Factors Enabling and Hindering Value Co-Creation in Continuous Service Development: A Systematic Literature Review

Jenny Elo
Faculty of IT
University of Jyväskylä
jenny.m.elo@jyu.fi

Juuli Lumivalo
Faculty of IT
University of Jyväskylä
juuli.k.lumivalo@jyu.fi

Tuure Tuunanen
Faculty of IT
University of Jyväskylä
tuure@tuunanen.fi

Markus Salo
Faculty of IT
University of Jyväskylä
markus.t.salo@jyu.fi

Abstract

This paper presents a systematic literature review (SLR) investigating the factors that enable and hinder value co-creation in organizations' continuous service development processes. Employing the lens of service-dominant (S-D) logic, we classify the identified factors into three interrelated dimensions: institutions, resources, and service exchange. Our systematic findings may inform organizations' efforts to support the emergence of positive rather than negative value outcomes when implementing continuous practices in their service development. In addition, we outline avenues for further research in this emerging topic area.

1. Introduction

Driven by the opportunities brought by digitalization and the demands of vastly competitive and dynamic markets, organizations across various industries are increasingly adopting continuous development practices to propose value to their customers faster and without compromises in service quality. As one of the most recent phenomena of digital service development, DevOps (development and operations) [1] and continuous practices (e.g., continuous delivery and deployment) have attracted growing interest amongst practitioners and researchers, especially in the software engineering (SE) community [2-4]. The DevOps approach, built on agile and lean principles, breaks down organizational silos and aligns actors' incentives in service development [5-7]. To date, many leading digital service providers, such as Facebook, Netflix, Google, and Amazon, have changed their service development and innovation processes into continuous ones by adopting DevOps and continuous practices (see, e.g., [8]) and reduced their time to deliver service updates from days to minutes down to seconds [4, 9].

Extant literature presents various benefits of continuous development for organizations. For example, increased performance and productivity (e.g., [10-12]), shorter time to market (e.g., [11, 13-15]), cost savings (e.g., [16, 17]), improved quality (e.g., [10-14, 18, 19]), enhanced customer and employee experience and satisfaction (e.g., [16, 17, 20]), and improved employee well-being (e.g., [10, 12, 19, 21]) are among the positive outcomes realized by employing continuous development.

Still, the literature remains scattered and does not effectively explain *how* different outcomes emerge, i.e., *how* continuous development affects value co-creation (VCC) between the focal actors in the organizations' service development ecosystems. Furthermore, arguments have been made that DevOps and continuous practices have not been sufficiently studied in the scientific literature. Instead, the current understanding of these topics is centered on industry and practice (e.g., [4, 18, 22]). To address this gap, we conduct a systematic literature review (SLR) addressing the effects of DevOps and continuous practices on organizations. To this end, we have set the following research question (RQ): *Which factors have been found in the literature to enable/hinder VCC in continuous service development?*

Employing the metatheoretical framework of service-dominant (S-D) logic [23-25], we understand service development ecosystems as "relatively self-contained, self-adjusting system[s] of resource-integrating actors connected by shared institutional arrangements and mutual value creation through service exchange" [24:10-11]. Our findings provide a systematic understanding of the factors that enable and hinder VCC between these ecosystem actors (e.g., internal teams, partners, and customers) and thus may support the emergence of positive rather than negative value outcomes in the organizations' continuous development processes.

From a theoretical perspective, our research expands prior research by focusing on the specific characteristics of continuous service development and explains how they influence VCC. Our research shows that

URI: https://hdl.handle.net/10125/79506
978-0-9981331-5-7

institutions, resources, and service exchange are the deciding dimensions in VCC, all affected by enabling and hindering factors via continuous service development. Furthermore, we identify opportunities for future research.

The remainder of our paper is structured as follows. Next, we provide a background on DevOps and continuous practices as manifestations of continuous development and S-D logic as our lens for understanding VCC in continuous service development. Then, we introduce our research method, followed by our findings. Finally, in the fifth section, we discuss the contribution and implications of our study and conclude with limitations and suggestions for future research.

2. Theoretical background

2.1. Continuous service development

During the past two decades, traditional service development methods (e.g., waterfall) with rigid step-by-step development projects, usually ending with the first major release of the system, have increasingly been replaced with lightweight and iterative development methods for systems and software [7, 22, 26]. In recent years, the development and operations (DevOps) [1] approach in particular has become popular in the SE practitioner and research community [2] as a means to achieve development continuity.

DevOps promotes actor collaboration and automation [4, 27] and relies on various continuous practices that shorten the time between committing a change and deploying it to production while ensuring high service quality [26]. DevOps enables organizations to transform their service development and innovation processes into continuous ones and boost their service delivery speeds from an initial idea to release and from continuous customer and process feedback to rapid enhancements of services to meet customers' dynamic expectations [5].

DevOps is founded on the agile (see Agile Manifesto [28]) and lean principles [29] of software development and enables the scaling of agility to the entire digital service organization. DevOps can be considered a general term that describes various continuous practices (see, e.g., [8]). The most common of these practices are continuous integration, continuous delivery, and continuous deployment [2].

In continuous integration, development teams frequently integrate changes to a system for automatic testing [26, 27]. This enables continuous delivery whereby system development teams release new versions of a working system several times a day through optimization, automation, and the utilization of the build, test, and release process [27]. Continuous deployment goes one step further by automatically releasing a system into production as soon as it is ready [27, 30]. However, continuous practices are also discussed in the literature separate from DevOps. For this reason, in this study, we use the general term "continuous (service) development" to address the implications of these practices and development continuity for VCC in organizations' service development ecosystems.

2.2. S-D logic and VCC in continuous service development

S-D logic [23-25] offers a metatheoretical framework for a systemic understanding of VCC [31]. It identifies service—the process by which actors apply their resources for the benefit of others (or themselves)—as the fundamental basis of exchange, and the core of this exchange is VCC [23, 31]. Moreover, value in this VCC process is considered an emergent, positively or negatively valenced outcome of an actor's well-being or viability [31]. This study focuses on the ways by which the co-creation of value and the positive value outcomes are enabled or hindered by continuous service development.

S-D logic allows for the adjustment of the lens of investigation to different levels of aggregation [32]. For example, it is possible to zoom in to focus on understanding individual actors (micro-level) or zoom all the way out to attain a more holistic understanding of the VCC process among an extensive network of actors (e.g., macro-level; society) and much more in between [24, 33]. Our study focuses on the organizational level and how the actor-to-actor exchange is affected by continuous development in organizations' service development ecosystems. Based on S-D logic, we conceptualize these ecosystems as systems of *resource-integrating* and *service-exchanging* actors (e.g., internal teams, partners, and customers) coordinated by *institutions and their arrangements* for mutual VCC [24, 31].

Institutions in S-D logic are understood as the rules, norms, meanings, practices, and other similar elements enabling and constraining actors' resource integration and service exchange within service ecosystems [24]. Institutions and their essential role as the coordination mechanisms for VCC have been emphasized in the recent literature on S-D logic [24, 31]. Further, S-D logic identifies two broad types of resources—operand (e.g., natural resources) and operant (e.g., knowledge and skills)—that actors integrate for VCC. Finally, the purpose of the interactive service exchange is to enable mutual VCC between the ecosystem actors [23, 24]. These three interrelated dimensions (institutions, resources, and service exchange) constitute the

foundation for our understanding of VCC in continuous service development in this study. The application of the dimensions is described in later sections.

S-D logic has been utilized in various disciplines and domains to introduce and understand the service perspective to exchange and to support the systemic understanding of VCC (see [32] for details). However, apart from individual studies (see, e.g., [34]), its potential has remained untapped in the context of continuous service development. For example, regarding the effects of continuous development, previous studies have mainly focused on presenting the general benefits and challenges of continuous practices for organizations. In this study, we strive to advance this understanding and view S-D logic as the best suited lens to provide an understanding of the actors' activities in service ecosystems and how continuous development may positively or negatively affect the outcomes of these activities.

3. Research methodology

We conducted a systematic literature review (SLR) [35] employing six scientific databases: ProQuest (446), Emerald Insight (89), Science Direct (255), IEEE Explore (354), AIS Library (6), and ACM Digital Library (113). We performed abstract searches in December 2020 using a search string (DevOps OR "development and operations" OR continuous integration OR continuous development OR continuous deployment OR continuous delivery OR continuous "software development" OR continuous "software engineering" OR continuous "system* development" OR continuous innovation) AND (co-crea* OR cocrea* OR co-dest* OR codest* OR collaborat*) AND (service OR product OR system* OR software). The search string was modified slightly for different databases due to limitations and differences in search possibilities (e.g., only nine keywords were allowed in ScienceDirect). Further, databases that did not allow for the searching of only abstracts (e.g., Google Scholar, Springer Link) were not employed.

We evaluated the relevance of the records (n = 1263) to our study in three rounds. First, the records were evaluated solely based on their title. At this stage, the inclusion criteria were 1) The article addresses continuous service development/innovation or DevOps and 2) It is written in English. Based on these criteria, records with titles indicating that they were not relevant to the study were excluded. Records that may have been relevant to the study based on their title were included in the second round. Records with titles that could not unequivocally indicate whether they could be relevant to the study were also included in the second round for abstract review.

Second, applying the same inclusion criteria, the relevance of the remaining records (n = 345) was evaluated based on their abstracts. At this stage, we also eliminated duplicate records (n = 25). After this round, 153 articles remained for the third and final round, where the relevance of the remaining records was evaluated based on their full text. Here, we added three more inclusion criteria to evaluate the records: 3) The article is peer-reviewed, 4) The full-text article is available (to us), and 5) The article focuses on implications (e.g., benefits/challenges) of DevOps or continuous practices. The fifth criterion was added to critically evaluate the remaining records based on their ability to answer our research question and strictly target articles focused on continuous development implications. Subsequently, nine articles were found relevant to be included in our SLR.

The nine included articles were subjected to backward and forward searches to find additional relevant records. The search led to the discovery of 76 additional records (based on title relevance), which were evaluated through the same process (abstract and full-text review) and inclusion criteria as the initial records. This process resulted in 14 more relevant articles for our systematic review, for a total of 23 included articles. The limited number of articles can be attributed to the emerging topic area and our specific focus on the implications of continuous development. More detailed information on the articles is provided in Table 1.

Table 1. Information on the included articles

Publication year		
2021	1	4%
2020	4	17%
2019	2	9%
2018	1	4%
2017	4	17%
2016	7	30%
2015	3	13%
2014	0	0%
2013	1	4%
Publication type		
Conference paper	11	48%
Journal article	10	43%
Magazine (IEEE Software)	2	9%
Method		
Case Study	10	43%
Systematic Literature Review (SLR) and Survey	3	13%
Survey (Quantitative)	2	9%
Interviews (Qualitative)	2	9%
SLR	2	9%
SLR and Interviews	1	4%
Mixed Method	1	4%
Multivocal Literature Review	1	4%
Systematic Mapping Study	1	4%
Continuous development approach/practice(s)		
DevOps	14	61%
Continuous Delivery	4	17%
Continuous Deployment	3	13%
Continuous Delivery and Deployment; DevOps	1	4%
DevOps/Agile	1	4%

For the analysis, the first author made extensive notes on each included article and coded the enabling and hindering factors of VCC found in them in a spreadsheet format. A total of 135 codes related to

enabling factors and 197 codes related to hindering factors were established. The codes (factors and their descriptions) were first inductively labeled (e.g., culture, communication and collaboration, tools) and, through several iterations, classified into representative groups to determine focal categories of VCC in continuous service development. We then further classified the determined categories into three interrelated VCC dimensions (institutions, resources, and service exchange) using the S-D logic lens [24]. Following S-D logic descriptions, categories that represented coordinating factors (e.g., guidelines, practices, assumptions, and beliefs) guiding actors' activities in the continuous development process (e.g., culture, roles and responsibilities) were classified under the institutional dimension. Categories representing integrated resources in continuous development (e.g., technology and tools, knowledge and skills) were classified under the resource dimension. Finally, categories describing interactive exchange elements of VCC (e.g., communication and collaboration) were designated to the service exchange dimension. The categories and their placement in the different VCC dimensions are further described in the next section.

4. Findings

In this section, we present the findings of our systematic review. Figure 1 presents a conceptual framework that, building on S-D logic and extended by our findings, depicts the three interrelated and dynamically interacting VCC dimensions and the focal VCC categories we derived from the continuous development literature.

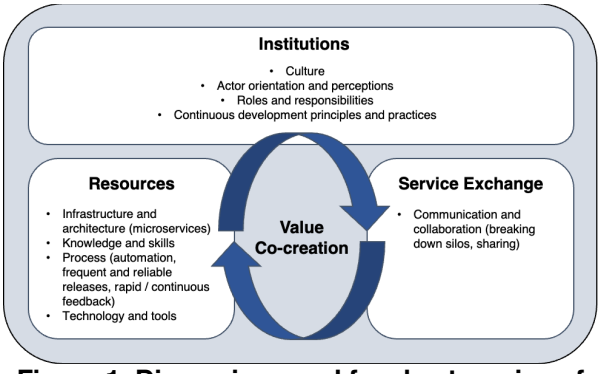

Figure 1. Dimensions and focal categories of value co-creation in continuous service development

All three dimensions of VCC include factors that enable and hinder the co-creation of value in the actor-to-actor networks of organizations' service development ecosystems. Enablers represent factors of continuous development that support the process of VCC between the ecosystem actors within the discovered categories. Hindrances, on the other hand, are factors that undermine the process through institutional, resource, or service exchange reasons. These have often been presented in the previous literature as challenges resulting from the adoption and use of continuous practices.

Based on our literature review, significant categories in the institutional dimension that affect VCC are (organizational) culture, actor perceptions and orientations, roles and responsibilities, and continuous principles and practices, comprising both enabling and hindering factors that influence VCC in continuous service development. Further, our study revealed various categories of resources that affect actors' resource integration in continuous service development either directly or through their actions. These resources include service infrastructure and architecture, knowledge and skills, technology and tools, and the development processes characterized by, for example, automation, frequent and reliable releases, and continuous/rapid feedback. Finally, service exchange includes the interactive category of communication and collaboration and is also affected both positively and negatively by continuous development. Next, we describe the focal findings (VCC categories and factors) under each dimension. A summary of our findings is presented in Table 2.

4.1. Institutions

4.1.1. Culture. DevOps emphasizes culture. Indeed, a few of the reviewed studies report the positive effects of continuous development on organizational culture. They were realized as experiences of a less of a culture of blame [20] and overall advances in organizational culture and mindset [12]. In [20], improved culture was connected to more engaged teams, thus positively affecting VCC. However, cultural challenges might also arise from continuous development, especially at the adoption stage, as significant changes to the cultural mindset are required [15], and changing a longstanding organizational culture may prove difficult [6, 19]. The reviewed literature reports cultural hindrances to VCC, such as a lack of (organizational) motivation to adopt continuous practices [30] and the difficulty of obtaining buy-in from stakeholders [15]. [16] suggests that cultural factors, more so than technical factors, may limit the adoption of continuous development. Furthermore, the lack of organizational vision [3], strategic suggestions from management [36], and lack of management support for implementing continuous practices [18, 19] were among the found challenges.

4.1.2. Perceptions. Continuous development facilitates trust relationships among actors [15, 16, 37]. In [16], continuous development was reported to enable teams to feel more valued, positively affecting their engagement [16]. In turn, actor perceptions may hinder VCC as a result of the challenges of overcoming the Dev versus Ops mentality [36, 38], having to adapt mindsets [18, 19, 39], and actors' unwillingness to accept continuous development [11]. Customers, for example, may not be prepared to handle a shorter release cycle and receive constant updates on service features [14, 30]. In addition, work overload resulting from continuous development was reported as a factor that hinders employee well-being [40]. The most significant perceptual challenge, primarily related to the adoption of continuous development, is resistance to change [3, 14, 17, 20, 36, 38-40]. Resistance may occur at different organizational levels but also from the partner and customer organizations [17].

4.1.3. Orientation. Shared goals and objectives are essential in promoting continuous development and VCC [3]. Development and operations teams were reported to better align their goals to business needs through DevOps [17]. However, actors' focus on their personal goals and priorities [15, 39] was reported as a challenge. Everyone has their interests and approaches to working, and these competing goals may cause tensions, hindering VCC [37, 39]. This challenge is not unique to continuous development, but it reminds us that the transition to continuous development (e.g., breaking down silos) does not imply the automatic alignment of actors' goals and intentions but that organizations should actively enforce working towards a shared goal [41].

4.1.4. Roles and responsibilities. Continuous development enables teams to take on new responsibilities. In [20], increased development responsibilities provided more team autonomy, increasing teams' engagement. In [12], development teams' full ownership of the developed service was reported to remove barriers, bureaucracy, and waiting overhead. Furthermore, granting more power to operations and engaging them in service development from the start was found in [41] to enable VCC. In [42], mixing the responsibilities of development and operations was found to be beneficial, as teaching and learning from each other improved teams' trust relationships. The hindering aspect of roles and responsibilities arises from continuous development causing actors to reassess their established roles and adapt to new tasks and responsibilities [6, 30]. The changes in responsibilities may lead to misunderstandings about who is responsible for which activities [20]. Thus, resource and responsibility accountability issues may occur [36, 39, 42]. Furthermore, broadening the responsibilities of developers may negatively impact their productivity in core tasks [41]. There may also be resistance to the increase in responsibilities [18, 21]. In [42], the introduction of operations tasks to development teams was found to be complicated. Finally, our findings suggest that continuous development may cause friction between development and operations teams due to their different approaches to working [18], thus resulting in a hindrance of mutual VCC.

4.1.5. Standard definition/practices for DevOps. Although introduced in 2009, DevOps still has no standard definition [3, 17] and no standard framework that provides a complete roadmap of its activities [3, 6]. The lack of standard practices and definitions for continuous development may make it difficult for organizations to establish continuity and decide which principles and practices to follow [6], thus hindering the foundations for VCC.

4.2. Resource integration

4.2.1. Infrastructure and architecture. Examples of infrastructural enablers of VCC were also found in the literature. In [10], fewer bugs after service deployment were attributed to automated testing, static code analysis, and production-like environments in the deployment pipeline. Furthermore, infrastructure automation and virtualization were found to lessen operations teams' maintenance workloads [12] and enhance the readiness for infrastructural changes [10]. In [18], DevOps infrastructure automation was found to considerably reduce on-call escalations and false and repetitive alarms. Infrastructure challenges that hindered VCC concerned the building and maintaining the infrastructure and deployment pipeline [11, 15, 16, 36] and challenges in moving from legacy infrastructure to microservices [36, 38]. There may also be a lack of adequate infrastructure to support automation [17]. Achieving compatibility between DevOps and legacy systems and dealing with applications not suited to continuous development were among the reported architectural challenges [3, 15, 17, 19]. [16] notes that to mitigate these challenges, the continuous development ecosystem should be built over time using an approach based on continuous improvement.

4.2.2. Knowledge and skills. Shared technical knowledge between operations and development enforced by continuous development enables VCC, contributing to more frequent releases and faster diagnoses and resolutions of problems [20].

Table 2. Summary of the findings

Dim.	Category	Examples of enabling (+) and hindering (-) factors	References*
Institutions	Culture	+ Improvements in culture	[20, 36]
		- Lack of organizational interest and motivation - Lack of management/stakeholder support	[3, 6, 11, 12, 15–19, 30, 40]
	Orientation	+/- Aligned/competing goals - Lack of team commitment	[3, 15, 17, 37, 39]
	Perceptions	+/- Improved/lack of trust relationship + Improved employee/customer engagement	[15, 16, 20, 36, 37, 42]
		- Resistance to change - Changing employee/customer mindsets - Dev vs. Ops mentality	[3, 11, 14, 17–20, 30, 36, 38–40]
	Roles and responsibilities	+/- Shared responsibilities	[10, 12, 17, 20, 41, 42]
		- Changes/uncertainty in roles and responsibilities - Added responsibilities	[6, 18, 20, 21, 30, 36, 39–42]
	Standard definition / practices for DevOps	- Lack of a standard definition for DevOps - Lack of standard practices/frameworks for DevOps	[3, 6, 17, 42]
Resources	Architecture	- DevOps and legacy systems - Moving from legacy to microservices architecture	[3, 15, 17, 19, 36, 37]
	Infrastructure	+/- Infrastructure automation	[10–12, 17, 18]
		- Moving from legacy infrastructure to microservices - Building the deployment pipeline	[14–16, 36, 38]
	Knowledge and skills	+ Shared knowledge	[10, 20]
		- Lack of/insufficient skills - Lack of training/coaching	[3, 11, 12, 14, 15, 17–20, 30, 36, 39–42]
	Process	+ Rapid/continuous feedback +/- High level of automation (build, test, deployment, etc.) + Frequent and reliable releases	[3, 6, 10–12, 14–20, 37, 38, 40–43]
		- Adapting existing processes to DevOps - Balancing speed vs. quality	[11, 12, 14, 15, 17, 19, 21, 30, 36, 37, 40]
	Technology and tools	+ Tools to support the process (automation) and collaboration	[10, 16, 21, 38, 41, 42]
		- Implementing new technology stacks and tools	[10, 12, 15, 19, 20, 38]
Service Exchange	Communication and collaboration	+ Preventing/breaking down silos + Improved communication and collaboration	[6, 10, 14, 16–18, 20, 40–42]
		- Adjusting to close collaboration - Lack of transparency and sharing	[3, 6, 10, 16, 18, 19, 30, 36, 40, 42, 44]

*The numbers refer to the references in each category (+/- in their own rows), not to the sources of the presented examples

In [10], continuous development via reducing dependence on individual developers and limited tacit knowledge was found beneficial. Knowledge and skills category also included factors that hindered VCC in continuous development, as DevOps demands highly skilled people and the development of new skillsets [15, 30, 41], and it is not always easy to find people with experience and expertise in DevOps to meet these demands [18, 19, 39]. The steep learning curve, again especially at the adoption stage (e.g., new roles and responsibilities, new technology and tools), necessitates high-quality training that provides the skills required for continuous development [20, 40]. The learning and adaptation that continuous development requires from the customer/end-user perspective may also hinder VCC if not adequately supported [11].

4.2.3. Process. Continuous development supports a frequent and reliable release process. This was reported in the reviewed literature, among other things, to enable innovation, build trust relationships, increase confidence, and contribute to a working environment characterized by less waste and stress, better work morale, and job satisfaction [10, 12, 14, 16, 37]. Continuous development processes bring benefits to the organization but also improve the ways of working, positively contributing to the improved well-being of actors [42].

The improved speed and productivity of service delivery is a perceived process benefit of continuous development. Productivity in continuous development is enabled, for example, through continuous integration, testing, and feedback [17], automated deployment [11], releases "with a click of a button" [15], and a more agile

way of specifying requirements [10]. Moreover, the reported enablers behind improved service quality include a more straightforward quality assurance process [10], prototyping [10], deployment in small increments [12, 37], fast feedback cycles [12, 16], and build, test, and deployment automation [14]. Besides productivity and quality, continuous development also supports proposing superior value to customers and end-users, for example, through a flexible process to accommodate change requests at any time [17], the ability to include more features into the pipeline [6], shorter development cycles, the possibility to of frequent release frequently [6], early/rapid feedback from the end-users [6, 15], and testing with real customers (continuous experimentation) [6].

However, various process challenges have the potential to hinder VCC by negatively affecting process resources. These challenges included, among other things, inconsistent environments [36], lack of service virtualization [36], traceability across the DevOps landscape [36], lack of feedback and the prioritization of bugs [36], operations stability [17], the adaptation of existing processes to DevOps [19], difficulty in managing various configurations and run-time environments [11], release planning and managing the fast-paced environment [11], the use of small batches [30], and balancing speed and quality [12, 21].

As revealed through the process section, automation is one of the focal enablers of VCC in continuous development. An increase in automation supports, for example, the consistency of environments across different stages of the software lifecycle [17], renders knowledge explicit and transparent to developers [12], and drives improved service productivity and quality [11, 14, 37]. However, there may also be risks that remain hidden by automation [40], and achieving full automation brings its challenges [3, 42].

Another clear process enabler of continuous development is rapid and continuous feedback. The presence of a short feedback loop at each service development stage enables teams to produce high-quality software and identify and solve problems faster [10, 14, 16, 17, 38, 41, 43]. As disclosed in earlier paragraphs, continuous feedback about users also facilitates the development of relevant service features and functionalities, thus enabling better customer value propositions [11, 12, 14, 37, 40]. It also enables discovering alternative solutions to make better decisions on what service features to (dis)continue developing [14]. Besides user/customer feedback, continuous and real-time feedback from the entire deployment pipeline and automated infrastructure enable the more rapid identification and resolution of problems [11].

4.2.4. Technology and tools. Continuous development tools (see, e.g., [16, 26, 38]) enabling, for example, collaboration, and management and automation of the development process, are central to continuous development and key to supporting VCC activities presented in many of the other categories. Challenges related to technology and tools, potentially hindering VCC concerned providing complex technology environments needed for DevOps [19], implementing DevOps technology [19], setting up and managing the tools for the deployment pipeline [10, 15, 19, 20], scarcity of tools [44], and Dev and Ops teams having separate toolsets and metrics [38].

4.3. Service exchange

4.3.1. Communication and collaboration. Continuous development prevents and breaks down functional and physical silos [16, 17] and supports increased collaboration among actors [14, 17, 20]. In [18], improved communication was found to improve the speed and effectiveness of problem-solving. Increased collaboration also improves knowledge and experience sharing between teams. In [42], collaboration enabled VCC as development and operations teams became more trusting and understanding of each other. The communication and collaboration category also included factors that hinder VCC. The literature included reports of communication and collaboration issues [36], challenges in achieving effective communication [16, 19], hardware dependency (in the embedded domain) [44], exhaustion from being available for close collaboration [10], challenges adjusting to new ways of working [42], and fostering transparency and the culture of sharing [3].

5. Discussion and conclusion

We have reviewed and synthesized existing continuous development literature to determine factors enabling and hindering VCC in continuous service development. Our findings show that institutions (culture, actor perceptions and orientation, roles and responsibilities, and continuous principles and practices), resources (architecture and infrastructure, knowledge and skills, processes, and technology and tools), and service exchange (communication and collaboration) are the deciding dimensions of VCC, all of which are affected by enabling and hindering factors via continuous development.

Our contribution to the literature is threefold. First, our study expands the theoretical understanding of S-D logic and VCC in the context of continuous service development. Through our synthesis of hindering and

enabling VCC factors based on the reviewed studies, we deepen the understanding of the S-D logic's meta-theoretical constructs also from a practical perspective.

Second, through the application of S-D logic, our research goes beyond the synthesizing of general benefits and challenges of continuous development to organizations that currently dominate the understanding of development continuity effects in the continuous development literature. Instead of merely focusing on the outcomes (e.g., benefits), we address how these outcomes are enabled and hindered (i.e., how continuous development affects VCC between the focal actors in the organizations' service development ecosystems). To the best of our knowledge, such a perspective on continuous development has not been presented before. The identification of the enabling and hindering factors and the affected VCC categories and dimensions may reveal new opportunities for both research and practice for studying and improving the VCC potential resulting from continuous service development in organizations. The advanced understanding may help managers in their efforts to support positive rather than negative value outcomes in their continuous service development processes.

Our review, while systematic, by no means attempts to provide an exhaustive list of all possible factors that may support and hinder the co-creation of value in continuous development. However, the dimensions and categories of VCC we have identified through our analysis can help managers consider the full effects of continuous development from institutional, resource, and service exchange perspectives. As an important implication for research and practice, our study highlights that continuous development is more than a technical or resource issue. This is shown, for example, by our findings of various institutional factors that enable and hinder VCC among actors in the service development ecosystem, which are in line with the latest emphases of S-D logic [24, 31].

Further, we argue that understanding both positive (enabling) and negative (hindering) factors affecting VCC is essential for organizations to efficiently support VCC between ecosystem actors. The presented categories and examples of factors can enable managers to consider the levels at which the identified enablers are realized in their service development processes. It is equally important, if not more important, to identify the hindrances to be avoided or mitigated in order to strengthen VCC and realize the potential benefits of continuous development described previously.

Third, through our review, we have identified gaps in the current stream of literature that pose exciting opportunities for future research contributions on this topic. First, many challenges of continuous development reported in the extant literature are related explicitly to the introduction and adoption of continuous practices in organizations (e.g., resistance to change, moving from legacy systems to microservices, adjusting to close collaboration). This reflects the novelty of continuous development as a phenomenon and for organizations. In the future, it would be particularly interesting to examine the enablers and hindrances to VCC that are identified in continuous development processes after continued use. In this regard, longitudinal research could be conducted to study how the practices and implications change over time.

Another compelling continuation of our research is to move from the identified factors to understanding the mechanisms of VCC in continuous development (i.e., an empirical investigation on what and how dynamic value outcomes emerge in the continuous service development processes). While this study identified factors that potentially support or undermine realizing the VCC potential in organizations' continuous service development processes, actor-specific experiences of process outcomes and their underlying mechanisms remain uncovered. We are currently taking steps to address this gap through an empirical investigation conducting semi-structured interviews with multiple case organizations from different digital service domains. We welcome other researchers to join us in this endeavor.

Furthermore, current research on continuous development has focused mainly on the web domain, although research has started to expand, for example, to the embedded systems context (see, e.g., [44]). We see potential in moving from software-intensive web applications to understanding continuous development in other service contexts. For example, cyber-physical systems (CPS) enabled services that represent a new frontier for service research [45, 46] could provide a topical context where, for example, connectivity, data, and the merging of cyber and physical worlds introduce exciting opportunities for investigation. The ways in which the characteristics of these kinds of services affect and are affected by continuous development is an exciting topic for future research.

Finally, as our review shows, continuous development literature focuses significantly on the internal actors of organizations (especially the development and operations teams). Expanding the understanding of how continuity affects and can be supported in more extensive networks of actors (e.g., from the perspective of engaging partners, customer organizations, and end-users) in the service development process and the means of leading continuity through the entire organization's development and innovation activities are also interesting topics to explore.

Our study also has limitations. First, the articles included in the review may be limited due to our choice of databases, the design of the search string, and the abstract-level search. We aimed to actively reduce these limitations by executing trials on different keyword combinations and conducting backward and forward searches on the relevant articles. Through these actions, we are confident that the set of included literature provides a firm base for our conceptualizations of factors, categories, and dimensions focal to VCC. Furthermore, the number of included articles in our review reflects the topic's novelty and our decision to focus on reported implications of continuous development. Second, the analysis and coding process of the enabling and hindering VCC factors was based primarily on the first author's interpretations. However, this limitation was mitigated by the provision of extensive notes of the process and comprehensive discussions on the interpretations and findings with the other authors of the paper.

To conclude, this paper is the first step towards our research on VCC in continuous service development and innovation. We encourage others to contribute to this emerging topic, hopefully motivated by our findings and suggestions for future contributions. As our study shows, continuous development is an evolving, multifaceted, socio-technical phenomenon into which research from diverse fields can provide novel insights.

6. Acknowledgements

This research has been partly funded by the Foundation for Economic Education, Finland [grant number 34014860].

7. References

[1] P. Debois, "Devops: A Software Revolution in the Making?", Cutter IT Journal, 8(24), 2011.

[2] D. Stahl, T. Martensson, and J. Bosch, "Continuous practices and devops: beyond the buzz, what does it all mean?", In Proceedings of the 43rd Euromicro Conference on Software Engineering and Advanced Applications, SEAA, 2017, 440-448.

[3] A.A. Khan and M. Shameem, "Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process", Journal of software: evolution and process, 32(10), 2020, pp. 1-26.

[4] L.E. Lwakatare, P. Kuvaja, and M. Oivo, "An exploratory study of DevOps: extending the dimensions of DevOps with practices", In Proceedings of the 11th International Conference on Software Engineering Advances, ICSEA, 2016, 91-99.

[5] D. Teixeira, R. Pereira, T.A. Henriques, M. Silva, and J. Faustino, "A Systematic Literature Review on DevOps Capabilities and Areas", International journal of human capital and information technology professionals, 11(3), 2020, pp. 1-22.

[6] L. Riungu-Kalliosaari, S. Mäkinen, L.E. Lwakatare, J. Tiihonen, and T. Männistö, "DevOps Adoption Benefits and Challenges in Practice: A Case Study", In Product-Focused Software Process Improvement, PROFES, 2016, 590-597.

[7] J. Humble and J. Molesky, "Why enterprises must adopt devops to enable continuous delivery", Cutter IT Journal, 8(24), 2011, pp. 6-12.

[8] B. Fitzgerald and K. Stol, "Continuous software engineering: A roadmap and agenda", The Journal of systems and software, 12(3), 2017, pp. 176-189.

[9] T. Savor, M. Douglas, M. Gentili, L. Williams, K. Beck, and M. Stumm, "Continuous deployment at Facebook and OANDA", In Proceedings of the 38th International Conference on software engineering companion, 2016, 21-30.

[10] J. Itkonen, R. Udd, C. Lassenius, and T. Lehtonen, "Perceived Benefits of Adopting Continuous Delivery Practices", In Proceedings of the 10th ACM/IEEE International Symposium on empirical software engineering and measurement, 2016, 1-6.

[11] P. Rodríguez, A. Haghighatkhah, L.E. Lwakatare, S. Teppola, T. Suomalainen, J. Eskeli, T. Karvonen, P. Kuvaja, J.M. Verner, and M. Oivo, "Continuous deployment of software intensive products and services: A systematic mapping study", The Journal of systems and software, 12(3), 2017, pp. 263-291.

[12] L.E. Lwakatare, T. Kilamo, T. Karvonen, T. Sauvola, V. Heikkilä, J. Itkonen, P. Kuvaja, T. Mikkonen, M. Oivo, and C. Lassenius, "DevOps in practice: A multiple case study of five companies", Information and Software Technology, 11(4), 2019, pp. 217-230.

[13] J. Roche, "Adopting DevOps practices in quality assurance", Communications of the ACM, 56(11), 2013, pp. 38-43.

[14] M. Leppänen, S. Mäkinen, M. Pagels, V. Eloranta, J. Itkonen, M.V. Mantyla and T. Mannisto, "The highways and country roads to continuous deployment", IEEE Software, 32(2), pp. 64-72.

[15] L. Chen, "Continuous Delivery: Overcoming adoption challenges", The Journal of systems and software, 12(8), 2017, pp. 72-86.

[16] L. Leite, C. Rocha, F. Kon, D. Milojicic, and P. Meirelles, "A Survey of DevOps Concepts and Challenges", ACM computing surveys, 52(6), 2019, pp. 1-35.

[17] B. de França, J. Jeronimo Helvio, and G. Travassos, "Characterizing DevOps by Hearing Multiple Voices", In Proceedings of the 30th Brazilian Symposium on software engineering, 2016, 53-62.

[18] F.M.A. Erich, C. Amrit, and M. Daneva, "A qualitative study of DevOps usage in practice", Journal of software: evolution and process, 29(6), 2017.

[19] M. Rowse and J. Cohen, "A Survey of DevOps in the South African Software Context", In Proceedings of the 54th Hawaii International Conference on System Sciences, HICSS, 2021, 6785-6794.

[20] M. Senapathi, J. Buchan, and H. Osman, "DevOps Capabilities, Practices, and Challenges: Insights from a Case Study", In Proceedings of the 22nd International

Conference on evaluation and assessment in software engineering, 2018, 57-67.
[21] S. Neely and S. Stolt, "Continuous Delivery? Easy! Just Change Everything (Well, Maybe It Is Not That Easy)", In Proceedings of Agile Conference, 2013, 121-128.
[22] A. Wiedemann, N. Forsgren, M. Wiesche, H. Gewald and H. Krcmar, "Research for practice: the DevOps phenomenon", Communications of the ACM, 62(8), pp. 44-49.
[23] S.L. Vargo and R.F. Lusch, "Evolving to a New Dominant Logic for Marketing", Journal of marketing, 68(1), 2004, pp. 1-17.
[24] S. Vargo and R. Lusch, "Institutions and axioms: an extension and update of service-dominant logic", Journal of the Academic Marketing Science, 44(1), 2016, pp. 5-23.
[25] S. Vargo and R. Lusch, "Service-dominant logic: continuing the evolution", Journal of the Academic Marketing Science, 36(1), 2008, pp. 1-10.
[26] S. Mäkinen, M. Leppänen, T. Kilamo, A. Mattila, E. Laukkanen, M. Pagels, and T. Männistö, "Improving the delivery cycle: A multiple-case study of the toolchains in Finnish software intensive enterprises", Information and software technology, 8, 2016, pp. 175-194.
[27] Humble, J. and D. Farley, Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Addison-Wesley Professional, Upper Saddle River, NJ, 2010.
[28] K. Beck, M. Beedle, A. van Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, S. Mellor, K. Schwaber and J. Sutherland, "Manifesto for Agile Software Development", 2001, Accessed Jun 11, 2021, https://agilemanifesto.org.
[29] Poppendieck, T. and M. Poppendieck, Implementing Lean Software Development: From Concept to Cash, Addison-Wesley Professional, 2006.
[30] G.G. Claps, R. Berntsson Svensson, and A. Aurum, "On the journey to continuous deployment: Technical and social challenges along the way", Information and software technology, 57(1), 2015, pp. 21-31.
[31] S.L. Vargo, K. Koskela-Huotari, and J. Vink, "Service-Dominant Logic: Foundations and Applications", In E. Bridges and K. Fowler (eds.), The Routledge Handbook of Service Research Insights and Ideas. Routledge, 2020, 3-23.
[32] S.L. Vargo and R.F. Lusch, "Service-dominant logic 2025", International journal of research in marketing, 34(1), 2017, pp. 46-67.
[33] X. Wang, Y.D. Wong, C. Teo, and K.F. Yuen, "A critical review on value co-creation: towards a contingency framework and research agenda", Journal of service theory and practice, 29(2), 2019, pp. 165-188.
[34] N. Barqawi, K. Syed, and L. Mathiassen, "Applying service-dominant logic to recurrent release of software: an action research study", The Journal of business & industrial marketing, 31(7), 2016, pp. 928-940.
[35] C. Okoli, "A Guide to Conducting a Standalone Systematic Literature Review", Communications of the Association for Information Systems, 37(43), 2015, pp. 879 – 910.
[36] M.A. Akbar, W. Naveed, S. Mahmood, A.A. Alsanad, A. Alsanad, A. Gumaei, and A. Mateen, "Prioritization Based Taxonomy of DevOps Challenges Using Fuzzy AHP Analysis", IEEE Access, 8, 2020, pp. 202487-202507.
[37] L. Chen, "Continuous Delivery: Huge Benefits, but Challenges Too", IEEE Software, 32(2), pp. 50-54.
[38] B.G. Ghantous and A. Gill, "DevOps: Concepts, Practices, Tools, Benefits and Challenges", In Proceedings of the 21st Pacific Asia Conference on Information Systems, PACIS, 2017.
[39] P. Perera, M. Bandara, and I. Perera, "Evaluating the impact of DevOps practice in Sri Lankan software development organizations", In Proceedings of the International Conference on Advances in ICT for Emerging Regions, ICTer, 2016, 281-287.
[40] A. Hemon-Hildgen, F. Rowe, and L. Monnier-Senicourt, "Orchestrating automation and sharing in DevOps teams: a revelatory case of job satisfaction factors, risk and work conditions", European journal of information systems, 29(5), 2020, pp. 474-499.
[41] M. Shahin, M. Zahedi, M. Babar, and L. Zhu, "Adopting Continuous Delivery and Deployment: Impacts on Team Structures, Collaboration and Responsibilitie", In Proceedings of the 21st International Conference on evaluation and assessment in software engineering, 2017, 384-393.
[42] K. Nybom, J. Smeds and I. Porres, "On the Impact of Mixing Responsibilities Between Devs and Ops", In Proceedings of the 17th International Conference on Agile Processes in Software Engineering, and Extreme Programming (XP), 2016, 131–143.
[43] M.A. Akbar, S. Mahmood, M. Shafiq, A. Alsanad, A.A. Alsanad, and A. Gumaei, "Identification and prioritization of DevOps success factors using fuzzy-AHP approach", Soft computing, 2020.
[44] L.E. Lwakatare, T. Karvonen, T. Sauvola, P. Kuvaja, H.H. Olsson, J. Bosch, and M. Oivo, "Towards DevOps in the Embedded Systems Domain: Why is It So Hard?", In Proceedings of the 49th Hawaii International Conference on System Sciences, HICSS, 2016, 5437-5446.
[45] T. Tuunanen, E. Kazan, M. Salo, R. Leskelä, and S. Gupta, "From digitalization to cybernization: Delivering value with cybernized services", Scandinavian Journal of Information Systems, 13(2), 2019, pp. 83-96.
[46] C. Peters, P. Maglio, R. Badinelli, R.R. Harmon, R. Maull, J.C. Spohrer, T. Tuunanen, S.L. Vargo, J.J. Welser, H. Demirkan, T.L. Griffith, and Y. Moghaddam, "Emerging Digital Frontiers for Service Innovation", Communications of the Association for Information Systems, 39, 2016, pp. 136-149.