

# Validation and Clarification of Critical Success Factors of DevOps Processes

Michiel van Belzen<sup>a</sup>, Jos Trienekens and Rob Kusters<sup>b</sup>  
*Faculty of Science, Open University, Heerlen, The Netherlands*

**Keywords:** CI/CD-processes, Continuous Delivery, Continuous Deployment, Continuous Integration, Critical Success Factors, DevOps, Validation.

**Abstract:** Context: Critical Success Factors (CSFs) may contribute to solve challenges regarding Continuous Integration, Continuous Delivery and Continuous Deployment processes (CI/CD-processes). Prior research found some CSFs related to CI/CD and aspects of DevOps, but they are limited regarding validation, clarification and comprehensiveness. Objective: This study aims to contribute to the success of CI/CD-processes by showing and clarifying which CSFs determine the success of CI/CD-processes. Method: A three-phase process was followed. In the first phase, we conducted a systematic literature review in which we identified 144 potential CSFs. In the second phase, we classified the CSFs found into nineteen potential CSFs. Finally, we conducted a multiple case study with the following objectives: (1) to find examples of application to show that the potential CSFs were recognized by experts in the field, (2) to use the examples to validate the potential CSFs and show how the CSFs could be operationalized, and (3) to clarify why the validated CSFs are important to the success of CI/CD-processes. Results: Our main contribution to theory is a validated and structured model of nineteen clarified CSFs of CI/CD-processes, which were understood, recognized and grounded in practice by examples and clarifications on the importance of CSFs. Conclusions: Presenting a comprehensive model of CSFs, it appears that we achieved consensus regarding CSFs of CI/CD-processes in literature. In addition, IT-organizations could apply this model of CSFs to take steps towards successful results of CI/CD-processes.


## 1 INTRODUCTION


IT-organizations want to continuously provide new software products and software improvements to remain competitive. Continuous integration, continuous delivery and continuous deployment (CI/CD) is therefore the goal they strive for. CI/CD is a capability to ensure continuous value provisioning. With CI/CD they embrace business change, pursue economic efficiency, create business opportunities and focus on valuable product features provided in short cycles (Chen, 2017; Claps et al., 2015; Rodríguez et al., 2017).

To obtain CI/CD it is important to have processes in place (Alahyari et al., 2017; Shahin et al., 2017). Thus, IT-organizations invest in processes of Continuous Integration, Continuous Delivery or Continuous Deployment. These processes are mentioned CI/CD-processes in this study (Rostami

Mazrae et al., 2023). CI/CD-processes provide software releases continuously by performing interconnecting steps to achieve the goal of continuous value provision.

Various disciplines are needed working closely together to perform the activities of CI/CD-processes. For example, operations personnel are needed for deployments of product features. Therefore, an often-mentioned collaboration in relation to CI/CD is the collaboration between developers and operations personnel. This collaboration is called DevOps, a compound of development and operations (Sebastian et al., 2017). Through this close collaboration CI/CD becomes possible. It also enables a short feedback loop in CI/CD-processes. For example, this feedback loop ensures that events in a production environment will soon be known to developers. However, prior research reports barriers and problems that hinder success of CI/CD-processes. For example, unclear

<sup>a</sup>  <https://orcid.org/0000-0002-5068-4442>

<sup>b</sup>  <https://orcid.org/0000-0003-4069-5655>

definition of delivery, late scope changes and dependencies with other teams (Alahyari et al., 2017).

In an attempt to address problems that hinder success of CI/CD-processes, prior research has been conducted on factors that determine the success of CI/CD-processes (Azad & Hyrynsalmi, 2023; Ramzan et al., 2023). However, we did not find a comprehensive overview of validated Critical Success Factors (CSFs) of CI/CD-processes in literature.

With this study we aim to contribute to the success of CI/CD-processes by showing and clarifying which CSFs determine the success of CI/CD-processes.

Our main research question is: What are CSFs of CI/CD-processes? The sub questions are: (SQ1) What are CSFs of CI/CD-processes? (SQ2) How are these CSFs addressed in a DevOps context? (SQ3) Why are these CSFs important to the success of CI/CD-processes?

This study is relevant because more rigorous research and case studies on CI/CD is needed (Azad & Hyrynsalmi, 2023; Rodríguez et al., 2017).

In Section 2, we describe prior research on the concepts of Continuous Integration, Continuous Delivery and Continuous Deployment. In Section 3, we describe our methodology consisting of a systematic literature review, a classification and a multiple case study to validate and clarify the CSFs found and to show how the CSFs could be operationalized. We present the results in Section 4. In Section 5, we discuss the implications of the results, the limitations of our study and present some options for further research. Finally, Section 6 contains the conclusions of our research.

## 2 THEORETICAL BACKGROUND

Prior studies have attempted to find CSFs of Continuous Integration (CI), Continuous Delivery (CDE), Continuous Deployment (CD) and CSFs of DevOps. Some studies conducted systematic literature reviews to create overviews of CSFs found. However, the results differ, since we found different lists of CSFs that were limited in terms of validation. Thus, there seems to be no consensus yet. In this section, we review related research and justify the need for a comprehensive overview of validated CSFs of CI/CD-processes.

To address the high rate of business changes and to focus on valuable product features provided in short cycles, organizations started to embrace

software development agility (Lee & Xia, 2010). With this capability, software development teams are able to effectively and efficiently respond to requirement changes (Lee & Xia, 2010). To move from cyclic to continuous value provision, organizations evolved software development agility to CI/CD-processes (Rodríguez et al., 2017). CI/CD-processes are typically executed at team level (Rodríguez et al., 2017), and several DevOps teams can be involved. A DevOps team may choose to conduct CDE or CD in which CI is integrated in both cases. We are aware that by using the term CI/CD we do not distinguish between CDE and CD in this study. We do not consider this to be a problem, since our study focus on CSFs of all these processes. Some researchers called the combination of these three concepts simply continuous practices as presented in Portela & de França (2023). However, it should be noted that the concept of CI/CD is also used for the combination of CI and CDE as presented in Laukkanen et al. (2017). Each CI/CD-process is a standardized way of work performing the same activities in the same order each time to provide software releases and achieve the goal of continuous value provision. CI/CD-processes are expected to minimize the time between a changed requirement and its release in production (Chen, 2017). Therefore, the CI-process is an inherent part of the processes of CDE and CD through which software releases are built, tested and deployed in production (Fitzgerald & Stol, 2017; Laukkanen et al., 2017; Shahin et al., 2017).

CI is defined as a process comprising steps such as, compiling code, running unit and acceptance tests, validating code coverage, checking compliance with coding standards, and building deployment packages (Fitzgerald & Stol, 2017). The CI-process is usually automatically triggered and highly automated (Fitzgerald & Stol, 2017; Ståhl & Bosch, 2014a). Multiple developers are required to integrate their work frequently to a common code repository, enabling the system to be built and tested. This ensures fast feedback to developers and quick problem solving (Fowler & Foemmel, 2006; Laukkanen et al., 2017). This feedback loop increases confidence in the source code as it progresses through the system (Ståhl & Bosch, 2014a). However, literature mentioned several issues which threaten the continuity of the CI-process. Ståhl & Bosch (2014b) found differences in CI-process implementations. Humble & Farley (2010) and Ståhl et al. (2017) mentioned lower continuity in the case of large software size, big software modules, large organization size and relatively few developers in the

organization. In addition, Ståhl et al. (2017) found that individual developers commit not often as expected and that larger organizations may be unable or unwilling to directly integrate with the mainline. Thus, contextual factors may impact the continuity of CI/CD-processes. Other researcher report challenges on tool support (Debroy et al., 2018), test automation and infrastructure (Azad & Hyrynsalmi, 2023), continuous monitoring and team dynamics (Ramzan et al., 2023) and security (Portela & de França, 2023).

We define CDE as a process comprising steps such as, continuous integration, tests and manual deployment of a release to production (Laukkanen et al., 2017). Therefore, the release is kept in a certain state until a human decides to deploy to production (Chen, 2017; Humble & Farley, 2010; Laukkanen et al., 2017). This decision-maker is often a representative of the customer. CDE requires continuous integration to obtain the builds (Shahin et al., 2017). It should be noted that CDE is defined differently by some researchers. For example, Chen (2017) considers CDE as a software engineering approach. Fitzgerald & Stol (2017) and Humble (2018), however, consider CDE as a capability which is composed of principles, patterns and practices to enable reliable deployments to an environment anytime. Laukkanen et al. (2017) found factors that negatively impact CDE such as, system design problems, resource problems and organizational problems. In line with these results, Humble (2018) states inadequate architecture and a nongenerative culture. Caprarello et al. (2020) found time-waste due to high workload of operations personnel. Pereira et al. (2021) reported culture challenges and technical challenges such as, appropriate technologies.

We define CD as a process comprising steps such as, continuous integration, tests and the automated deployment of a release to production (Claps et al., 2015; Humble & Farley, 2010; Rodríguez et al., 2017; Shahin et al., 2017). In contrast to CDE, CD is an automated process (Rodríguez et al., 2017). Therefore, CD gains feedback much faster reducing costs and improving quality. Due to the similarities between CDE and CD and the application of automation in the CD-process, CDE is considered to enable CD (Humble, 2018). However, it should be noted that some researchers do not distinguish between CDE and CD as presented in Claps et al. (2015). Some researchers report limitations and challenges on CD. For example, contrary to CDE which can be applied in any domain, CD is typically limited to cloud services or datacenter hosted services (Humble, 2018). Furthermore, CD requires parallelization of the process (Rodríguez et al., 2017).

Finally, Saeeda et al. (2023) reports quality challenges such as, ignored coding standards to rush deployment and unstructured code.

In this study we define DevOps as a collaboration between developers and operations personnel working together as a team and executing CI/CD-processes. This is in line with prior research which consider DevOps as a prerequisite for CI/CD-processes. For example, Fitzgerald & Stol (2017) emphasized that continuous integration requires a collaboration between development and operations. Furthermore, Chen (2017) present a dedicated multi-disciplinary team as a strategy to overcome adoption challenges of CDE.

We define a CSF as a factor leading to successful results, which is in line with Ram et al. (2013). We searched for a comprehensive overview of validated CSFs of CI/CD-processes in the context of DevOps and we found some lists of CSFs. For example, Dwi Harfianto et al. (2022) found 20 challenges by a literature review, yet did not validate the factors found. Saeeda et al. (2023) conducted an exploratory case study and identified thirteen challenges in large-scale agile in two teams. Portela & de França (2023) conducted a rapid literature review and found 121 challenges. However, they were limited to technical challenges and they did not validate them. Ramzan et al. (2023) found 40 success factors related to DevOps and cloud by means of a systematic literature review, but did not validate the factors. Azad & Hyrynsalmi (2023) reviewed literature and found ten CSFs of DevOps including CI/CD. However, they did not validate the CSFs. Finally, we published the first batch of 90 potential CSFs found by a literature review in Van Belzen et al. (2019). Yet, these potential CSFs were not validated.

In summary, the lists of CSFs found differ from each other and there seems to be no consensus. Furthermore, the CSFs are limited validated and analyzed yet. Thus, we still need a comprehensive overview of validated CSFs of CI/CD-processes build on prior research.

### 3 RESEARCH METHODOLOGY

To address our research goal, we followed a three-phase process. In phase 1, we conducted a Systematic Literature Review (SLR) to obtain a comprehensive list of CSFs of CI/CD-processes. However, we could not find a comprehensive list of CSFs. Yet, we found a lot of different potential CSFs of CI/CD. Therefore, we classified the potential CSFs found in phase 2 by means of a metaplan (Schnelle, 1979) session. This

resulted in a list of potential CSFs manageable for phase 3. In phase 3, we conducted a multiple case study to validate the potential CSFs classified, to show how the CSFs could be operationalized and to clarify why the validated CSFs are important to the success of CI/CD-processes.

### 3.1 Systematic Literature Review

We searched in five digital libraries following the search process according to Kitchenham et al. (2010). We used these libraries suggested by Kitchenham & Brereton (2013): IEEE computer society digital library, ACM digital library, SpringerLink, Web of Science and SCOPUS as they contain good quality papers on the topic of our study. The search queries were based on ‘All fields’ (ACM, SpringerLink and Web of Science), ‘abstract’ (IEEE), ‘Title, abstract and keywords’ (SCOPUS). We started searching from 2001 because at that moment the agile manifesto emerged. We used the following search string for every library: (“Continuous Delivery” OR “Continuous Deployment” OR “Continuous Integration”) AND DevOps. We did not add the key words “Critical Success Factor” or “Factor” to the search string, because this resulted in fewer relevant papers. After deduplication and removal of obviously irrelevant results or papers not in the English language, we rejected papers based on screening of title and abstract on the basis that they did not include CSFs. Next, we read the full texts to find CSFs. In the cases we found CSFs, we extracted the name of the CSF and a citation of the corresponding description from the papers.

### 3.2 Classification of CSFs Found in Literature

The systematic literature review resulted in a lot of potential CSFs with different abstraction levels and overlap, homonyms and synonyms. Thus, we had to classify the potential CSFs. The classification also resulted in an orderly list, which made the potential CSFs manageable during the multiple case study. We classified the potential CSFs found based on the metaplan-method of Schnelle (1979), which is a form of open card sorting (Lewis & Hepburn, 2010). Therefore, we made a card of each potential CSF and a pile of all cards in advance. Subsequently, we sort the cards into piles based on the descriptions found in literature in order to obtain groups of similar potential CSFs. Throughout card sorting we discussed the grouping of each card and the name of each emerged potential CSF. Next, we named each pile of emerged

potential CSF. Subsequently, we derived the descriptions of the emerged potential CSFs from the grouped potential CSFs using the descriptions found in literature. This was also a means to verify the quality of our work.

Although we found potential CSFs, they were not validated in a consistent way. Furthermore, we did not know why the CSFs are important to the success of CI/CD-processes. Therefore, we conducted a case study in the subsequent phase of our research methodology.

### 3.3 Multiple Case Study

A multiple case study was chosen as a relevant method, because it enables us to obtain depth and explanation on social phenomena (Verschuren & Doorewaard, 2010; Yin, 2018). That was important as we had to validate the emerged potential CSFs and to explain why these potential CSFs are important to the success of CI/CD-processes. The multiple case study enabled us (1) to find examples of application to show that the potential CSFs were recognized by experts in the field, (2) to use the examples to validate the potential CSFs and to show how the CSFs could be operationalized, and (3) to clarify why the validated CSFs are important to the success of CI/CD-processes.

To prepare the multiple case study we followed these five steps proposed by Yin (2018): (1) designing the case study, (2) preparing to collect evidence, (3) collecting evidence, (4) analyzing evidence and (5) reporting results.

In the first step, we designed the multiple case study in order to assure the evidence to be collected corresponds with the research question (Yin, 2018). Thus, we chose an inductive approach and carried out cross-sectional semi-structured interviews to validate the potential CSFs. This approach enabled us to achieve depth, elaboration, clarification and improvisation (Runeson & Höst, 2009; Verschuren & Doorewaard, 2010). Furthermore, it allowed us to address real-life experiences.

In the second step, we prepared to collect the evidence needed to answer our research questions. Therefore, we defined criteria to select experienced organizations and interviewees and to obtain real-life examples elaborating and validating the potential CSFs. After that, we searched for organizations willing to participate. We used the following criteria to select case organizations: (1) provide IT services including software development, (2) has preferably an organization size of more than 1000 employees, (3) established DevOps teams which consists of

software developers and operation personnel, (4) has implemented CI, CDE and/or CD at least 2 - 3 years ago, (5) CI, CDE and/or CD process steps, role distribution and responsibilities could be clearly explained by representatives of the organization, (6) has established shared goal(s) concerning CI, CDE and/or CD, and (7) could provide at least six interviewees willing to participate. We contacted gatekeepers of appropriate organizations and verified whether they were willing to participate. Therefore, we explained our study, the criteria to select organizations and interviewees and compiled a list of potential interviewees. To select interviewees, we used the following criteria: (1) member of a DevOps team, (2) at least seven years of work experience, (3) at least five years of work experience in current role inside the case organization, (4) at least three years of relevant work experience concerning DevOps and CI/CD, (5) at least two years of relevant work experience concerning DevOps and CI/CD inside the case organization, (6) is willing to participate. We used the list of potential interviewees to contact them. We explained our study and asked whether they are willing to participate.

To guide the interviews and to assure reliability, we developed an interview protocol. The protocol described the steps to take from invitation up to coding the transcriptions and had an invitation letter, letter of consent, list of definitions of our research concepts, the list of potential CSFs (table 1) and the questionnaire attached. We sent the invitation letter together with the definitions, potential CSFs, letter of consent and questionnaire to the interviewees willing to participate. We did this in order to enable the participants to prepare for the interview, which saved us time during the interviews. The questionnaire contained the following interview questions: (1) Do you understand the description of this CSF? (2) Have you ever experienced this CSF and can you give an example? (3) Can you indicate on a scale of 1 to 5 what the degree of importance is of this CSF? (4) Why is this CSF important? Question one was asked to verify whether the interviewee had read and comprehended the corresponding CSF. This was important to answer question two. Question two validated the potential CSFs by appealing to their expert knowledge and experience. We asked question three to stimulate an answer on question four. Therefore, we used a five-level scale with the following values: (1) Not important, (2) Somewhat important, (3) Reasonable important, (4) Important, (5) Very important. The answer on question four clarified the importance of a particular CSF. We asked in-depth questions when appropriate. To test

the interview protocol, we conducted a pilot interview. We defined codes for recognized potential CSFs and corresponding examples, the degree of importance and the corresponding clarifications.

In the third step, we collect the evidence according to our interview protocol. We conducted the interviews accordingly, took notes as appropriate, recorded each interview and transcribed them upon completion. We gave the interviewees the opportunity to amend the transcriptions.

In the fourth step, we analyzed the evidence to achieve two objectives. First, we had to verify the validation of the potential CSFs. Second, we had to verify the clarification. To obtain the first objective, we first determined that a potential CSF is validated when we found at least one example. To find examples per CSF, we applied open coding (Saldaña, 2013). Therefore, we defined a code per CSF, familiarized myself with the content of the transcriptions, selected relevant text and coded examples mentioned using ATLAS.ti (<https://atlasti.com/>). We did not classify the examples. To obtain the second objective, we applied open coding and subsequently axial coding to analyze the clarifications on the importance per CSF (Saldaña, 2013). To apply open coding, we defined an additional code per CSF, familiarized myself with the content of the transcriptions, selected relevant text and coded clarifications mentioned using ATLAS.ti. Next, we conducted axial coding on the coded clarifications mentioned using a metaplan session to prevent the bias of an individual. During the metaplan session, we classified all similar clarifications found into types of clarifications. Therefore, we made a card of each clarification and a pile of all cards in advance. Subsequently, we sorted the cards into piles based on the citation of the clarification in order to obtain groups of similar clarifications. Throughout card sorting we discussed the grouping of each card and the name of each emerged type of clarification. Next, we named each pile of emerged type of clarification. Subsequently, we derived the descriptions of the emerged types of clarifications from the grouped clarifications using their citations. This was also a means to verify the quality of our work.

In the fifth and last step, we made a table in which we put the name, definition and references of the validated CSFs, the types of clarifications, the description of each type and the corresponding specific clarifications mentioned during the interviews, and the corresponding examples mentioned.

## 4 RESULTS OF THE SLR, CLASSIFICATION AND MULTIPLE CASE STUDY

After conducting the SLR, we found potential CSFs in literature. The CSFs were subsequently classified into potential CSFs of CI/CD-processes. We validated these CSFs based on real-life examples mentioned during a multiple case study. Furthermore, we clarified the CSFs through classification of the clarifications mentioned by the interviewees.

### 4.1 Results of the Systematic Literature Review

We applied the search strings according to our systematic literature review approach and we found 2011 papers in total. After deduplication and removal of obviously irrelevant results or papers not in the English language a total of 1476 papers remained. Next, we rejected papers based on screening of title and abstract on the basis that they did not include potential CSFs. We also rejected papers due to limited access to library records as we were limited to the subscription of our institution. This resulted in 29 remaining papers. After reading the full texts, we found 19 papers containing potential CSFs. Furthermore, we added two additional and relevant papers of which we were aware: Yaman et al. (2016) and Laukkanen et al. (2017).

We extracted the following data from the 21 papers found: name of the potential CSF and a citation of the corresponding description. However, we found that some potential CSFs were missing a description. Therefore, we extracted explanations or examples mentioned. After data extraction, we removed duplicate potential CSFs and condensed the description of the remaining 144 potential CSFs.

### 4.2 Results of the Classification

During preparation of the metaplan-session, we used descriptions, explanations or examples of potential CSFs to create cards. Next, we conducted card sorting according to our methodology. We found that this went well, because we had a lot of data on potential CSFs extracted from literature. For the new emerged CSF preconditions, we chose to adopt a description from Smyth (2018) since it corresponded with a variety of potential CSFs found in literature.

The classification resulted in nineteen classified potential CSFs and corresponding descriptions. This answered the first sub research question. Each potential

CSF mentioned was provided with a name and initial description, both of which may be improved based on new insights after validation during the multiple case study in the next phase of our research method.

### 4.3 Results of the Multiple Case Study

After conducting the classification, we started to collect the evidence. This means that we searched for organizations and corresponding interviewees willing to participate, tested the interview protocol prepared, and conducted the remaining steps according to the multiple case study approach.

We found two large IT service providers producing software willing to participate meeting our selection criterion 1. We discussed our remaining criteria with the gatekeepers of both organizations. Both organizations provide IT services including software development. Case organization 1 (CO1) has approximately 3000 employees and 45 DevOps teams, and case organization 2 (CO2) has approximately 1800 employees and 28 DevOps teams. CO1 implemented CI years ago and prior to CDE. CD to production was deliberately not implemented for security reasons and because there is no business need. CO2 implemented CI/CD, but it was not known exactly when. They implemented DevOps 6-7 years ago. Process steps, role distribution and responsibilities could be clearly explained by representatives of CO1. CO2 had defined roles, goals and performance requirements.

In cooperation with the gatekeepers, we found six interviewees willing to participate in each organization. We contacted the interviewees, explained our study, verified the extent to which they meet the selection criteria and asked whether they were actually able and willing to participate. In CO1 we found four developers, one architect and one software engineer, whom were members of a DevOps team. In CO2 we found two developers, one tester, one information engineer, one database administrator (DBA) and one integration specialist, whom also were members of a DevOps team. All interviewees had at least seven years of work experience. Five interviewees of CO1 had at least seven years of work experience in their current role. One interviewee had four years of work experience in his current role. All interviewees of CO1 had at least three years of experience with DevOps and CI/CD and four interviewees had even more than five years of experience. All interviewees of CO1 had at least three years of experience concerning DevOps and CI/CD inside the case organization. All interviewees of CO2 had at least five years of work experience in their

current role and they had at least three years of experience with DevOps and CI/CD. All interviewees had at least two years of experience concerning DevOps and CI/CD inside the case organization. Interviewee 2 (I2) was a member of a so-called continuous delivery team in CO1. This team supported other DevOps teams by providing and managing the software development life cycle.

We tested our interview protocol in both organizations and found no alterations were necessary. Next, we started to interview the selected interviewees in both organizations. We used video conferencing tools to conduct and record the interviews. The interviews took approximately 45-90 minutes. After the interviews we transcribed the recordings and asked the interviewees to amend the transcriptions if appropriate. Finally, we anonymized the transcriptions.

After conducting open coding, we analyzed the data. In both case organizations all interviewees understand the description of each potential CSF. We also found that all nineteen potential CSFs could be elaborated by at least one real-life example. Thus, all nineteen CSFs were validated and no changes regarding name or description of the CSFs were required. We present the nineteen CSFs of CI/CD-processes in table 1.

Table 1: CSFs of CI/CD-processes.

Nr	Potential CSF	Description
1	Preconditions	Establishing the optimal provision of value (e.g., generating new capabilities, supporting routines and competencies, restructuring) for realization in use and context where standardization and routinization do not currently exist (Smyth, 2018). In other words, affairs which are not under direct control of CI/CD-processes.
2	Goals	Clear goals for the teams migrating towards CI/CD-processes and assimilation metrics.
3	Strategy and approach	Approaches to drive CI/CD-processes assimilation, and branching strategies.
4	Architecture	Diverse aspects on architecture of the product and related infrastructure.
5	Process design	Institutionalizing CI/CD-processes and aspects of the process e.g., design, effort to initially setting up the process, management, planning, sufficient time/resources, waste in the process and accuracy of the process.

Nr	Potential CSF	Description
6	Motivation	Motivation to adopt CI/CD-processes and to get past early difficulties and effort, and discipline to commit often, test diligently, monitor the build status and fix problems as a team.
7	Resistance to change	Difficulty to change established organizational policies and cultures.
8	Complexity across customer organization boundary	No access to or control on a production environment or diversity and complexity of customer sites, which make it harder to fully automate CI/CD-processes.
9	Acceptance by customer	Adopting the practice of continuous releases. Customer perception of their involvement in development and customer behaviour. Domain constraints. Feature discovery.
10	Sales and intermediaries	When user data is not accessible due to intermediaries.
11	Quality	Preserving quality and adequate documentation.
12	Customer involvement	Preparing and receiving customer input, establishing a customer sample group, and providing feature growth.
13	Test complexity & source code control	Aspects on automating, configuring and using test environments, and source code control. Due to higher demands as compared to traditional (not agile) ways of work.
14	Coordination	Increased need for coordination between team members and multiple teams. Organizational structure. Co-locate by feature not discipline.
15	Communication	Intra and inter team communication, the right communication tools, awareness and transparency.
16	Knowledge and training	Sufficient proficiency, knowledge, skills and experience.
17	Tooling	Maturity of the tools and their surrounding infrastructure that sufficiently support CI/CD-processes, including security, access controls and consensus among users about the choice of tools. Heterogeneous programming languages, operating systems and communication tools.
18	Pace	Improving the speed of providing deployments to the customer.
19	Pressure	Increased pressure on the team to have code ready to be deployed immediately, and too much transparency which causes customers to interfere with developers' work.

Some examples mentioned are: the implementation of release automation (example of CSF Precondition), a gradual change to CI/CD (example of CSF Strategy and approach) and architecture was used to explain why a register was needed and to make clear what the boundaries are (example of CSF Architecture). The CSF sales and intermediaries was considered relevant by only two interviewees and appear to be dependent on the local context. The interviewees were able to explain why each potential CSF is important to the success of CI/CD. We classified all similar clarifications found into 74 types of clarifications. For example, CSF Preconditions contains three types: (1) Efficiency, described as ‘To ensure the production of desired results without waste such as, incurring costs, frustration and unavailability’, (2) Dependencies, described as ‘Others deliver services on which the team depends. Vendor lock-in’, (3) Legacy, described as ‘Addressing preconditions is more important when dealing with legacy systems instead of systems build on modern technologies. Every migration to a modern technology decreases the importance of preconditions and dependencies’. Next, we made a table of CSFs, corresponding definitions and references, the classified clarifications (type of clarification) and corresponding clarifications on the importance mentioned by the interviewees.

## 5 DISCUSSION AND TECHNOLOGICAL IMPLICATIONS

Considering the results of our study, we are able to answer our research question and discuss their implications.

The results show a structured model of nineteen validated CSFs of CI/CD-processes. Therefore, it answers our main research question.

The results have several theoretical implications. First, it appears that we achieved consensus regarding CSFs of CI/CD-processes since the resulting model of CSFs is more comprehensive compared to previous literature. Furthermore, it includes CSFs found previously based on an extensive and structured literature review. For example, Azad & Hyrynsalmi (2023) found technical factors, organizational factors, and social and cultural factors which are included in our model by several CSFs. To illustrate the overlap, they found integration, build and test automation, and infrastructure as factors which were mainly included in our model by the CSFs process design and tooling. A second example is the study of Harfianto et al. (2022), who found the following factors: project documentation, internal policy, cultural behavior, top

level management and IT infrastructure. We included these factors in our model by the CSFs quality, resistance to change and tooling. Second, all CSFs on CI/CD found in literature are valid for CI/CD-processes. This confirms existing literature on CSFs of CI/CD used in our study. Third, our study clarified why these CSFs are relevant based on clarifications of experts in the field of CI/CD and DevOps.

The results also have several practical implications. First, our model shows which CSFs effectively lead to the success of CI/CD-processes. Second, it clarifies why these CSFs are relevant. Third, the real-life examples show how the CSFs could be operationalized. Thus, organizations may take advantage of these clarifications and examples and could apply this model of CSFs to take steps towards a successful CI/CD-process.

We have strived to make the model of CSFs as complete as possible by using a well-defined methodology to address study selection validity threat (Ampatzoglou et al., 2019). However, we can not ensure the completeness of the model of CSFs. Another concern on internal validity is that there may be a bias in selecting interviewees in cooperation with the gatekeepers. There may also be accidental coding errors and processing errors. However, we applied a well-defined methodology to minimize these threats (Verschuren & Doorewaard, 2010).

The results of this study may not just be generalized to other organizations, because the extent to which the CSFs are relevant depends on the local context. However, the results are based on two case organizations and therefore appear to be more broadly applicable.

Finally, we mention some opportunities for future research. First, we could not discuss all aspects of each CSF during the interviews due to the comprehensiveness of the model of CSFs. Therefore, future research could focus on interrelations between CSFs or a specific dimension of CI/CD-processes and corresponding aspects of relevant CSFs. Furthermore, the model of CSFs and descriptions could be used to further develop consistent and uniform definitions of CSFs based on the examples and clarifications presented. This could ease the application of the CSFs and enables research on measuring the impact of the CSFs.

## 6 CONCLUSIONS

CI/CD enables IT-organizations to continuously provide new software products and software improvements to remain competitive. To obtain



CI/CD they invest in CI/CD-processes. To address challenges, barriers and problems that hinder success of CI/CD-processes, this study presents a comprehensive overview of CSFs based on literature and real-life examples which validated them. Furthermore, we clarified the CSFs through classification of the clarifications mentioned by the experts in the field.

IT-organizations could apply this model of CSFs to take steps towards successful results of CI/CD-processes. Therefore, they may take advantage of the clarifications on the importance of CSFs and the real-life examples which elaborate the CSFs.

All data emerged during the research process are available on request.

## REFERENCES

- Alahyari, H., Berntsson Svensson, R., & Gorschek, T. (2017). A study of value in agile software development organizations. *Journal of Systems and Software*, 125, 271–288. <https://doi.org/10.1016/j.jss.2016.12.007>
- Ampatzoglou, A., Bibi, S., Avgeriou, P., Verbeek, M., & Chatzigeorgiou, A. (2019). Identifying, categorizing and mitigating threats to validity in software engineering secondary studies. *Information and Software Technology*, 106, 201–230. <https://doi.org/10.1016/j.infsof.2018.10.006>
- Azad, N., & Hyrinsalmi, S. (2023). DevOps critical success factors—A systematic literature review. *Information and Software Technology*, 107150. <https://doi.org/10.1016/j.infsof.2023.107150>
- Caprarelli, A., Di Nitto, E., & Tamburri, D. A. (2020). Fallacies and Pitfalls on the Road to DevOps: A Longitudinal Industrial Study. In J.-M. Bruel, M. Mazzara, & B. Meyer (Eds.), *Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment* (Vol. 12055, pp. 200–210). Springer International Publishing. [https://doi.org/10.1007/978-3-030-39306-9\\_15](https://doi.org/10.1007/978-3-030-39306-9_15)
- Chen, L. (2017). Continuous Delivery: Overcoming adoption challenges. *Journal of Systems and Software*, 128, 72–86. <https://doi.org/10.1016/j.jss.2017.02.013>
- Claps, G. G., Berntsson Svensson, R., & Aurum, A. (2015). On the journey to continuous deployment: Technical and social challenges along the way. *Information and Software Technology*, 57, 21–31. <https://doi.org/10.1016/j.infsof.2014.07.009>
- Debroy, V., Miller, S., & Brimble, L. (2018). Building lean continuous integration and delivery pipelines by applying DevOps principles: A case study at Varidesk. *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering - ESEC/FSE 2018*, 851–856. <https://doi.org/10.1145/3236024.3275528>
- Dwi Harfianto, H., Raharjo, T., Hardian, B., & Wahbi, A. (2022). Agile Transformation Challenges and Solutions in Bureaucratic Government: A Systematic Literature Review. *Proceedings of the 2022 5th International Conference on Computers in Management and Business*, 12–19. <https://doi.org/10.1145/3512676.3512679>
- Fitzgerald, B., & Stol, K.-J. (2017). Continuous software engineering: A roadmap and agenda. *Journal of Systems and Software*, 123, 176–189. <https://doi.org/10.1016/j.jss.2015.06.063>
- Fowler, M., & Foemmel, M. (2006). *Continuous integration*. Thought-Works. [http://www.thoughtworks.com/Continuous Integration. pdf](http://www.thoughtworks.com/Continuous%20Integration.pdf), 122, 14.
- Humble, J. (2018). Continuous delivery sounds great, but will it work here? *Communications of the ACM*, 61(4), 34–39. <https://doi.org/10.1145/3173553>
- Humble, J., & Farley, D. (2010). *Continuous delivery: Reliable software releases through build, test, and deployment automation*. Addison-Wesley.
- Kitchenham, B., & Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information and Software Technology*, 55(12), 2049–2075. <https://doi.org/10.1016/j.infsof.2013.07.010>
- Kitchenham, B., Pretorius, R., Budgen, D., Pearl Brereton, O., Turner, M., Niazi, M., & Linkman, S. (2010). Systematic literature reviews in software engineering – A tertiary study. *Information and Software Technology*, 52(8), 792–805. <https://doi.org/10.1016/j.infsof.2010.03.006>
- Laukkanen, E., Itkonen, J., & Lassenius, C. (2017). Problems, causes and solutions when adopting continuous delivery—A systematic literature review. *Information and Software Technology*, 82, 55–79. <https://doi.org/10.1016/j.infsof.2016.10.001>
- Lee, G., & Xia, W. (2010). Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data. *MIS Q.*, 34(1), 87–114.
- Lewis, K. M., & Hepburn, P. (2010). Open card sorting and factor analysis: A usability case study. *The Electronic Library*, 28(3), 401–416. <https://doi.org/10.1108/02640471011051981>
- Pereira, I. M., Carneiro, T. G. de S., & Figueiredo, E. (2021). Investigating Continuous Delivery on IoT Systems. *Proceedings of the XX Brazilian Symposium on Software Quality*. <https://doi.org/10.1145/3493244.3493261>
- Portela, A., & de França, B. B. N. (2023). Empirical Evidence on Technical Challenges When Adopting Continuous Practices. *Proceedings of the XXXVII Brazilian Symposium on Software Engineering*, 11–20. <https://doi.org/10.1145/3613372.3613390>
- Ram, J., Corkindale, D., & Wu, M.-L. (2013). Implementation critical success factors (CSFs) for ERP: Do they contribute to implementation success and post-implementation performance? *International Journal of Production Economics*, 144(1), 157–174. <https://doi.org/10.1016/j.ijpe.2013.01.032>

- Ramzan, S., Khan, S.-U.-R., Hussain, S., Wang, W.-L., & Tang, M.-H. (2023). Identification of Influential Factors for Successful Adoption of DevOps and Cloud. *Proceedings of the 27th International Conference on Evaluation and Assessment in Software Engineering*, 423–429. <https://doi.org/10.1145/3593434.3594239>
- Rodríguez, P., Haghhighatkah, A., Lwakatare, L. E., Teppola, S., Suomalainen, T., Eskeli, J., Karvonen, T., Kuvaja, P., Verner, J. M., & Oivo, M. (2017). Continuous deployment of software intensive products and services: A systematic mapping study. *Journal of Systems and Software*, 123, 263–291. <https://doi.org/10.1016/j.jss.2015.12.015>
- Rostami Mazrae, P., Mens, T., Golzadeh, M., & Decan, A. (2023). On the usage, co-usage and migration of CI/CD tools: A qualitative analysis. *Empirical Software Engineering*, 28(2), 52. <https://doi.org/10.1007/s10664-022-10285-5>
- Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, 14(2), 131–164. <https://doi.org/10.1007/s10664-008-9102-8>
- Saeeda, H., Ahmad, M. O., & Gustavsson, T. (2023). Identifying and Categorizing Challenges in Large-Scale Agile Software Development Projects: Insights from Two Swedish Companies. *SIGAPP Appl. Comput. Rev.*, 23(2), 23–43. <https://doi.org/10.1145/3610409.3610411>
- Saldaña, J. (2013). *The Coding Manual for Qualitative Researchers* (Second Edition). SAGE.
- Schnelle, E. (1979). *The Metaplan-Method communication tools for planning and learning groups*. Metaplan-GmbH.
- Sebastian, I. M., Ross, J. W., Beath, C., Mocker, M., Moloney, K. G., & Fonstad, N. O. (2017). How Big Old Companies Navigate Digital Transformation. *MIS Quarterly Executive*, 16(3), 197–213.
- Shahin, M., Ali Babar, M., & Zhu, L. (2017). Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices. *IEEE Access*, 5, 3909–3943. <https://doi.org/10.1109/ACCESS.2017.2685629>
- Smyth, H. (2018). Projects as creators of the preconditions for standardized and routinized operations in use. *International Journal of Project Management*, 36(8), 1082–1095. <https://doi.org/10.1016/j.ijproman.2018.08.004>
- Ståhl, D., & Bosch, J. (2014a). Automated software integration flows in industry: A multiple-case study. *Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014*, 54–63. <https://doi.org/10.1145/2591062.2591186>
- Ståhl, D., & Bosch, J. (2014b). Modeling continuous integration practice differences in industry software development. *Journal of Systems and Software*, 87, 48–59. <https://doi.org/10.1016/j.jss.2013.08.032>
- Ståhl, D., Mårtensson, T., & Bosch, J. (2017). The continuity of continuous integration: Correlations and consequences. *Journal of Systems and Software*, 127, 150–167. <https://doi.org/10.1016/j.jss.2017.02.003>
- Van Belzen, M., Trienekens, J., & Kusters, R. (2019, August 28). Critical Success Factors of Continuous Practices in a DevOps Context. *Information Systems Development: Information Systems Beyond 2020 (ISD2019 Proceedings)*. Information Systems Development: Information Systems Beyond 2020, Toulon, France: ISEN Yncréa Méditerranée.
- Verschuren, P., & Doorewaard, H. (2010). *Designing a Research Project* (2nd edition). Eleven International Publishing.
- Yaman, S. G., Sauvola, T., Riungu-Kalliosaari, L., Hokkanen, L., Kuvaja, P., Oivo, M., & Männistö, T. (2016). Customer Involvement in Continuous Deployment: A Systematic Literature Review. *Requirements Engineering: Foundation for Software Quality*, 9619, 249–265. [https://doi.org/10.1007/978-3-319-30282-9\\_18](https://doi.org/10.1007/978-3-319-30282-9_18)
- Yin, R. K. (2018). *Case study research and applications: Design and methods* (Sixth edition). SAGE.