area apply restrictive control. That is, customers and partners are required to agree to the service platform's terms and conditions. Within each subscribe activity, the service platform exerts sanctional control. That is, it may make use of its right and the power to exclude customers as well as partners. On the influence, linking the subscribe activity and the gateway, Dropbox applies informative and motivational control. For instance, this could be communication about the amount of subscribed customers or an offer to invite new customers. The influence pointing from the subscribe activity to the DBX Platform partners also includes two control mechanisms: informative control and market regulative control.

Deployed services are of internal and external origin. External services may be developed and later deployed by subscribed DBX Platform partners. The DBX Platform can exert prescriptive control means to manage the activities, e.g. API requirements, as well as sanctional control, e.g., to un-deploy a service. The DBX Platform also exerts prescriptive control on the Dropbox internal development (e.g., in response to results from reputation systems) and may exert restrictive control on the transaction between activities within the control area. The influence, pointing from the deploy activity to the gateway includes the control mechanism of informative control. It implies clear and targeted information about the services. Furthermore, the DBX Platform purchases required services from external services providers to run the service platform. These service providers are not influenced by the amount of deployed services or subscribed customers.

The following figure provides a diagrammatic overview using the notation introduced in the main test:

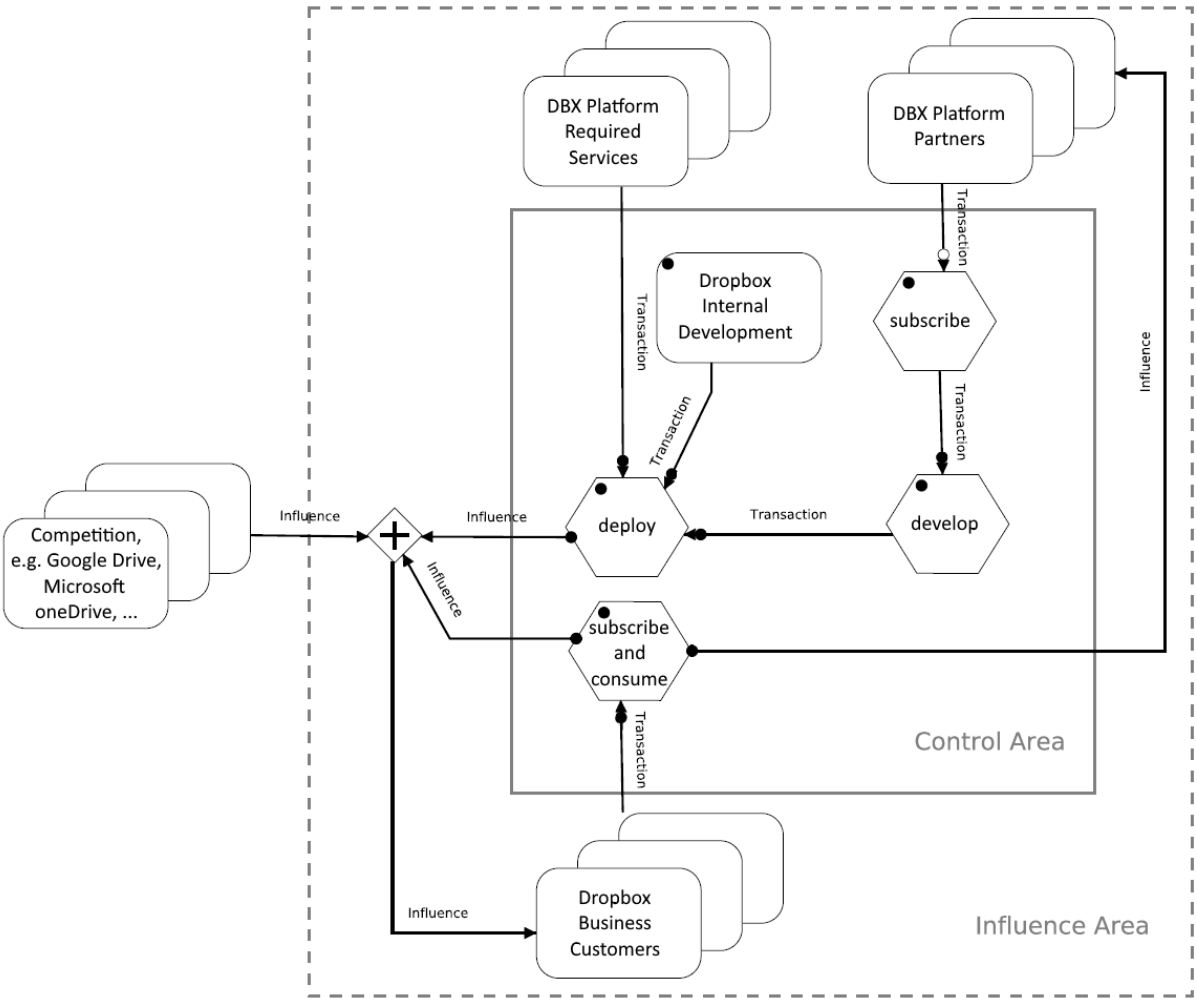


Figure G1. Case A

Understanding Case B

This is an example of direct and indirect network effects based on Dropbox's DBX Platform for business users. The DBX Platform provides third-party business services on top of the regular Dropbox storage service, for example, data loss prevention services or document management lifecycle support. The case shows how Dropbox Business customers positively affect other Dropbox Business customers as well as DBX Platform partners through the application of several control methods in the context of causal loops.

Potential Dropbox Business customers subscribe and consume services that were deployed by Dropbox internal development and third-party service providers (henceforth, DBX Platform partners). The quantity of customers can motivate—together with a quantity and quality of deployed services—new potential customers and DBX Platform partners in the influence area to subscribe to the DBX Platform. This motivation is weakened by competitive offers of other SaaS storage provider in the noise area.

The interactions pointing from the customers and partners to the respective subscribe activities apply restrictive control. That is, the customer or partner is required to agree to the service platform's terms and conditions. Within each subscribe activity, the service platform exerts sanctional control. That is, it may make use of its right and the power to exclude customers as well as partners. On the interaction, linking the subscribe activity and the gateway, Dropbox applies informative and motivational control. For instance, this could be communication about the amount of subscribed customers or an offer to invite new customers. The interaction pointing from the subscribe activity to the DBX Platform partners also includes two control mechanisms: informative control and market regulative control.

Deployed services are of internal and external origin. External services may be developed and later deployed by subscribed DBX Platform partners. The DBX Platform can exert prescriptive control means to manage the activities, e.g. API requirements, as well as sanctional control, e.g., to un-deploy a service. The DBX Platform may exert restrictive control on the interactions between activities. The interaction, pointing from the deploy activity to the gateway includes the control mechanism of informative control. It implies clear and targeted information about the services. Furthermore, the DBX Platform purchases required services from external services providers to run the service platform. These service providers are not affected by the amount of deployed services or subscribed customers.

The following figure provides a diagrammatic overview using the notation introduced in the main text:

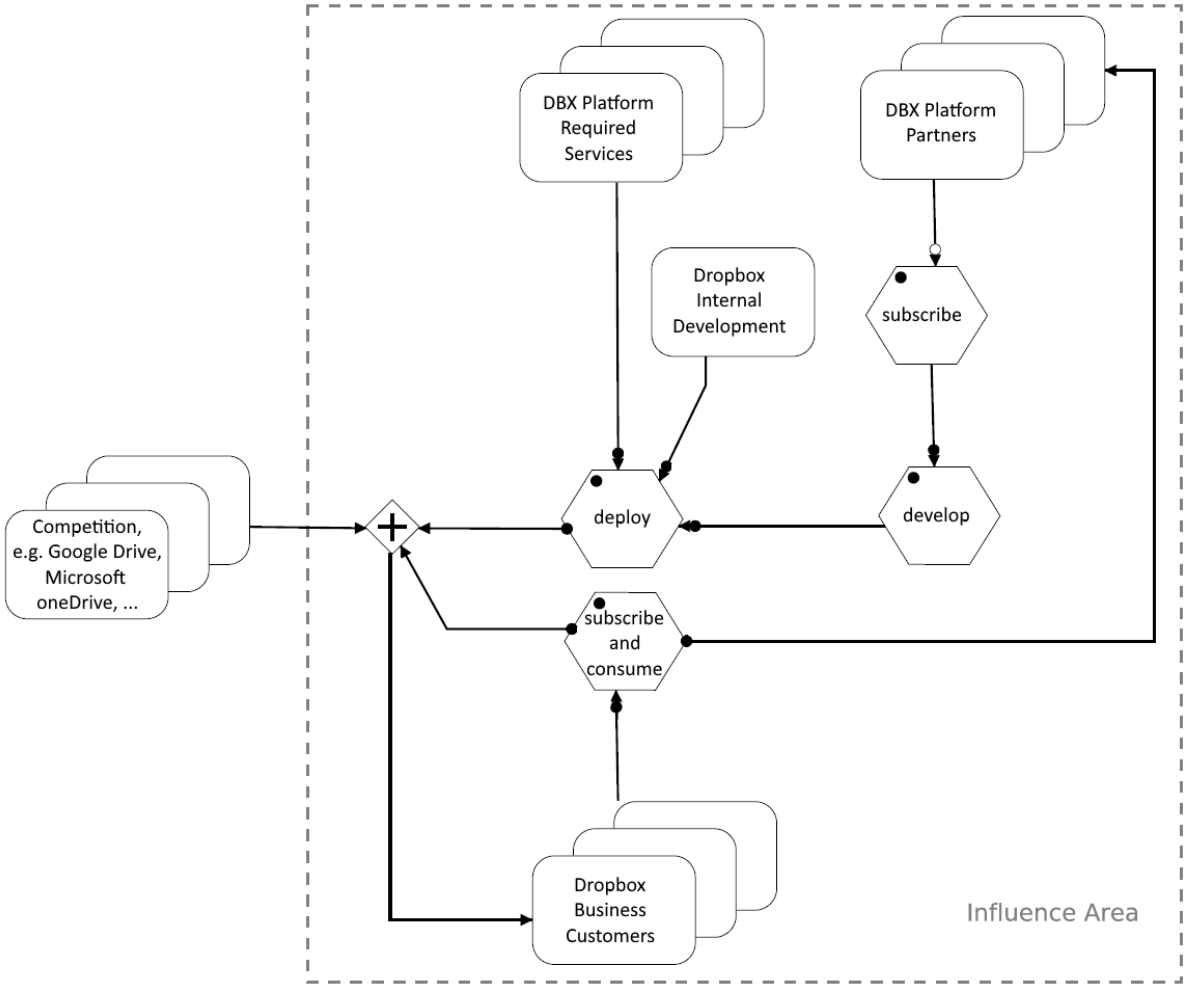


Figure G2. Case B

Retention Questions: Initial – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Over how many activities does the platform operator have control?

   [0 ------ 5, unknown]

2. How many participants can the platform operator influence?

   [0 ------ 5, unknown]

3. How many participants can the platform operator control?

   [0 ------ 5, unknown]

4. Does the documentation depict a service platform that enables third-party services?

[Yes or No or Unknown]

Retention Questions: Search/Recognition – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Is the Dropbox internal development currently actively controlled by the platform operator?

[Yes or No or Unknown]

2. Are the Dropbox Business customers controlled directly by the platform operator?

[Yes or No or Unknown]

3. Are the competitors controlled or influenced directly by the platform operator?

[Yes or No or Unknown]

4. Does Dropbox internal development deploy services?

[Yes or No or Unknown]

5. Do the DBX Platform partners deploy services?

[Yes or No or Unknown]

6. Does the competition influence the deployment of services?

[Yes or No or Unknown]

7. Is a control placed on the deployed activity?

[Yes or No or Unknown]

8. Is a control placed on the interaction of the consume and subscribe activity toward the gateway leading to the customers?

[Yes or No or Unknown]

9. Is a control placed on the interaction of customers with the subscribed activity?

[Yes or No or Unknown]

Retention Questions: Inference – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Could Dropbox internal development theoretically be controlled by the platform operator?

[Yes or No or Unknown]

2. Could the provider of DBX Platform Required Services theoretically be controlled by the platform operator and its services?

[Yes or No or Unknown]

3. Does the deployment of services lead to more deployment of services?

[Yes or No or Unknown]

4. Do Dropbox Business customers influence the subscribe activity connected to DBX Platform partners?

[Yes or No or Unknown]

5. Could the DBX Platform customers be controlled by the platform operator?

[Yes or No or Unknown]

6. Can the combined effect of the interaction of deploy and subscribe activities be controlled by the platform operator?

[Yes or No or Unknown]

Transfer Questions: Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Does the platform operator have the same level of authority over all participants? Please explain.

[Text Field]

2. How can the platform operator gain more control over competitors?

[Text Field]

3. How can you incentivize the customers to have more interaction (subscribe and consume) with the service platform?

[Text Field]

4. How can you remove the influence of competitors on your service platform?

[Text Field]

5. What are the necessary antecedents so that Dropbox internal development can be prescriptively controlled by the platform operator? What has to change (if at all)?

[Text Field]

6. Assume another kind of third-party partner. They would not add services to Dropbox but would use Dropbox in their products to store files as a service. How could they be influenced to add Dropbox to more of their products? How much authority would the platform operator have over them?

[Text Field]

Recall Questions (Cloze Test): Case A

Please fill in the blanks:

This is an example of [*direct and indirect*] network effects based on Dropbox's DBX Platform for business users. The DBX Platform provides third-party business services on top of the regular Dropbox storage service, for example, data loss prevention services or document management lifecycle support.

The case shows how Dropbox Business customers positively [*influence*] other Dropbox Business customers as well as DBX Platform partners through the application of several [*control*] methods in the context of causal loops.

Potential Dropbox Business customers [*subscribe and consume*] services that were [*deployed*] by Dropbox internal development and third-party service providers (henceforth, DBX Platform partners). The quantity of customers can [*motivate*]—together with a quantity and quality of deployed services—new potential customers and DBX Platform partners in the [*influence*] area to subscribe to the DBX Platform. This [*motivation*] is weakened by competitive offers of other SaaS storage provider in the noise area.

The transactions pointing from the customers and partners to the subscribe activities in the control area apply [*restrictive*] control. That is, the customer and partners are required to agree to the service platform's terms and conditions. Within each subscribe activity, the service platform exerts [*sanctional*] control. That is, it may make use of its right and the power to exclude customers as well as partners. On the [*influence*], linking the subscribe activity and the gateway, Dropbox applies [*informative*] and [*motivational*] control. For instance, this could be communication about the amount of subscribed customers or an offer to invite new customers. The [*influence*] pointing from the subscribe activity to the DBX Platform partners also includes two control mechanisms: [*informative*] control and [*market regulative*] control.

Deployed services are of internal and external origin. External services may be developed and later deployed by subscribed DBX Platform partners. The DBX Platform can exert prescriptive control means to manage the activities, e.g. API requirements, as well as sanctional control, e.g. to [*undeploy*] a service. The DBX Platform also exerts [*prescriptive*] control on the Dropbox internal development (e.g., in response to results from reputation systems) and may exert [*restrictive*] control on the transaction between activities within the [*control*] area. The influence, pointing from the deploy activity to the gateway includes the control mechanism of [*informative*] control. It implies the clear and [*targeted*] information about the services. Furthermore, the DBX Platform purchases required services from external services providers to run the service platform. These service providers are not [*influenced*] by the amount of deployed services or subscribed customers.

Recall Questions (Cloze Test): Case B

Please fill in the blanks.

This is an example of [*direct and indirect*] network effects based on Dropbox's DBX Platform for business users. The DBX Platform provides third-party business services on top of the regular Dropbox storage service, for example, data loss prevention services or document management lifecycle support.

The case shows how Dropbox Business customers positively [*affect*] other Dropbox Business customers as well as DBX Platform partners through the application of several [*control*] methods in the context of causal loops.

Potential Dropbox Business customers [*subscribe and consume*] services that were [*deployed*] by Dropbox internal development and third-party service providers (henceforth, DBX Platform partners). The quantity of customers can [*motivate*]—together with a quantity and quality of deployed services—new potential customers and DBX Platform partners in the [*influence*] area to subscribe to the DBX Platform. This [*motivation*] is weakened by competitive offers of other SaaS storage provider in the noise area.

The interactions pointing from the customers and partners to the respective subscribe activities apply [*restrictive*] control. That is, the customer or partner is required to agree to the service platform's terms and conditions. Within each subscribe activity, the service platform exerts [*sanctional*] control. That is, it may make use of its right and the power to exclude customers as well as partners. On the [*interaction*], linking the subscribe activity and the gateway, Dropbox applies [*informative*] and [*motivational*] control. For instance, this could be communication about the amount of subscribed customers or an offer to invite new customers. The [*interaction*] pointing from the subscribe activity to the DBX Platform partners also includes two control mechanisms: [*informative*] control and [*market regulative*] control.

Deployed services are of internal and external origin. External services may be developed and later deployed by subscribed DBX Platform partners. The DBX Platform can exert [*prescriptive*] control means to manage the activities, e.g. API requirements, as well as [*sanctional*] control, e.g. to [*undeploy*] a service. The DBX Platform may exert [*restrictive*] control on the interactions between activities. The interaction, pointing from the deploy activity to the gateway includes the control mechanism of [*informative*] control. It implies the clear and [*targeted*] information about the services.

Furthermore, the DBX Platform purchases required services from external services providers to run the service platform. These service providers are not [*affected*] by the amount of deployed services or subscribed customers.

Expressiveness: Principle Deficiency – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Have you ever had the need to distinguish concepts (such as participants) you can control from those you cannot (e.g., internal teams vs. partners)?

[Yes or No]

2. Does the design method provides sufficient means to distinguish concepts you can control from those you cannot?

[1=not at all confident ------ 7=Very confident]

3. Our design method could be made more complete by distinguishing participants you can control from those you cannot.

[1=not at all confident ------ 7=Very confident]

4. I cannot use the design method adequately to distinguish participants you can control from those you cannot.

[1=not at all confident ------ 7=Very confident]

5. Have you ever had the need to distinguish interactions with participants in transactions and influences?

[Yes or No]

6. Does the design method provide sufficient means to distinguish interactions with participants in transactions and influences?

[1=not at all confident ------ 7=Very confident]

7. Our design method could be made more complete by distinguishing interactions with participants in transactions and influences.

[1=not at all confident ------ 7=Very confident]

8. I cannot use the design method adequately to distinguish interactions with participants in transactions and influences.

[1=not at all confident ------ 7=Very confident]

9. Have you ever had the need to prescribe rules for an internal participant of the platform operator (e.g., escalation routines for a call center agent)?

[Yes or No]

10. Does the design method provide sufficient means to prescribe rules for an internal participant of the platform operator?

[1=not at all confident ------ 7=Very confident]

11. Our design method could be made more complete by adding the ability to prescribe rules for an internal participant of the platform operator.

[1=not at all confident ------ 7=Very confident]

12. I cannot use the design method adequately to prescribe rules for an internal participant of the platform operator.

[1=not at all confident ------ 7=Very confident]

Expressiveness: Principle Redundancy – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Have you ever had the need to represent staged areas of platform authority?

[Yes or No]

2. I often have to choose between means to represent one kind of authority requirement.

[1=not at all confident ------ 7=Very confident]

3. The design method often provides two or more means that can be used to represent the same kind of authority on a service platform.

[1=not at all confident ------ 7=Very confident]

4. Using the design method, one kind of platform authority can often be represented by different means.

[1=not at all confident ------ 7=Very confident]

5. Have you ever had the need to represent interactions?

[Yes or No]

6. I often have to choose between equal concepts to represent one kind of interaction on a service platform.

[1=not at all confident ------ 7=Very confident]

7. The design method often provides two or more concepts that can be used to represent the same kind of interaction on a service platform.

[1=not at all confident ------ 7=Very confident]

8. Using the design method, one kind of interaction can often be represented by different concepts.

[1=not at all confident ------ 7=Very confident]

9. Have you ever had the need to prescribe rules for an internal participant of the platform operator (e.g., escalation routines for a call center agent)?

[Yes or No]

10. Does the design method provide sufficient means to prescribe rules for an internal participant of the platform operator?

[1=not at all confident ------ 7=Very confident]

11. Our design method could be made more complete by adding the ability to prescribe rules for an internal participant of the platform operator.

[1=not at all confident ------ 7=Very confident]

12. I cannot use the design method to adequately prescribe rules for an internal participant of the platform operator.

[1=not at all confident ------ 7=Very confident]

Expressiveness: Principle Overload – Case A/B

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Have you ever used an influence area for a service platform?

[Yes or No]

2. I often have to provide additional information to clarify the context in which I want to use the influence area on a service platform.

[1=not at all confident ------ 7=Very confident]

3. Influence areas can have more than one meaning.

[1=not at all confident ------ 7=Very confident]

4. I often use influence areas to represent more than one type of real-world phenomena.

[1=not at all confident ------ 7=Very confident]

5. Have you ever used [transactions and influences / interactions] on a service platform?

[Yes or No]

6. I often have to provide additional information to clarify the context in which I want to use [transactions and influences / interactions] on a service platform.

[1=not at all confident ------ 7=Very confident]

7. [Transactions and influences / Interactions] can have more than one meaning.

[1=not at all confident ------ 7=Very confident]

8. I often use [transactions and influences / interactions] to represent more than one type of real-world phenomena.

[1=not at all confident ------ 7=Very confident]

[Comment: Since no control is merged as in the other two groups, it is not possible to test for an overload of controls with this survey in a comparable manner.]

Expressiveness: Principle Excess – Case A only

Please answer each question as precisely as possible; if you are unsure, please make an estimate.

1. Have you ever used the control area for a service platform?

[Yes or No]

2. The control area does not have a real-world meaning for a service platform.

[1=not at all confident ------ 7=Very confident]

3. I often cannot precisely ascribe a real-world meaning to a control area.

[1=not at all confident ------ 7=Very confident]

4. A control area does not represent any relevant real-world phenomenon.

[1=not at all confident ------ 7=Very confident]

5. Have you ever used a transaction in a service platform design?

[Yes or No]

6. The transaction does not have a real-world meaning for a service platform.

[1=not at all confident ------ 7=Very confident]

7. I often cannot precisely ascribe a real-world meaning to a transaction.

[1=not at all confident ------ 7=Very confident]

8. A transaction does not represent any relevant real-world phenomenon.

[1=not at all confident ------ 7=Very confident]

9. Have you ever used prescriptive control on a participant?

[Yes or No]

10. Prescriptive control on a participant does not have a real-world meaning.

[1=not at all confident ------ 7=Very confident]

11. I often cannot precisely ascribe a real-world meaning to prescriptive control on a participant.

1[1=not at all confident ------ 7=Very confident]

12. Prescriptive control on a participant does not represent any relevant real-world phenomenon.

[1=not at all confident ------ 7=Very confident]

Final Questions

Rate the level of helpfulness of the diagram to understand the written text.

[1=superfluous ------ 7=would not have worked without]

Rate the level of helpfulness of the diagram to answer the questions.

[1=superfluous ------ 7=would not have worked without]

Would you have liked to see more information included in the text/ diagrams? If so, what would that be?

[Text Field]

About the Authors

Christian Janiesch is assistant professor at the University of Würzburg. He received his PhD, MSc, and a BSc in information systems from the University of Münster, Germany. Christian worked full-time at the European Research Center for Information Systems in Münster, at the SAP Research Center Brisbane at SAP Australia Pty Ltd, and at the Karlsruhe Institute of Technology before being appointed in Würzburg. Most of his research is at the intersection of business process management and business analytics but he also investigates smart services and the Industrial Internet of Things. He is on the Department Editorial Board for BISE and has co-chaired more than half a decade of ECIS tracks. He has authored over 100 scholarly publications. His work has appeared in *Business & Information Systems Engineering*, *Future Generation Computer Systems*, *Information & Management*, *Communications of the Association of Information Systems* as well as various major international conferences including ICIS, ECIS, BPM, and HICSS and has been registered as US patents.

Christoph Rosenkranz is a professor of integrated information systems in the Cologne Center for Information Systems of the Faculty of Economics, Management and Social Sciences at the University of Cologne, Germany. Christoph holds a doctoral degree (Dr. rer. pol.) from Goethe University and a *Diplom* degree (Dipl. Wirt.-Inform.) from the University of Münster. His research focuses on information systems development and IT project management, digital transformation and business process management, and organizational effects of digital transformation. He has presented his work at international conferences such as ECIS and ICIS and has published articles in publications including *European Journal of Information Systems*, *Information Systems Journal*, *Journal of Information Technology*, and *Journal of the Association for Information Systems*.

Ulrich Scholten is an internationally active cloud computing scientist, entrepreneur, and angel investor. He received his PhD in information technology from the Karlsruhe Institute of Technology working on viral growth effects around internet services and platforms. While at the institute, he led a project on Strategic Value Nets with SAP Research, researching network effects, emergence, and control mechanisms in platforms and distributed cloud scenarios. Ulrich invented the dynamic network model and notation to model and investigate service platforms for their ability to create or control network effects. He has extensive experience in creating, capitalizing, and evolving start-ups, including successful exits, and he holds several patents on intelligent sensors in the US and the EU. He is a founding member of SkyRadar.com, and he set up VentureSkies, which offers consultation on viral growth concepts for cloud platforms. Ulrich's current focus is on the impact of artificial intelligence and the internet of things on web-based marketing and growth strategies, taking into consideration technology's role, market offer and demand, and the legal framework in the concerned areas of jurisdiction.