A Software Ecosystem for the Development of Digital Service Design Tools: A Conceptual Framework

Timon Sengewald(✉), Max Jalowski, and Martin Schymanietz

Chair of Information Systems – Innovation and Value Creation, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Nuremberg, Germany
{timon.sengewald,max.jalowski,martin.schymanietz}@fau.de

Abstract. Numerous service design tools, techniques, and methods have been developed in science and practice alike (e.g., *Persona*, *Service Blueprint*, *Stakeholder Map*). Some of these tools build on each other, e.g., different developed *Personas* can be positioned and placed in a broader overall context within a *Stakeholder Map*. Digitisation of the design process can generate significant benefits; for instance, results can be transferred between different digital tools and devices using a common or standardised file format. However, there are few digital tools to support the service design process and most of them are not interoperable. This prevents the design of services using digital technologies from achieving their full benefits. To address this problem, an ecosystem is needed that can foster the development of digital-enabled service design tools. The paper follows a design science approach to create a framework involving five design objectives as necessary steps towards such a software ecosystem.

Keywords: Service design · Digital-enabled service design · Service design tools · Service innovation · Design science research

1 Introduction

For some time now, it has been apparent that the economy is moving from product-based to service-based [1, 2], and Vargo and Lusch's Service-Dominant Logic [3, 4] has influenced business model development from a product-based to a service-based perspective. For researchers and companies alike, the issues of service innovation and service business model innovation have become increasingly central to new forms of competitiveness [5, 6], with calls for a more systematic and structured approach to innovation [7] and service innovation [8]. Service Science addresses this need by advancing the systematic development of services and service systems [2, 9] to make results more predictable [7] and to support creativity during the innovation process [10, 11]. Service design plays an important role in facilitating customer-centred and systematic innovation [12], including a systems perspective that takes account of the extended networks of actors and digital technologies now involved in service provision [13, 14]. This systematic approach is the focus of service systems engineering research [14], and the methods,

F. Ahlemann et al. (Eds.): WI 2021, LNISO 46, pp. 203–218, 2021.
https://doi.org/10.1007/978-3-030-86790-4_15

techniques and tools of service innovation, service design and service systems engineering are extensively reported both in the scientific literature (e.g., *Service Blueprint* [15]; Service *Experience Blueprint* [16]; *Information Service Blueprint* [17]; *TRIGGER* [18]) and in practitioner publications (e.g., *Business Model Generation* [19]; *The digital transformation playbook* [20]; *This is Service Design Doing* [21]; *Sprint* [22]). As well as tools for designing single services or service systems, there are tools for designing the associated business models (e.g., *Service Business Model Canvas* [23]; *SDBM/R* [24]) and for modelling how a company creates, appropriates and captures value by providing a service [25]. As Gilsing et al. [25] have shown, methods can also be developed for evaluating the design of business models.

A systematic approach to innovation is increasingly necessary for a number of reasons, including increased probability of success [7] and mastery of complex systems [8, 14]. This systematic development process can be partly or wholly supported by methods or tools [26, 27]. In many cases, the steps of the process are worked through interactively [28, 29], often involving collaboration with potential customers [30] and other relevant actors [31, 32]. In developing solutions systematically, the design process is also likely to be iterative [28, 29]. Relevant terms, methods, techniques and tools will be more precisely defined in Sect. 2.

Many of the methods, techniques and tools developed by the scientific community to support the service design process have been combined in new ways to create coherent methods (e.g., [18]). For instance, as Li and Peters [33] have shown, the formal structuring and analysis of service systems can also produce service innovations. In relation to the use of digital technologies to support the innovation process, earlier research confirms that IT (especially software) can support user creativity [34, 35] as well as systematic and structured development [10]. For example, because digital objects can be easily duplicated, digital tools make it much easier to create multiple scenarios or variants of a basic design, and the ability to undo or redo actions makes it easier to experiment [10]. Researchers have also investigated the use of digital technologies for collaborative design of business models (e.g., [36–38]).

The advantages of digital solutions for collaborative work extend to support for remote corporate teamworking in crisis situations such as the current COVID-19 pandemic [39]. However, although innovative tools or IT artefacts can make a valuable contribution to the development of service-oriented business models or transformation of existing business models [40], only a few such tools have been developed within service science research and practice. To explore how the development of such tools might be promoted, this paper addresses the following research question:

> **RQ:** What technical standards and prerequisites need to be considered to support the creation of a software ecosystem for the development of digital-enabled service design tools?

We argue that this research is highly relevant as it can help increase the transfer of complex IT artefacts for service design to practice and vice versa. An ecosystem, which interlocks science and practice more closely, can help to apply the quality criteria of research in the development of digital-enabled service design tools in practice and can

thus ensure more rigor. The artefacts can then in turn be tested in practice in real-world scenarios and thus transfer knowledge back to science.

The paper is structured as follows. The next section provides an overview of the current state of research on software ecosystems and defines the concept of digital-enabled service design. We go on to describe the artefact by presenting a conceptual framework for a software ecosystem for developing digital-enabled service design. We then discuss how the field of service innovation can benefit from the creation of such a software ecosystem to foster the development of digital-enabled tools and methods for service design research and practice. Finally, we discuss the study's contribution to Information Systems and Service Science, as well as limitations and directions for future research.

2 Theoretical Background

2.1 Digital-Enabled Service Design

In distinguishing between digital and non-digital services [29], the former can be defined as *'a service, which are obtained and/or arranged through a digital transaction (information, software modules, or consumer goods) over Internet Protocol (IP)'* [29, p. 506]. The distinction from a normal service is that the specification of the supply as digital is more restrictive by comparison [29]. The term *digital innovation* has two possible meanings, referring either to digital technologies that support the innovation process itself or to the output of that process [27]. In research contexts, the term *digital service design* commonly refers to the design of digital services as an object of research (e.g., [28]). Equally, however, the term may imply that the design process itself is digital-based. To avoid misinterpretation or ambiguity, the term *digital-enabled service design* is used here to refer to the design of services using digital technologies.

According to Brinkkemper, the terms *method*, *technique* and *tool* are used in the present context [41], but these are not clearly defined in the literature and are used interchangeably [28]. Brinkkemper defines a *method* as *'an approach to perform a systems development project, based on a specific way of thinking, consisting of directions and rules, structured in a systematic way in development activities with corresponding development products'* [41, p. 275 f.] (see also [18, 42]). On the other hand, a *technique* is '*a procedure, possibly with a prescribed notation, to perform a development activity*' [41]. (For an example of a technique, see [43]). Finally, a *tool* is '*a possibly automated means to support a part of a development process*' or to support the entire development cycle [41] (e.g., *Information Service Blueprint* [17])—for instance, a software application, a design template, or a hardware device to support the design process [28]. A tool usually supports one part of a technique [44], although complete mapping of a technique or method is also possible [41]. Based on these distinctions, we understand the term *digital-enabled service design tool* to mean a digital tool that supports the design of services or service systems by representing a design method or technique in full or in

part. In this definition, the use of a template in a digital whiteboard tool, as they are common with miro[1], mural[2], or strategyzer[3], would also be understood as a digital-enabled service design tool. This type of tool provides users with a digital whiteboard to which digital post-its can be attached. By providing drawing tools, almost every service design tool that can be displayed on a canvas for a workshop can also be displayed digitally. However, this paper focuses on more complex digital-enabled service design tools - such as ExperienceFellow[4] or smaply[5]. Smaply, for example, is a web service that enables its users to create virtual personas and also offers ready-made avatars, quotes and visualizations [45]. Tools that provide users with ready-made content for exploration can increase users' creativity by providing inspiration [46]. The app ExperienceFellow enables the collection of customer data. Users can capture different types of data (text, video or images) to document emotional experiences at customer touchpoints. These two tools therefore have a very high degree of specificity. These two examples enable two different activities, while the second enables the collection of data, the former directly supports the design process.

2.2 Development of An Software Ecosystem for Digital-Enabled Service Design

A software ecosystem is a system based on a software solution whose functionalities can be extended, for example, by third-party plugins [47]. Software ecosystems have attracted increasing research interest [47–49] and are increasingly used as industry business models [48, 50]. In general, software ecosystems can be defined in business and technical terms [51] or from a social perspective [48]. Scientific definitions differ [e.g., 51–53], but from a business perspective, a software ecosystem can be characterised as *'the set of software solutions that enable, support and automate the activities and transactions by the actors in the associated social or business ecosystem and the organisations that provide these solutions'* [50, p. 2]. Typically, such ecosystems encompass the necessary technology for implementation, the overall project infrastructure (e.g., project repositories, community platform) and the development methodology (e.g., standards, documentation) [52]. The ecosystem is controlled by a central actor or hub, usually the provider of the core software solution [47, 53]. Software ecosystems facilitate the construction of extensive software systems on a central platform, bringing together the components created by internal and external actors [48]. These individuals or organisations have different incentives for participating in the system [54], including increased benefits for existing users, increased attraction of potential new customers, potential for open innovation and reduced total cost of ownership [50]. In defining a set of rules for communication and cooperation among these actors, it is important to adopt a social perspective [48].

Various theories and strategies already exist for the systematic development of software ecosystems [e.g., 47, 50, 52]. Research to date suggests that standardisation can

[1] For a more detailed description follow the link: https://miro.com/.
[2] For a more detailed description follow the link: https://www.mural.co.
[3] For a more detailed description follow the link: https://www.strategyzer.com.
[4] For a more detailed description follow the link: https://www.experiencefellow.com.
[5] For a more detailed description follow the link: https://www.smaply.com.

increase innovation capacity [55]—for example, a standardised software architecture or data format [50]. Software architecture can be defined as a *'structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them'* [48, p. 1295]. The software architecture should be designed to be easy for developers to understand, and it should be well documented [52]. The openness of the system and its core components is also crucial [56]—that is, open standards, open formats and open source [47]. Open standards support the development of interchangeable and interoperable components by different actors; open formats are a specific form of open standard that allow the exchange of data, information and knowledge [47]. Open source is the highest form of openness, as enabling access to the source code generates knowledge, modification and re-provision [47]. The ecosystem community and the associated form of organisation is another key factor [48, 52], and within this community or ecosystem, rules for communication and legalities must be clearly defined [52]. A well-formed and integrated community can increase a software ecosystem's robustness [47].

3 Research Design

To address the research question, we adopt a design science research (DSR) approach following Hevner et al. [57]. Within the DSR paradigm, a designer or researcher attempts to solve a problem by designing an innovative artefact [58]. In doing so, this work aims to develop an artefact which can be applied to the real-world and solve the problem [58, 59] of insufficient interoperability of digital-enabled service design tools. The artefact designed to solve the presented problem is a construct, where a construct is understood as a concept or conceptualisation [60, 61]. For artefact development, the six steps of the Design Science Research Methodology (DSRM) by Peffers et al. [62] are applied. These steps are as follows: (1) Problem Identification, (2) Objective Definition, (3) Design & Development, (4) Demonstration, (5) Evaluation and (6) Communication [62]. The methodology follows the Build and Evaluate pattern of Sonnenberg and vom Brocke [63]. This means that we already start the evaluation of the problem and objectives (ex ante evaluation) before we start with the specific design of the artefact. Following step 1, the problem underlying the artefact was treated and explained in the introduction. The design and development of the artefact adopts a conceptual approach, which explores new ideas and new connections between existing theories [64]. As in the case of empirical research methods, there are divergent approaches to conceptual research [65, 66]. In general, these can be divided into theory synthesis, theory adaptation, typology and modelling [65]. To develop the theoretical framework outlined below, we employed a conceptual modelling approach, in which the focal concept [65] is digital-enabled service design and the systematic development of ecosystems. In particular, we explored the factors that promote the formation of an ecosystem for developing digital-enabled service design tools, and to specify a roadmap for creating such an ecosystem—that is, a series of steps or events that achieve the desired output [65], where 'event' means something that causes a certain effect [65] as a mechanism that influences the overall goal. As a framework for theory building, the proposed roadmap takes the form of five design objectives based on previously untested relationships [64] as a foundation for the creation

of such an ecosystem. These design objectives are derived from a literature review of existing knowledge about the formation of software ecosystems. The evaluation of the artefact following step 5 is artificial since it is not applied in a real-world scenario [63, 67]. The proposed solution was discussed with a group of researchers from the fields of service innovation and digital technologies. Hence, the artefact was evaluated against the criteria suitability, importance, applicability and novelty [63]. This results in a justified problem statement, a justified research gap and justified design objectives [63]. Step 6 is conducted by this publication and addresses itself primarily to the scholar's community.

4 Conceptual Framework

The five design objectives set out below are prerequisites for an ecosystem for the development of digital-enabled service design tools (cf. Fig. 1).

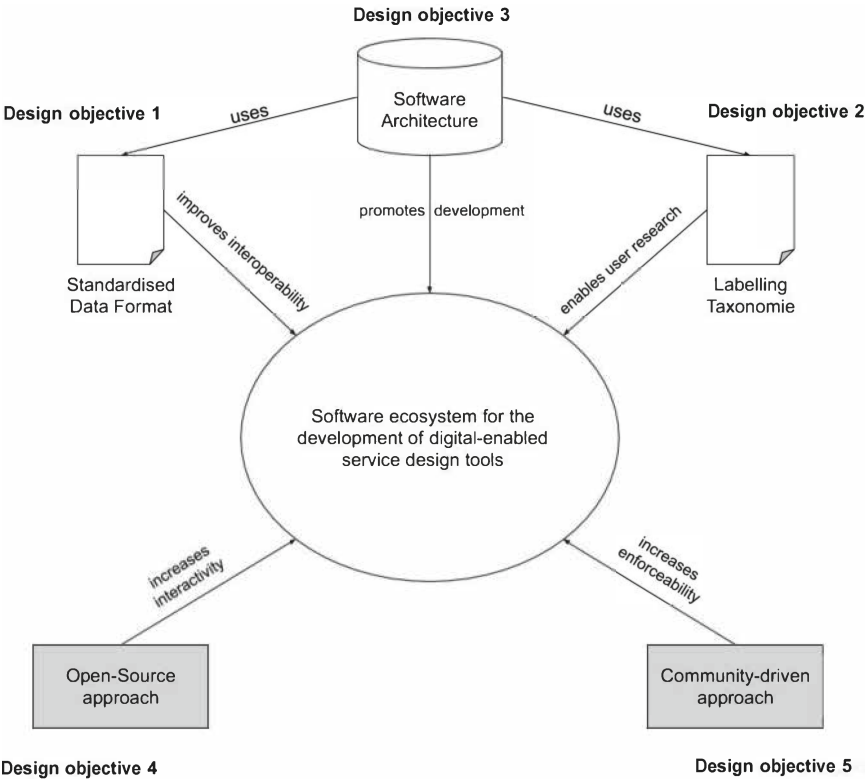

Fig. 1. Fostering ecosystem formation: key design objectives

As mentioned above, many tools build on each other to derive a process or method; others enable a differentiated level of abstraction through mutual use. For example, the *TRIGGER* method developed by Höckmayr and Roth uses four tools for the systematic

development of digital-enabled service systems [18]. Another multistage process is classically applied in service design as follows: First, one or more *Personas* are designed, and a *Customer Journey* is then created, depicting each customer's various touchpoints. Based on this, a *Stakeholder Map* can then be drawn for the various actors [21]. Both processes use different tools whose results build on each other, but they also share similar core components. Both the *Job Map* within the *TRIGGER* method from the first example and the *Customer Journey* from the second example map the customer's activities to achieve a specific goal. Both represent a specific sequence of activities, as does the *Service Blueprint* [15], but they differ in terms of the involvement of different actors or the collection of additional information. While the *Customer Journey* records the touchpoints [21], the *Job Map* summarises activities as higher-level universal steps that are necessary to get a specific job done [68]. The tools, therefore, partly have similar structures with different additional information. Using a standardised data format, data can be transferred between different tools within a single design process such as *TRIGGER*.

As another possibility, one or more tools can be used for the design process, and the resulting result can be evaluated using another tool; this also requires the transfer of data or results. In general, digital data can be characterised as unstructured, semi-structured or structured [69]; tools require structured data for processing, and this, in turn, requires a standardised data format to ensure the interoperability of different tools. Digital-enabled service design tools could thus be developed by several actors who can exchange intermediate results via a standardised structured data format. Suppose three different digital-enabled service design tools were developed by three different actors: One to design personas, another one to design customer journeys, and a third one to design stakeholder maps. The output of the first tool can be used directly as input for a second tool via the structured data format and so on. In this way, the structured data format provides a link between otherwise independent digital-enabled service design tools to enable an iterative design process as is common in service design. Referring to the previous examples, one digital-enabled service design tool could help a user to design a persona. Properties of the designed persona can then be exported to a file that conforms to the standardised structured data format. The service designer could then use a second digital-enabled tool, which is specialised in designing a customer journey, to import the persona from this file and design the appropriate customer journey. The third tool could be used independently of tool two and could also process the output data from tool one.

A standardised data format used by a single instrument can also increase the value of the ecosystem, as standards provision generally increases innovation within an ecosystem [47]. For example, it could be used to develop software that digitises the results of an offline workshop and transfers them into an equivalent digital version for further editing. The use of standards enables developers to include functions for saving intermediate results, which supports documentation of the design process. In the case of digital tools, in particular, this capability supports service designers in creating different alternative scenarios and exploring multiple solutions [10]. This results in:

Design Objective 1: A standardised structured data format should be designed to improve the interoperability of digital-enabled service design tools.

Many service design tools require specific knowledge about the customer (e.g., demographic data of a *Persona*), customer goal or job in the *Job Map*. Service designers typically undertake research to acquire knowledge about the customer. Digital technologies can support user research—for example, by enabling customer data collection [e.g., 60]. For instance, the mobile application ExperienceFellow[6]: The app can capture different types of data (text, video or images) to document emotional experiences at customer touchpoints. Digital tools of this kind can also be used to collect customer data at various points in the company or during the product or service life cycle [e.g., 61]; for example, customer support can actively record negative or positive customer experiences for later evaluation or make use of existing data silos either within (e.g., CRM, ERP systems) or outside the company (e.g., social media) [71]. For digital-based business models, in particular, it is useful to collect customer data at various points for further analysis [72], using machine learning and computer algorithms to extract meaningful information for service innovation and service design [73]. Beyond social media and CRM systems, new data sources may be identified in the future [73, p. 36]. A standardised labelling taxonomy can also enhance the development of tools in the ecosystem for digital-enabled service design tools. Standardised labelling facilitates the development of specialised tools for data collection and others for analysing and evaluating those data. This results in:

Design Objective 2: A standardised labelling taxonomy for unstructured data should be designed to facilitate digitisation of user research.

The development of interoperable digital-enabled service design tools can also be enhanced by standardised modular software architecture. In general, the term *software architecture* refers to the core set of design decisions that define the software system itself [49], which is an important determinant of software ecosystem success [48, 52]. This includes, for example, the data formats recommended in *Design objective 1 and 2*. A well-designed architecture enhances the development of digital artefacts; for example, if each digital-enabled service design tool is constructed as a single module within a central software solution, a design method can then be represented as a process that calls individual modules in a specific order and transfers data between them. Therefore, individual design tools such as the *Job Map* or the *Service Blueprint* represent add-ons or modules for this solution. A modular software design can create considerable added value, as it makes it easier to experiment and validate different software components [74]. For instance, a specific service design process may already be well formed and validated, but the corresponding user interface of a digital-enabled service design tool may still have deficiencies. This could particularly occur when existing service design tools are converted into a digital-enabled version. Furthermore, functions necessary for such software, such as saving, undoing and redoing actions [10], can be implemented more easily, as design patterns already exist within the software architecture. However complex the architecture or its purpose, it should be easy to understand, communicate and document [52]. Besides, the architecture should be designed to allow changes, adaptations or new versions to be easily implemented [52]. This results in:

[6] For a more detailed description follow the link: https://www.experiencefellow.com.

Design Objective 3: A standardised modular software architecture should be designed to promote the development of digital-enabled service design tools for both science and industry.

Open source software is often characterised as a private public good produced by a community, involving such contributions as software, hardware, expertise or sponsorship [75] in various forms, which together with free software are referred to as Free/Libre/Open Source Software (FLOSS). FLOSS approaches are now widely used in the software industry to foster ecosystems, as this type of platform meets many of the necessary conditions [52]. It should be emphasised that an open source approach does not preclude partially proprietary use or distribution within the ecosystem; on the contrary, the ecosystem may even benefit from this approach [48, 76]. The content management system WordPress is a good example of an open source project that combines free, libre and proprietary software add-ons and a service ecosystem through its plugin system. Even in cases of partial proprietary use, the scientific community should be granted a free right to use for research purposes, based on a suitable licence (which may have to be developed) [75, 77]. Without appropriate licensing terms, there is no guarantee that researchers will be able to access the source code, which is important, for example, in promoting the further development of service design methods that use different tools in a specific order. An open source approach also has other advantages: As previous research has shown, open source communities can achieve better modularity than projects developed by a single actor [78]. This results in:

Design Objective 4: An ecosystem for digital-enabled service design tools should be designed through an open source approach to increase interactivity among actors.

For complex software systems, the ecosystem approach has proven to be useful [78]. Due to various existing barriers (technical, domain competency etc.), a large number of different stakeholders are usually involved in the design process, which has a positive effect on software development [78]. As mentioned above, this is also very much in the nature of the developing of digital-enabled service design tools. For example, the academic community probably has very high expertise in the domain of service design. However, the technical implementation, maintenance and support might be a barrier here. These in turn may be better provided by private sector actors, who in turn may develop their own business model. The academic community can again benefit from this, as developed IT artefacts may be easier to test in real-world scenarios. By combining the different expertise of these different stakeholders, the value generated for the ecosystem for digital-enabled service design tools can be increased, as research on software ecosystem shows [78]. However, there are also some peculiarities of a community-driven approach, such as software ecosystems are typically driven and controlled by a central actor or hub [47]. To the extent that private sector actor provisions are uncertain, we contend that a community-driven open source approach is essential for creating and enforcing such a system. Although many open source projects are community-driven, the term open source usually refers to software licensing while *community-driven* refers

to the perspective of the main driving actor [75]. With reference to the open source software approach advanced in *Design objective 4*, various success factors have already been investigated [79]. In general, an open source software project can be seen as a virtual organisation that bundles competencies to drive development forward [80]. Regarding *Design objective 4*, community suitability and activity are essential for a successful open source project [52]. Similar community-driven approaches already exist in other areas of the scientific community; for example, the Open Science Framework (OSF) platform[7] enables researchers to organise research projects and share research data. It also requires an actor or organisation to act as a catalyst for the ideas proposed. This may include the bundled provision of functions and tracking of activities via a community platform—for example, providing a shared code repository or at least linking relevant works or published artefacts as the platform allows. The OSF platform is also an excellent example of providing guidance (on how to conduct Open Science) and facilitating community networking. The platform can also be used for documentation purposes, fulfilling the requirements of *Design objective 1–3*, as well as enabling subsequent use within the ecosystem of extensions developed by the community [52]. This results in:

Design Objective 5: A community should be designed to increase enforceability of the ecosystem for digital-enabled service design tools.

5 Artefact Evaluation

In an artificial ex ante evaluation – in specific EVAL 1 – following Sonnenberg and vom Brocke [63], the problem statement, the research gap and the design objectives were justified. The proposed theoretical framework was demonstrated to a group of researchers as well as two practitioners and was discussed afterwards. This evaluation is in its nature artificial; however, DSR artefact designs have to be justified and validated before they have been put into use [63]. The goal was to evaluate each *Design objective* against its suitability, importance, applicability and novelty. The artefacts suitability and applicability were confirmed as the proposed design objectives are based on broad fundaments and already sufficiently researched areas of information systems. In contrast, the novelty within the group of researchers was questioned. One of the participants stated that although the proposals were not new, they were applicable to the problem. However, the authors argue that it is not the proposed design objectives that should be seen as novel, but rather their application within the field of service design. For example, one of the practitioners said that networked or interoperable tools are very important as participants in longer workshop sessions often lose interest in transferring intermediate results. The authors conclude from the evaluation results that the proposed design objectives offer a starting point for further research. The evidence will of course have to be further empirically proven within the next evaluation cycles.

6 Discussion

These findings have several implications for service design research and practice. As discussed above, building an ecosystem for the development of digital-enabled service

[7] The platform is accessible at https://osf.io.

design tools affords new opportunities for the evolution of such tools, and the five design objectives advanced here serve as a starting point or roadmap. Although making no claim to completeness, the design objectives are mutually dependent. This does not imply simultaneity; implementation can be modular and linear or non-linear. In terms of simplicity of implementation, *Design objective 1* seems the most reasonable; science and practice should agree on a common standard whose form remains to be further evaluated. *Design objective 3* raises the question of whether a universal architecture is possible—for example, whether the transfer of such software to other platforms should be facilitated [51].

For any realisation in practice, it should be noted that a number of factors can influence the success of implementation, especially concerning *Design objective 4* and *5* [e.g., 68, 69]. These are again intertwined, as *Design objective 5* is an influence variable for *Design objective 4*. A combination of *Design objective 4* and *5*—that is, a community-driven open source project—can help researchers to make IT artefacts more stable and persistent for subsequent use. For example, a faulty code base can be corrected and enhanced by other community members [52, 77]. It should also be mentioned that the ecosystem in question must be designed to provide some incentive for the actors involved [77, 81], who can, in turn, reap certain benefits [77]. For example, science can contribute new ideas [81] or the validation of tools, methods and techniques developed by the community. The community is therefore among the most important factors for long-term ecosystem success [52]. In turn, practitioners can benefit from a greater flow of ideas that can be directly exploited.

7 Conclusion

7.1 Contribution

This paper contributes to the Information Systems knowledge base by arguing the need for an ecosystem for digital-enabled service design tools and by proposing a conceptual framework for this endeavour. As a result, a justified problem statement and a justified research gap, which can serve as a basis for further research. Therefore, the terms *digital service design* and *digital-enabled service design* should be differentiated to distinguish between the different research streams. While the first term refers to research into digital services, the second refers to research into digital technologies for use in service design. To develop a software ecosystem for digital-enabled service design tools, five design objectives were proposed as focus points or roadmap for future research activities. These five design objectives do not claim completeness but represent a snapshot and are subject to modification [82]. The design objectives contribute to the knowledge base in particular as follows. *Design objective 1 to 3* identify essential standardisations within a software ecosystem for the development of digital-enabled service design tools, proposing two data formats and a software architecture construct. One of these standardised data formats would increase tool interoperability, and the other would enable the collection of data for use in service design. A standardised software architecture to increase the reuse of software components is proposed in *Design objective 4,* which addresses the creation of rules or licenses within the ecosystem and recommends an open source approach to

promote further development in research and practice. Finally, *Design objective 5* suggests a community-driven approach to building and leading the ecosystem. The *Design objective 1 to 5* were justified [63] for the further design of one or more concrete artefacts, which can be applied to real-world scenarios. A contribution to the knowledge base within the DSR is also seen as a contribution if the presented solution is transferable to other similar problems [59]. Although this paper discusses the application in the field of service design tools, the transfer to similar domains as innovation design is conceivable, so that other researchers could benefit from the presented metamodel.

7.2 Limitations

As our mostly conceptual DSR approach mostly focused on the integration and creation of new relationships between structures without reference to data, the ideas advanced remain to be empirically proven [64]. While *Design objective 1 to 3* are more empirical, *Design objective 4 and 5* seem more difficult to verify, as they are complex and entail a number of potential influencing factors. It should also be mentioned that the logic of the argument essentially represents the researchers' own perspective, and other relevant design objectives may not have been considered here. This is legitimate for conceptual work [64, 83], but there remains an obligation to demonstrate need and utility in practice.

7.3 Directions for Future Research

Future research can draw on various points in this work. For example, the above limitations serve as a starting point for empirical research in relation to utility and feasibility. Equally, the design objectives advanced here offer a starting point for design science research or action research projects. Although the individual design objectives together constitute a framework for building an ecosystem for the development of digital-enabled service design tools, a detailed development plan is also needed. By creating and evaluating new and innovative artefacts [57], design-oriented research can lay the essential foundations for such an ecosystem—for example, in the form of data formats developed from *Design objective 1 and 2* or a software architecture based on *Design objective 3.* Concerning *Design objective 4,* future research can help to create an appropriate licensing model for the software ecosystem that takes account of attractiveness for private sector participation as well as research needs. Action research can extend the relevance of academic research to practical application through intensive exchanges between researchers and practitioners [84], which may help to accelerate ecosystem formation. Furthermore, the implementation of the proposed design objectives could possibly lead to conclusions on how software ecosystems can be created or strengthened.

References

1. Barrett, M., Davidson, E., Fayard, A.-L., Vargo, S.L., Yoo, Y.: Being innovative about service innovation: Service, design and digitalization. In: Thirty Third International Conference on Information Systems (ICIS), Orlando, pp. 1–6 (2012)
2. Chesbrough, H., Spohrer, J.: A research manifesto for service science. Commun. ACM **49**, 35–40 (2006)

3. Vargo, S.L., Lusch, R.F.: Evolving to a new dominant logic for marketing. J. Mark. **68**, 1–17 (2004)
4. Vargo, S.L., Lusch, R.F.: Service-dominant logic: continuing the evolution. J. Acad. Mark. Sci. **36**, 1–10 (2008)
5. Patrício, L., Gustafsson, A., Fisk, R.: Upframing service design and innovation for research impact. J. Serv. Res. **21**, 3–16 (2018)
6. Ostrom, A.L., Parasuraman, A., Bowen, D.E., Patrício, L., Voss, C.A.: Service research priorities in a rapidly changing context. J. Serv. Res. **18**, 127–159 (2015)
7. Ulwick, A.W.: What Customers Want: Using Outcome-Driven Innovation to Create Breakthrough Products and Services. McGraw-Hill Education Ltd, New York (2005)
8. Spohrer, J., Maglio, P.P.: The emergence of service science: toward systematic service innovations to accelerate co-creation of value. Prod. Oper. Manag. **17**, 238–246 (2008)
9. Alter, S.: Metamodel for service design and service innovation: integrating service activities, service systems, and value constellations. In: Thirty Second International Conference on Information Systems (ICIS), Shanghai, pp. 2529–2548 (2011)
10. Shneiderman, B.: Creating creativity: user interfaces for supporting innovation. ACM Trans. Comput. Interact. **7**, 114–138 (2000)
11. Siemon, D., Narani, S.K., Robra-Bissantz, S.: The benefits of creativity support systems for entrepreneurs: an exploratory study. In: Twenty-Third Americas Conference on Information Systems (AMCIS), Boston, pp. 1–10 (2017)
12. Ostrom, A.L., et al.: Moving forward and making a difference: research priorities for the science of service. J. Serv. Res. **13**, 4–36 (2010)
13. Maglio, P.P., Vargo, S.L., Caswell, N., Spohrer, J.: The service system is the basic abstraction of service science. Inf. Syst. E-bus. Manag. **7**, 395–406 (2009)
14. Böhmann, T., Leimeister, J.M., Möslein, K.: Service systems engineering. Bus. Inf. Syst. Eng. **6**, 73–79 (2014)
15. Shostack, G.L.: Designing services that deliver. Harv. Bus. Rev. **62**, 133–139 (1983)
16. Patrício, L., Fisk, R.P., Falcão E Cunha, J.: Designing multi-interface service experiences: the service experience blueprint. J. Serv. Res. **10**, 318–334 (2008)
17. Lim, C.-H., Kim, K.-J.: Information service blueprint : a service blueprinting framework for information-intensive services. Serv. Sci. **6**, 296–312 (2014)
18. Höckmayr, B., Roth, A.: Design of a method for service systems engineering in the digital age. In: Proceedings of the 38th International Conference on Information Systems (ICIS), Seoul, pp. 1–23 (2017)
19. Osterwalder, A.: Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers. Wiley, Hoboken (2010)
20. Rogers, D.L.: The Digital Transformation Playbook: Rethink Your Business for the Digital Age. Columbia Business School Publishing, New York (2016)
21. Stickdorn, M., Hormess, M.E., Lawrence, A., Schneider, J.: This is Service Design Doing: Applying Service Design Thinking in the Real World. O'Reilly Media Inc, Sebastopol (2018)
22. Knapp, J.: Sprint: How to Solve Big Problems and Test New Ideas in Just Five Days. Simon & Schuster Paperbacks, London (2016)
23. Zolnowski, A., Weiß, C., Böhmann, T.: Representing service business models with the service business model canvas - the case of a mobile payment service in the retail industry. In: Proceedings of the Annual Hawaii International Conference on System Sciences, pp. 718–727 (2014)
24. Turetken, O., Grefen, P.: Designing service-dominant business models. In: Proceedings of the 25th European Conference on Information Systems (ECIS), pp. 2218–2233 (2017)
25. Gilsing, R., Turetken, O., Ozkan, B., Adali, O.E.: A method for qualitative evaluation of service-dominant business models. In: European Conference on Information Systems - ECIS 2020, pp. 1–15. Association for Information Systems (2020)

26. Hund, A., Drechsler, K., Reibenspiess, V.: The current state and future opportunities of digital innovation: a literature review. In: Proceedings of the 27th European Conference on Information Systems (ECIS) (2019)
27. Nambisan, S., Lyytinen, K., Majchrzak, A., Song, M.: Digital innovation management: reinventing innovation management research in a digital world. MIS Q. **41**, 223–238 (2017)
28. Liu, X., Werder, K., Maedche, A.: A taxonomy of design techniques for digital services. In: ICIS 2016 Proceedings, Dublin, pp. 1–12 (2016)
29. Williams, K., Chatterjee, S., Rossi, M.: Design of emerging digital services: a taxonomy. Eur. J. Inf. Syst. **17**, 505–517 (2008)
30. O'Hern, M., Rindfleisch, A.: Customer co-creation: a typology and research agenda. Rev. Mark. Res. **6**, 84–106 (2008)
31. Jonas, J.M., Roth, A., Möslein, K.M.: Stakeholder integration for service innovation in German medium-sized enterprises. Serv. Sci. **8**, 320–332 (2016)
32. Jonas, J.M., Roth, A.: Stakeholder integration in service innovation - an exploratory case study in the healthcare industry. Int. J. Technol. Manag. **73**, 91–113 (2017)
33. Li, M.M., Peters, C.: From service systems engineering to service innovation - a modeling approach. In: Proceedings of the 27th European Conference on Information Systems (ECIS), Stockholm & Uppsala, pp. 1–16 (2019)
34. Seidel, S., Müller-Wienbergen, F., Becker, J.: The concept of creativity in the information systems discipline: past, present, and prospects. Commun. Assoc. Inf. Syst. **27**, 217–242 (2010)
35. Morris, D., Secretan, J.: Computational creativity support: using algorithms and machine learning to help people be more creative. In: CHI 2009 Extended Abstracts on Human Factors in Computing Systems, pp. 4733–4736 (2009)
36. Fritscher, B., Pigneur, Y.: Business model design: an evaluation of paper-based and computer-aided canvases. In: Proceedings of the 4th International Symposium on Business Modeling and Software Design (BMSD), pp. 236–244 (2014)
37. Szopinski, D., Schoormann, T., John, T., Knackstedt, R., Kundisch, D.: Software tools for business model innovation: current state and future challenges. Electron. Mark. (2019)
38. Zec, M., Dürr, P., Schneider, A.W., Matthes, F.: Improving computer-support for collaborative business model design and exploration. In: BMSD 2014 – Proceedings of the 4th International Symposium on Business Modeling and Software Design, pp. 29–37 (2014)
39. Thomas, O., Hagen, S., Frank, U., Recker, J., Wessel, L., Thomas, O.: Global crises and the role of BISE. Bus. Inf. Syst. Eng. (2020)
40. Becker, J., Beverungen, D., Knackstedt, R., Matzner, M., Müller, O., Pöppelbuß, J.: A framework for design research in the service science discipline. In: 15th Americas Conference on Information Systems (AMCIS), pp. 1–9 (2009)
41. Brinkkemper, S.: Method engineering: engineering of information systems development methods and tools. Inf. Softw. Technol. **38**, 275–280 (1996)
42. Poeppelbuss, J., Lubarski, A.: A classification framework for service modularization methods. Enterp. Model. Inf. Syst. Archit. **13**, 11–14 (2018)
43. Chai, K.H., Zhang, J., Tan, K.C.: A TRIZ-based method for new service design. J. Serv. Res. **8**, 48–66 (2005)
44. Palvia, P., Nosek, J.T.: A field examination of system life cycle techniques and methodologies. Inf. Manag. **25**, 73–84 (1993)
45. Chasanidou, D., Gasparini, A., Lee, E.: Design thinking methods and tools for innovation. In: Marcus, A. (ed.) DUXU 2015. LNCS, vol. 9186, pp. 12–23. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-20886-2_2
46. Shneiderman, B.: Creativity support tools: accelerating discovery and innovation (2007)

47. van den Berk, I., Jansen, S., Luinenburg, L.: Software ecosystems: a software ecosystem strategy assessment model. In: Proceedings of the Fourth European Conference on Software Architecture: Companion Volume (ECSA), pp. 127–134. Association for Computing Machinery, Copenhagen (2010)
48. Manikas, K., Hansen, K.M.: Software ecosystems-a systematic literature review. J. Syst. Softw. **86**, 1294–1306 (2013)
49. Medvidovic, N., Taylor, R.N.: Software architecture theory and practice. In: ACM/IEEE 32nd International Conference on Software Engineering (ICSE), pp. 471–472. IEEE, Cape Town (2010)
50. Bosch, J.: From software product lines to software ecosystems. In: 13th International Software Product Line Conference, pp. 111–119 (2009)
51. Jansen, S., Finkelstein, A., Brinkkemper, S.: A sense of community: a research agenda for software ecosystems. In: 31st International Conference on Software Engineering - Companion Volume (ICSE), pp. 187–190 (2009)
52. Kilamo, T., Hammouda, I., Mikkonen, T., Aaltonen, T.: From proprietary to open source - growing an open source ecosystem. J. Syst. Softw. **85**, 1467–1478 (2012)
53. Hanssen, G.K.: A longitudinal case study of an emerging software ecosystem: implications for practice and theory. J. Syst. Softw. **85**, 1455–1466 (2012)
54. Christensen, H.B., Hansen, K.M., Kyng, M., Manikas, K.: Analysis and design of software ecosystem architectures - towards the 4S telemedicine ecosystem. Inf. Softw. Technol. **56**, 1476–1492 (2014)
55. Farrell, J., Garth, S.: Standardization, compatibility, and innovation. RAND J. Econ. **16**, 70–83 (1985)
56. Burkard, C., Widjaja, T., Buxmann, P.: Software ecosystems. Bus. Inf Syst. Eng. **4**, 41–44 (2012)
57. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**, 75–105 (2004)
58. Hevner, A., Samir, C.: Integrated Series in Information Systems, vol. 28 (2010)
59. Gregor, S., Hevner, A.R.: Positioning and presenting design science types of knowledge in design science research. MIS Q. **37**, 337–355 (2013)
60. March, S.T., Smith, G.F.: Design and natural science research on information technology Salvatore. Decis. Support Syst. **15**, 251–266 (1995)
61. Ostrowski, Ł., Helfert, M., Xie, S.: A conceptual framework to construct an artefact for meta-abstract design knowledge in design science research. In: Proceedings of the Annual Hawaii International Conference on System Sciences, pp. 4074–4081 (2012)
62. Peffers, K., Tuunanen, T., Rothenberger, M.A., Chatterjee, S.: A design science research methodology for information systems research. J. Manag. Inf. Syst. **24**, 45–77 (2007)
63. Sonnenberg, C., vom Brocke, J.: Evaluations in the science of the artificial – reconsidering the build-evaluate pattern in design science research. In: Peffers, K., Rothenberger, M., Kuechler, B. (eds.) DESRIST 2012. LNCS, vol. 7286, pp. 381–397. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29863-9_28
64. Gilson, L.L., Goldberg, C.B.: Editors' comment: so, what is a conceptual paper? Gr. Organ. Manag. **40**, 127–130 (2015)
65. Jaakkola, E.: Designing conceptual articles: four approaches. AMS Rev. **10**(1–2), 18–26 (2020). https://doi.org/10.1007/s13162-020-00161-0
66. MacInnis, D.J.: A framework for conceptual contributions in marketing. J. Mark. **75**, 136–154 (2011)
67. Prat, N., Comyn-Wattiau, I., Akoka, J.: A taxonomy of evaluation methods for information systems artifacts. J. Manag. Inf. Syst. **32**, 229–267 (2015)
68. Bettencourt, L.A., Ulwick, A.W.: The customer-centered innovation map. Harv. Bus. Rev. **86**, 109–114 (2008)

69. Baars, H., Kemper, H.G.: Management support with structured and unstructured data - an integrated business intelligence framework. Inf. Syst. Manag. **25**, 132–148 (2008)
70. Murthy, D.: Digital ethnography: an examination of the use of new technologies for social research. Sociology **42**, 837–855 (2008)
71. Otto, B., Bärenfänger, R., Steinbuß, S.: Digital business engineering: methodological foundations and first experiences from the field. In: 28th Bled eConference: #eWellbeing – Proceedings, pp. 58–76 (2015)
72. Brenner, W., et al.: User, use & utility research: the digital user as new design perspective in business and information systems engineering. Bus. Inf. Syst. Eng. **6**, 55–61 (2014)
73. Antons, D., Breidbach, C.F.: Big data, big insights? Advancing service innovation and design with machine learning. J. Serv. Res. **21**, 17–39 (2018)
74. Sullivan, K.J., Griswold, W.G., Cai, Y., Hallen, B.: The structure and value of modularity in software design. In: Proceedings of the ACM SIGSOFT Symposium on the Foundations of Software Engineering, pp. 99–108 (2001)
75. O'Mahony, S.: Guarding the commons: how community managed software projects protect their work. Res. Policy. **32**, 1179–1198 (2003)
76. Campbell-Kelly, M., Garcia-Swartz, D.D.: The move to the middle: convergence of the open-source and proprietary software industries. SSRN Electron. J. **17**, 1–39 (2011)
77. Lerner, J., Tirole, J.: The simple economics of open source. SSRN Electron. J. L (2005)
78. Cataldo, M., Herbsleb, J.D.: Architecting in software ecosystems: interface translucence as an enabler for scalable collaboration. In: ACM International Conference Proceeding Series, pp. 65–72 (2010)
79. Radtke, N.P., Janssen, M.A., Collofello, J.S.: What makes free/libre open source software (FLOSS) projects successful? An agent-based model of FLOSS projects. Int. J. Open Source Softw. Process. **1**, 1–13 (2009)
80. Crowston, K., Scozzi, B.: Open source software projects as virtual organisations: competency rallying for software development. IEE Proc. Softw. **149**, 3–17 (2002)
81. Von Krogh, G.: Open-source software development. MIT Sloan Manag. Rev. **44**, 14–18 (2003)
82. Whetten, D.A.: What constitutes a theoretical contribution? The academy of management review what constitutes a theoretical contribution? Source Acad. Manag. Rev. Manag. Rev. **14**, 490–495 (1989)
83. Hirschheim, R.: Editorial Commentary Some Guidelines for the Critical Reviewing of Conceptual Papers (2008)
84. Rose, R., Grosvenor, I.: Action research. Doing Res. Spec. Educ. Ideas into Pract. **42**, 13–17 (2013)