IT Self-Service Blueprinting

A Visual Notation for Designing IT Self-Services

Marvin Schönwälder, Tom Szilagyi, Florian Bär, Birger Lantow, and Kurt Sandkuhl

University of Rostock, Albert-Einstein-Str. 22, 18059 Rostock, Germany
{marvin.schoenwaelder, tom.szilagyi, florian.baer, birger.lantow, kurt.sandkuhl}@uni-rostock.de

Abstract. IT self-services have gained momentum in more and more organizations in recent years. The rationale for IT self-services is the reduction of the IT personnel's workload in IT service production. However, prior research has shown that the realization of this anticipated benefit is not self-evident. Therefore, scholars have called for research leading to artifacts that guide IT operations at devolving service tasks to employees. In this paper, we present a visual notation for designing IT self-services. The meta-model of this visual notation includes the concepts and relationships in service blueprinting and extends these with concepts and relationships for IT self-service blueprinting. IT self-service blueprinting supports IT operations at analyzing the devolvement of service tasks at the design stage. The demonstration of the use of our visual notation proofs that the visual notation complies with the principles for cognitively effective visual notations.

Keywords: Service blueprinting, Self-service, Service operations, IT operations, Visual notation.

1 Introduction

A service can be viewed as a type of process representing a sequence of service tasks that allow the production of the service (i.e., the outcome of the service process) itself [1, 2]. In recent years, information technology (IT) has changed the way services are produced in organizations [3, 4]. A service is impacted by IT in two major ways: high convenience and high automation [5]. IT services are independent from geographical locations, i.e. service tasks can be performed via IT, such as personal computers, mobile phones, and tablets, across geographical distances. In addition, because such IT has become ubiquitous to people, service tasks can be performed at any time. Nevertheless, in IT services, only a few service tasks require manual input. Most of the service tasks, in IT services, are automated, i.e. performed by software or another IT.

In IT services, the IT operations (i.e., operation of an organization's IT infrastructure) function of an organization takes the role of the service provider. The IT-related outcomes (e.g., software, virtual machines, and containers) of IT services are produced for the employees[1] outside of IT operations. IT services must be considered as a continuum bounded by two extremes: IT full-services/autonomous services and IT self-services [6, 7]. In IT self-services, a portion of the service tasks, which otherwise would have been performed by the IT personnel, is performed by the employees on their own and independently. The more service tasks are performed by the employees, the higher is the self-service and co-production intensity [8, 9].

In recent years, a vast number of organizations has substituted phone and e-mail contact with IT self-services to reduce the workload of the IT personnel in IT service production [10, 11]. However, often organizations and IT operations fail at reducing the workload of their personnel in (IT) self-services, as it has been reported by prior research [12, 13]. Therefore, scholars call for more detailed research on how to guide managerial decisions on which service tasks to automate and to devolve to specific customers or employees [14, 15]. This research represents a first step towards filling this research gap, because it addresses the following research question: What is a cognitively effective visual notation for designing IT self-services from an IT operations perspective?

In this paper, we present a visual notation for IT self-service design. IT operations can use this visual notation to design IT self-services in which the IT personnel's workload is reduced compared to service processes in which all the service tasks are performed by the IT personnel. Based on an IT self-service blueprint, IT operations can analyze potential fail points and their impacts on the IT personnel's workload in an IT self-service. In addition, solutions to these fail points can be identified and specified in the IT self-service blueprint. IT self-service blueprinting enables IT operations to conduct these analyses before the cost-intensive implementations of IT self-services. The visual notation can be part of a method guiding IT operations at devolving IT services to employees.

2 Theoretical Foundations

2.1 IT Self-Services

From an IT operations perspective, the rationale for IT self-services is freeing the IT personnel from performing routine, recurrent service tasks. By devolving the routine and recurrent service tasks to the employees, IT operations aims for reducing the IT personnel's workload in IT self-services.

However, the research results of a multiple-case study demonstrate that the realization of this anticipated benefit is not self-evident [14]. The devolvement of service tasks to employees goes hand in hand with a transfer of control from IT operations to the employees [16]. Therefore, in self-services, the service providers often face a lack of service production control. The lack of service production control takes different forms, including ambiguous information [13], intentional misperformance [12], and general

[1] For simplification, in the following text we refer to the employees of other organizational functions for which IT services are produced as "employees".

self-service failures [17], depending on the design of the self-service. In IT self-services, we identified the lack of service production control to take the forms of a forbidden (i.e., IT self-service outcome is produced for a purpose that is not compliant with license terms, legal regulations, and corporate policies and guidelines) and an excessive service production (i.e., IT self-service outcome is produced excessively, and the employees are wasteful with the IT resources).

The lack of service production control in IT self-services is rooted in capability gaps and free IT self-service outcomes. Capability gaps will arise, if the employees do not possess the capabilities required to perform the devolved service tasks correctly. A free IT self-service outcome is an IT self-service outcome that can be ordered by the employees in the IT self-service, but whose production cost is not charged to these employees. A free IT self-service outcome will cause the lack of service production control in IT self-services, if it relies on IT resources (e.g., central processing unit (CPU), memory, and storage) that are limited in their amount to IT operations and its increase comes with cost. Based on the IT self-service outcome, IT self-services can be classified into [18]: information seeking (i.e., information is exchanged unilaterally), communication and interaction (i.e., information is exchanged bilaterally), and purchase and transactions (i.e., IT resources are exchanged).

To establish a sufficient level of service production control in IT self-services, IT operations must adopt one or more solutions comprising a set of behavioral patterns. We identified five behavioral patterns, whose adoptions in various combinations form solutions to the lack of service production control in IT self-services: chargeback and limitation (i.e., cost of producing IT services is allocated to the requesting employees or organizational functions), standardization of the IT self-service (i.e., reduction of the employees' options to customize the IT self-service outcome), authorization of service requests (i.e., review of the service request to approve or reject service production), showback (i.e., employees are informed, but not charged, about the cost of producing IT services), and training and support (i.e., employees are supported at performing the service tasks correctly and are provided with the required capabilities).

2.2 Service Blueprinting

Service operations research has introduced several visual notations for designing services [19]. However, service blueprints that are created by using these visual notations for IT self-service design do not depict whether the root causes of the lack of service production control do exist in designed IT self-services [14]. These visual notations do not support the specification of IT self-services. Furthermore, the following concepts are missing:

- IT self-service classification
- Employee capabilities required for the service tasks
- IT self-service specific types of fail points
- Behavioral patterns as solutions to these to fail points
- Specific IT-Service related resources

Therefore, a new visual notation for designing IT self-services must be developed, whose underlying concepts and their relationships extend the meta-model [20, 21] of a well-known and well-accepted visual notation for service design [14].

Service blueprinting [22, 23] is widely adopted by scholars and practitioners [2, 24]. Many of the visual notations for service design extend service blueprinting by additional concepts and relationships [19]. Although, there is no standard visual notation for service blueprinting, the visual notations that draw from service blueprinting are based on identical concepts and relationships between these [25, 26].

Service blueprinting supports the specification of action flows representing the sequences of actions by different categories of actors in a service process. Actor categories are customer, onstage personnel (i.e., service personnel that has face-to-face contact with the customer), backstage personnel/systems (i.e., service personnel or system that performs actions invisible to the customer), support personnel/systems (i.e., service personnel or system to which the customer has no contact, but that performs actions required for the service to be produced), and management (i.e., service managers that are responsible for planning, managing, and controlling). In a service blueprint, the actor categories are separated by four horizontal lines: line of interaction (i.e., separation between customer actions and onstage personnel actions), line of visibility (i.e., separation between onstage personnel actions and backstage personnel/systems actions), line of internal interaction (i.e., separation between backstage personnel/systems actions and support personnel/systems actions), and line of implementation (i.e., separation between support personnel/systems actions and management actions). A communication flow, which represents the flow of communication between the actor categories, can be specified using service blueprinting.

Props and physical evidences that are all the tangibles with which the customer interacts during the service process must be specified in a distinct area, which is separated from the customer actions by another horizontal line.

For actions that can fail when performed, service blueprinting supports the assignment of fail points. Numerous fail points can be assigned to an action. For each fail point, a subprocess must be specified that solves this potential failure.

Service blueprinting is not only for designing services, but also for analyzing services. For each action, the standard execution time and maximum execution time that is tolerated by the customers before lowering their assessments of service quality must be defined. Based on this information, the request fulfillment manager can determine the standard execution time of the service process and how much that time will increase, if fail-safe subprocesses must be performed.

2.3 Notation Quality

In order to evaluate the presented approach, notation quality criteria are applied. Major work on the quality of visual notations is provided by Moody with his article “Physics of Notations” [27]. However, Moody mainly focuses on the comprehensibility of notations. Furthermore, he uses notation and language synonymously. There is more about a language than its notation. A distinction can be made between the symbols and the concepts behind them together with the abstract syntax for these concepts [28]. A more comprehensive approach to language quality that integrates Moody's work is the SEmiotic QUALity framework (SEQUAL) by Krogstie [29]. Though originally defined for the assessment of conceptual models, it can also be used for modeling languages. In SEQUAL, the used modeling language is related to all quality criteria of a model. Thus, the criteria can also be applied to the language. Krogstie defines the following quality criteria for modeling languages [29]: domain appropriateness (i.e., concepts of the modeling language should be able to express anything in the domain of interest), comprehensibility appropriateness (i.e., actors in the modeling process should be able to interpret the model), participant appropriateness (i.e., concepts of the modeling language should match the concepts that the participants in the modeling process use to perceive the domain of interest), modeler appropriateness (i.e., modeler should be able to express his or her domain knowledge using the language), tool appropriateness (i.e., modeling language should be machine interpretable), and organizational appropriateness (i.e., modeling language should be aligned with organizational goals such as standardization and technology roadmaps).

This study focusses on the second quality criterium "comprehensibility appropriateness" that is well addressed by the nine principles for visual notations proposed by Moody [27]: semiotic clarity (i.e., there should be a 1:1 mapping between concepts and graphical symbols), perceptual discriminability (i.e., graphical symbols should be easily and accurately distinguishable from each other), semantic transparency (i.e., graphical symbols should intuitively reflect their semantics), complexity management (i.e., there should be constructs for different levels of abstraction and information filtering), cognitive integration (i.e., visual notation should provide explicit mechanisms to support navigation between different diagrams), visual expressiveness (i.e., visual notation should use the full range of visual variables such as size, shape, and color), dual coding (i.e., textual description should complement graphical symbols), graphic economy (i.e., number of different graphical symbols should not be too large), and cognitive fit (i.e., visual notation should be adapted to the audience).

Though mainly addressing "comprehensibility appropriateness", some of Moody's principles can be mapped to the other quality criteria of SEQUAL. The principle "semiotic clarity" can be mapped to "domain appropriateness" when it comes to missing graphical symbols (i.e., symbol deficit) or graphical symbols that are not linked to the domain (i.e., symbol excess). Comprehensibility is affected when multiple graphical symbols represent the same concept (i.e., symbol redundancy) or one graphical symbol represents several concepts (i.e., symbol overload). The principles "semantic transparency" and "cognitive fit" depend on the specific knowledge of the model users and thus address the quality criteria of "participant appropriateness" and "modeler appropriateness".

The principle "graphic economy" requires a definition of what is a large number of different graphical symbols. Moody defines this number to be six [27]. However, most languages for modelling processes have a lot more than six different graphical symbols. A research stream that allows an assessment of this number is language or method complexity analysis [30, 31]. Several complexity metrics have been defined to evaluate the concepts, relationships, and attributes of a language's meta-model. Based on these metrics, modeling languages can be compared.

3 Research Approach

As a result of the discussions in section 2, we developed a new visual notation for designing IT self-services, whose meta-model includes not only the concepts and relationships in service blueprinting (see section 2.2), but also the concepts and relationships that are relevant to the design of IT self-services (see section 2.1).[2]

The visual notation of IT self-service blueprinting was developed taking into account Moody's nine principles for cognitively effective visual notations (see section 2.3).[3] We demonstrated the use of the visual notation in one case (i.e., IT self-service), which was analyzed as part of a multiple-case study, to proof its feasibility. In section 4, the documentation of the outcome of this demonstration is presented. Furthermore, in section 4, we describe how the visual notation of IT self-service blueprinting works (i.e., graphical symbols used to represent the underlying concepts).

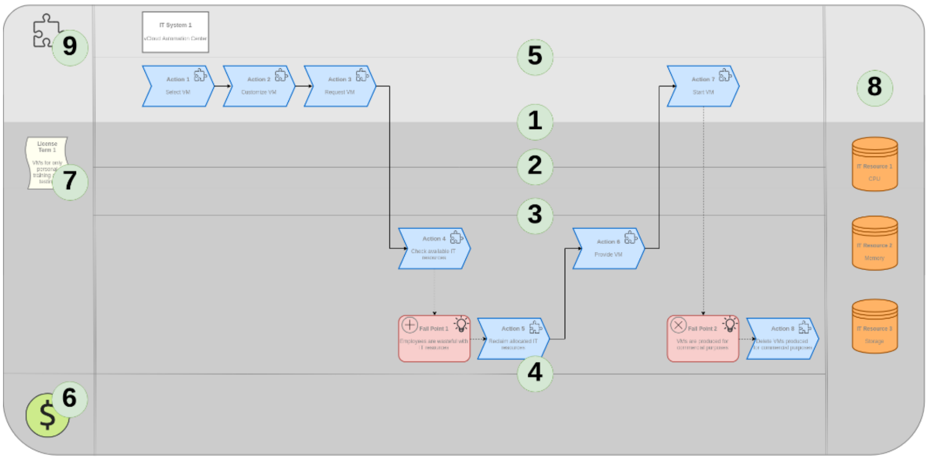

Fig. 1. IT self-service blueprint resulted from the demonstration of the visual notation.

4 Research Results

The documentation (i.e., IT self-service blueprint) of the outcome of our demonstration is depicted by Fig. 1. The different areas and graphical symbols of the IT self-service blueprint are described in the following.

---

[2] The meta-model of IT self-service blueprinting can be obtained from GitHub: https://github.com/Floble/IT-Self-Service-Blueprinting/blob/master/VPProjects/Self-Service%20Meta-Model.jpg. The white colored concepts represent the concepts in service blueprinting. The blue colored concepts represent the concepts that are relevant to the design of IT self-services (see section 2.1).

[3] The visual notation of IT self-service blueprinting was implemented as a custom library for the cloud-based modeling tool draw.io. This custom library can be obtained from GitHub: https://github.com/Floble/IT-Self-Service-Blueprinting/tree/master/VisualNotation.

The IT self-service blueprint is separated into two main areas by the line of interaction (see (1) in Fig. 1): employee area (light gray) and IT operations area (dark gray). While the service tasks that are performed by the employees are above the line of interaction, the service tasks that are performed by IT operations are below the line of interaction. The IT operations area includes the service tasks performed by IT onstage personnel, IT backstage personnel/systems, IT support personnel/systems, and IT management (see section 2.2). In the IT self-service blueprint, the IT personnel is separated by the line of visibility (see (2) in Fig. 1), line of internal interaction (see (3) in Fig. 1), and line of implementation (see (4) in Fig. 1). IT systems with which the employees interact during the IT self-service are in the employee area but are separated from the employee actions by a horizontal line (see (5) in Fig. 1).

In the lower left corner of the IT self-service blueprint, the class of the IT self-service is specified (see (6) in Fig. 1). As depicted by Fig. 2, each IT self-service class (see section 2.1) is represented by an icon.

The left side of the IT self-service blueprint, which is separated by a vertical line, shows the license terms that restrict the employees' usage of the IT self-service outcome (see (7) in Fig. 1 and Fig. 3). If the employees are aware (unaware) of the usage restrictions, the license terms must be in the employee (IT operations) area.

Fig. 2. Icons for classifying the IT self-service.

The IT resources on which the IT self-service outcome relies are in the right side of the IT self-service blueprint that is separated by a vertical line (see (8) in Fig. 1 and Fig. 3). IT resources that are provided and managed by IT operations are in the IT operations area. In contrast, the IT resources that are in the employee area are provided and managed by the employees or organizational functions.

Fig. 3. Graphical symbols for physical evidences, IT systems, IT resources, and license terms.

A click on the icon that is in the upper left corner of the IT self-service blueprint (see (9) in Fig. 1) opens a sub-model depicting the capabilities possessed by the employees. If employees, which possess similar capabilities but belong to different organizational functions, are involved in the IT self-service, the employees must be grouped, and the possessed capabilities are depicted by only one sub-model (see Fig. 4 a)). Actions are also linked to capability sub-models (same symbol as for the service but within the respective action symbol, see Fig. 1). The action's sub-model depicts the capabilities required to perform this action correctly (see Fig. 4 b)). The level, to which a specific capability is required, is specified in the upper right corner of the capability's graphical symbol (see Fig. 4). It ranges from 0.25 to 1.0 in increments of 0.25. The three icons at the bottom of the sub-model in Fig. 4 allow to navigate between sub-models and return to the main model depicting the action flow (see Fig. 1).

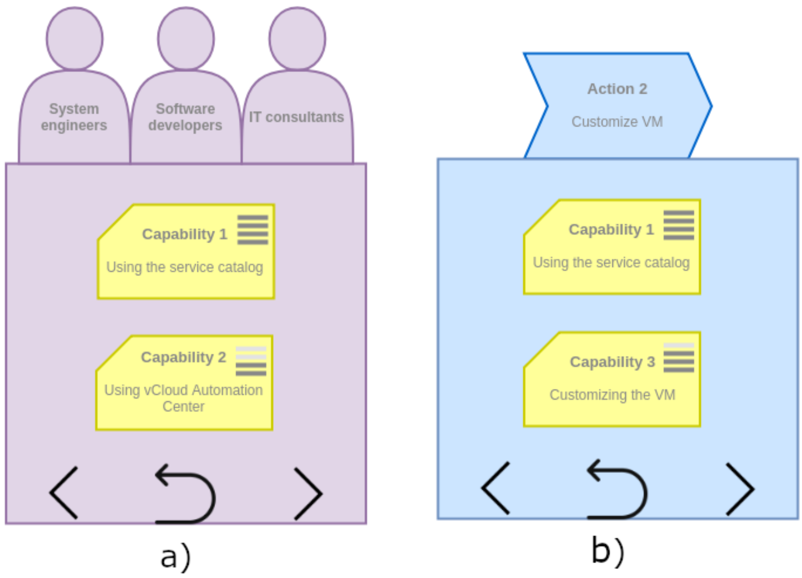

Fig. 4. Sub-model depicting the capabilities a) for employee groups and b) for actions.

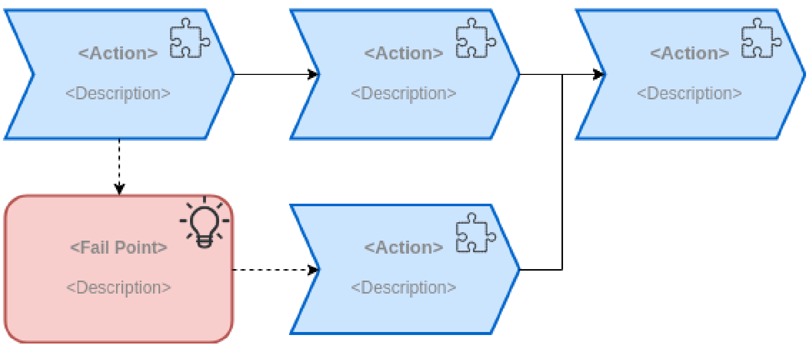

Fig. 5. Action flow including one fail point.

Actions are connected with each other by arrows indicating the flow of action (see Fig. 5). Fail points are assigned to actions by dotted arrows (see Fig. 5). A click on the icon that is in the upper right corner of a fail point's graphical symbol opens the fail point's sub-model depicting the solutions to this fail point (see Fig. 8). Each fail point is classified by an icon that is in the upper left corner of the fail point's graphical symbol (see Fig. 8). Fig. 6 depicts the icons for classifying the fail points (see section 2.1).

Fig. 6. Icons for classifying fail points.

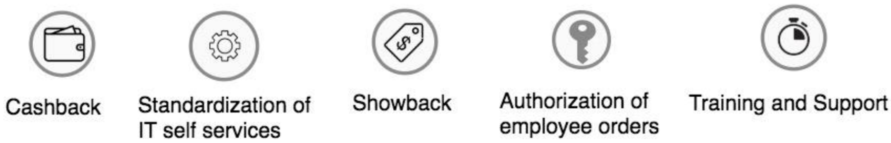

Fig. 7. Icons for classifying behavioral patterns.

A solution to a fail point comprises the adoption of one or more behavioral patterns (see section 2.1). The behavioral patterns are depicted by the sub-model of the fail point (see Fig. 8).

The icon in the upper left corner of a behavioral pattern's graphical symbol classifies the behavioral pattern (see Fig. 8). As depicted by Fig. 7, there are five different classes of behavioral patterns (see section 2.1).

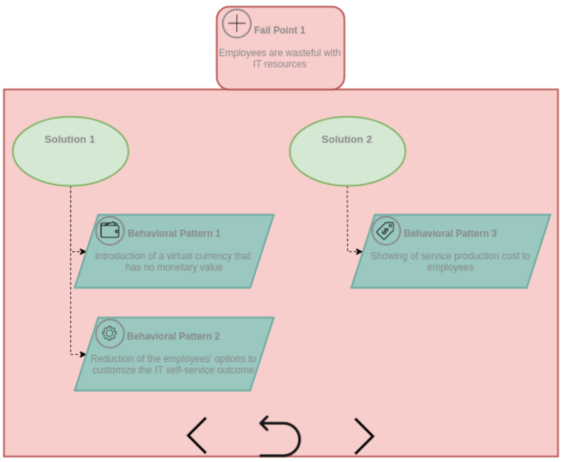

Fig. 8. Sub-model depicting the solutions to a fail point.

5 Discussion

In this paper, we present a visual notation for designing IT self-services. This visual notation supports IT operations at analyzing the devolvement of service tasks at the design stage. It thereby contributes to the filling of an important research gap in information systems and service operations research [14, 15]. The meta-model of the developed visual notation includes the concepts and relationships in service blueprinting and extends these with concepts and relationships for IT self-service blueprinting.

The presented visual notation for IT self-service design complies with the principles for cognitively effective visual notations suggested by Moody [27] (see Table 1).

Table 1. Compliance with the principles for cognitively effective visual notations [27].

Principles	**Compliance**
Semiotic clarity	Each concept included in the meta-model is represented by a graphical symbol.
Perceptual discriminability	Graphical symbols differ in shape and color. Fail point and behavioral pattern classes differ in icons.
Semantic transparency	Guidelines for semantic transparency are applied by specifying icons for fail points and behavioral patterns. Furthermore, action flow, sub-models, and solutions use sequence, spatial enclosure, and hierarchy to represent relationships. Semantic transparency has always an audience related component. This could be further investigated in the future.
Complexity Management	Sub-models allow the representation of the IT self-service at multiple levels of abstraction.
Cognitive integration	Action flow serves as the context diagram. At the top of each sub-model, the related graphical symbol instance is shown as contextual information. At the bottom of each sub-model, icons allow to navigate between sub-models and return to the main model.
Visual expressiveness	Horizontal position, shape, and color are the visual variables used in the visual notation. The visual notation uses a wide range of values of these visual variables.
Dual coding	Descriptions to graphical symbol instances convey additional information.

Graphic economy	Must be evaluated in the future by applying complexity metrics.
Cognitive fit	The need for different visual dialects for different tasks and/or audiences must be evaluated in the future.

So far, the use of the developed visual notation has been demonstrated in one case. Although such a demonstration can be considered as a preliminary evaluation, a true evaluation of the visual notations must be conducted in the future.

References

1. Leyer, M.: A framework for systemising services from an operations management perspective. Int. J. Serv. Oper. Manag. 23, 486–509 (2016).
2. Sampson, S.E.: Visualizing Service Operations. J. Serv. Res. 15, 182–198 (2012).
3. Huang, M.-H., Rust, R.T.: IT-Related Service: A Multidisciplinary Perspective. J. Serv. Res. 16, 251–258 (2013).
4. Huang, M.-H., Rust, R.T.: Artificial Intelligence in Service. J. Serv. Res. 21, 155–172 (2018).
5. Harvey, J., Lefebvre, L.A., Lefebvre, E.: Flexibility and technology in services: a conceptual model. Int. J. Oper. Prod. Manag. 17, 29–45 (1997).
6. Campbell, C.S., Maglio, P.P., Davis, M.M.: From self-service to super-service: a resource mapping framework for co-creating value by shifting the boundary between provider and customer. Inf. Syst. E-Bus. Manag. 9, 173–191 (2010).
7. Globerson, S., Maggard, M.J.: A Conceptual Model of Self-service. Int. J. Oper. Prod. Manag. 11, 33–43 (1991).
8. Baer, F., Leyer, M.: Towards assessing the value of digital self-service options from a provider perspective. AMCIS 2016 Proc. (2016).
9. Haumann, T., Güntürkün, P., Schons, L.M., Wieseke, J.: Engaging Customers in Coproduction Processes: How Value-Enhancing and Intensity-Reducing Communication Strategies Mitigate the Negative Effects of Coproduction Intensity. J. Mark. 79, 17–33 (2015).
10. Zaza, I.: No One Will Help You if You Won't Help Yourself. AMCIS 2016 Proc. (2016).
11. Zaza, I., Junglas, I.: IT Self-Service Engagement: A Theory of Trying Perspective. ICIS 2016 Proc. (2016).
12. Ellway, B.: Design vs practice: How problematic call routing and rerouting restructures the call centre service system. Int. J. Oper. Prod. Manag. 36, 408–428 (2016).
13. Kumar, A., Telang, R.: Does the Web Reduce Customer Service Cost? Empirical Evidence from a Call Center. Inf. Syst. Res. 23, 721–737 (2012).
14. Bär, F.: Tackling Knowledge Gaps in Digital Service Delivery. CEUR Workshop Proc. 1821, 55–60 (2017).

15. Scherer, A., Wünderlich, N.V., von Wangenheim, F.: The Value of Self-Service: Long-Term Effects of Technology-Based Self-Service Usage on Customer Retention. MIS Q. 39, 177–200 (2015).
16. Rowley, J.: An analysis of the e-service literature: towards a research agenda. Internet Res. 16, 339–359 (2006).
17. Hilton, T., Hughes, T., Little, E., Marandi, E.: Adopting self-service technology to do more with less. J. Serv. Mark. 27, 3–12 (2013).
18. Harrison, T., Waite, K.: Impact of co-production on consumer perception of empowerment. Serv. Ind. J. 35, 502–520 (2015).
19. Bär, F., Sandkuhl, K., Schmidt, R.: On the Maturity of Service Process Modeling and Analysis Approaches. In: Abramowicz, W., Alt, R., and Franczyk, B. (eds.) Business Information Systems. pp. 356–367. Springer International Publishing (2016).
20. Harel, D., Rumpe, B.: Meaningful modeling: what's the semantics of "semantics"? Computer. 37, 64–72 (2004).
21. Sprinkle, J., Rumpe, B., Vangheluwe, H., Karsai, G.: Metamodelling: State of the Art and Research Challenges. ArXiv14092359 Cs. (2014).
22. Shostack, G.L.: How to Design a Service. Eur. J. Mark. 16, 49–63 (1982).
23. Bitner, M.J., Ostrom, A.L., Morgan, F.N.: Service Blueprinting: A Practical Technique for Service Innovation. Calif. Manage. Rev. 50, 66–94 (2008).
24. Becker, J., Beverungen, D., Knackstedt, R., Matzner, M., Muller, O., Poppelbuss, J.: Bridging the Gap Between Manufacturing and Service Through IT-Based Boundary Objects. IEEE Trans. Eng. Manag. 60, 468–482 (2013).
25. Kazemzadeh, Y., Milton, S.K., Johnson, L.W.: A Comparison of Concepts in Service Blueprinting and Process Chain Network (PCN). Int. J. Bus. Manag. 10, p13 (2015).
26. Milton, S.K., Johnson, L.W.: Service blueprinting and BPMN: a comparison. Manag. Serv. Qual. Int. J. 22, 606–621 (2012).
27. Moody, D.L.: The "Physics" of Notations: Towards a Scientific Basis for Constructing Visual Notations in Software Engineering. IEEE Trans. Softw. Eng. 35, 23 (2009).
28. Karagiannis, D., Kühn, H.: Metamodelling Platforms. In: E-Commerce and Web Technologies. pp. 182–182. Springer, Berlin, Heidelberg (2002).
29. Krogstie, J.: Model-Based Development and Evolution of Information Systems: A Quality Approach. Springer-Verlag, London (2012).
30. Rossi, M., Brinkkemper, S.: COMPLEXITY METRICS FOR SYSTEMS DEVELOPMENT METHODS AND TECHNIQUES. 19.
31. Hasic, F., Smedt, J.D., Vanthienen, J.: Towards Assessing the Theoretical Complexity of the Decision Model and Notation (DMN). 8.