Received: 6 May 2015 | Revised: 30 April 2017 | Accepted: 12 May 2017

DOI: 10.1002/smr.1885

WILEY Journal of Software: Evolution and Process

SPECIAL ISSUE PAPER

A qualitative study of DevOps usage in practice

F. M. A. Erich[1] | C. Amrit[2] | M. Daneva[3]

[1]School of Integrative and Global Majors, University of Tsukuba
[2]School of Management and Governance, University of Twente
[3]Services, Cybersecurity and Safety Research Group, University of Twente

Correspondence
Floris Erich, School of Integrative and Global Majors, University of Tsukuba, Tsukuba, Japan.
Email: erich@ai.iit.tsukuba.ac.jp

Abstract

Organizations are introducing agile and lean software development techniques in operations to increase the pace of their software development process and to improve the quality of their software. They use the term *DevOps*, a portmanteau of development and operations, as an umbrella term to describe their efforts. In this paper, we describe the ways in which organizations implement DevOps and the outcomes they experience. We first summarize the results of a systematic literature review that we performed to discover what researchers have written about DevOps. We then describe the results of an exploratory interview-based study involving 6 organizations of various sizes that are active in various industries. As part of our findings, we observed that all organizations were positive about their experiences and only minor problems were encountered while adopting DevOps.

KEYWORDS

DevOps, software development life cycle, agile software development, empirical study, qualitative interviews, systematic literature review

1 | INTRODUCTION

In recent years, the cost of releasing software has decreased dramatically due to the shift from shrink-wrapped software to software as a service.[1] Organizations that can release software early and with a high frequency have a higher capability to compete in the market.[2] A new approach called *DevOps* promises to allow software organizations to reach these goals. DevOps has been defined as an organizational approach aimed at creating empathy and cross-functional collaboration.[3] DevOps has also been called a "stub for more global company collaboration."[4] The goal of DevOps has been defined as that of reducing the time between development and operation of software without negatively affecting quality.[5] International Business Machines has coined the term *Collaborative DevOps* as "designing processes for coordinating software development teams with IT operations teams."[6] DevOps has furthermore been described as infrastructure being governed by the same processes that govern development.[7]

There are various examples of organizations practicing DevOps, including Flickr, Netflix, and Etsy. Allspawn and Hammond argued that through cooperation between development and operations personnel, Flickr was able to release code over 10 times a day.[8] Rembetsy and McDonnell reported on how Etsy transformed its culture into a DevOps culture.[9] In these examples, the presenters claim that by implementing DevOps and variants thereof, they have improved the software delivery within their organization.

Many organizations seem interested in this new approach of organizing development and operations, as shown by the number of publications dealing with DevOps in popular press*. DevOps has also become a topic of active scientific research, as demonstrated by the increasing number of scientific papers published on the topic†.

We first came into contact with DevOps during a company visit to one of the largest financial institutes in The Netherlands. We had not heard of DevOps before that visit, but were intrigued by how advanced the company was in adopting DevOps and by the described benefits of DevOps. We were interested in finding out more about how organizations practice DevOps and what results they achieved by doing so. We started out by performing a systematic literature review (SLR) on the literature surrounding DevOps. We discovered that there exists little agreement about the characteristics of DevOps in the academic literature. Therefore, to resolve this disagreement, we performed our own study of how DevOps was being applied in practice and interviewed 6 organizations about their implementation of DevOps. We describe the research methodology in Section 2,

*A search for DevOps on press release indexing website www.PRNewsWire.com for a period of 180 days resulted in nearly 100 results in the Spring of 2015, over 160 results in the Spring of 2016 and around 300 in the Spring of 2017.
†In the ACM Digital Library, DevOps was mentioned in 1 paper published in 2011, 7 papers published in 2013, 7 papers published in 2014, 30 papers published in 2015 and 41 papers published in 2016

the results of the SLR in Section 3, and the results of the interviews in Section 4. In Section 5, we summarize our findings and discuss them. We conclude the paper in Section 6 with implications for research and practice.

This paper contributes to the research and practice of DevOps by (1) giving a comprehensive overview over what has been written about DevOps in academic literature and (2) by validating the findings through interviews.

2 | METHODOLOGY

2.1 | Literature review

As a starting point of our DevOps research, we performed an SLR[‡]. The main research objective of the SLR was to discover whether DevOps is beneficial for software engineering. To determine this, we used the following research questions:

1. How does the literature define DevOps? (RQ1)
2. Which effects can be observed when DevOps is being practiced? (RQ2)
3. What are the supportive factors in implementation of DevOps? (RQ3)
4. How are DevOps characteristics being implemented? (RQ4)

We adopted a qualitative approach for analyzing the included studies.[13] First, we read the papers, wrote research notes, and applied labels to each paper. We then grouped the labels with a similar meaning.[14] Whenever a label was added, we cross-checked whether this label could also be applied to the papers processed earlier. This approach was inspired by Grounded Theory,[15] in which newly collected knowledge is compared with existing knowledge throughout the research process. We used a spreadsheet for recording the labels.

2.2 | Interviews

We continued our research by collecting and analyzing empirical data in organizations that state that they use DevOps. We interviewed senior employees at a diverse set of organizations of different sizes, located in different countries, and active in different industries.

The goal of our exploratory interview-based survey was to develop an understanding of the practical usage of DevOps by organizations and to discover what benefits they gain from its usage. We conducted focused interviews[16] on the implementation and usage of DevOps principles and practices at 6 organizations in total. We applied the interview methodology described by Kvale.[17] Four of these organizations are based in The Netherlands, one organization is based in the United States, and one is based in the United Kingdom. We used a high-level question scheme to guide the focused interviews. The interviewees were allowed to deviate from the questions, and the interviewer was allowed to broaden or deepen the conversation by asking follow-up questions.

2.2.1 | Interview questions

In our interviews, we used 3 levels of questions.[18] Level 1 questions are the questions asked during the interview and are the means of gathering information for the higher-level questions. These questions differ per interview, are numerous, and sometimes hard to understand out of context. Some examples are

- Could you please introduce yourself and describe your affinity with DevOps?
- Could you describe how the transformation to DevOps started?
- Could you elaborate on the friction between development and operations personnel?
- Which problems did you encounter that made you decide to implement DevOps?

Level 2 questions are the research questions that we want to answer for each case. They consider single cases and are used as a checklist for structuring the interview and determining whether the goals of the interview were reached. The level 2 questions were

- Why did the organization decide to implement DevOps?
- How did the organization implement DevOps?
- What problems did the organization encounter when implementing DevOps?
- What were the expected and achieved results of implementing DevOps?

Level 3 questions are focused on patterns among the different cases. The level 3 questions were

- What problems do companies try to solve by implementing DevOps?
- What problems are encountered when implementing DevOps?
- What practices are considered part of DevOps?

[‡]Parts of this SLR has been described previously in a poster,[10] a short paper[11] and in an unpublished technical report[12]

2.2.2 | Interview subjects

Table 1 gives an overview of the interviewed organizations. We note that we replaced all of the names of the organizations that we interviewed with fictitious names, due to reasons of confidentiality. The interviewees were aware of this and therefore answered the interview questions with as much freedom as possible. It however means that we cannot reproduce the exact interview narratives, nor can we provide exact quotations.

The inverviewees of this research held senior positions at the following organizations:

- FinCom1, which provides banking products and services to private and corporate account holders. It has over 50 000 employees, operates worldwide, and has its headquarters in The Netherlands.
- FinCom2, which provides insurance products and services to private customers. It has over 3000 employees and is based in The Netherlands.
- SupportCom, which develops Software-as-a-Service customer support systems. It has over 300 employees, is based in The Netherlands, and has customers from all over the world.
- PortalCom, which develops Software-as-a-Service content management systems. It is a division of a large multinational organization with over 100 employees in The Netherlands and around 5000 employees worldwide.
- UtilCom, which provides cloud-based services worldwide and has over 1000 employees.
- CommunitySoft, which is a global online open source community, which develops software for nonprofits, and has around 1800 members. It is registered as a charity in the United Kingdom.

The first author attended a conference on cocreation to recruit participants. He asked the representatives of various organizations if they were currently practicing DevOps and if so, if they were interested in participating in the research. Three Dutch organizations were selected this way. A fourth organization in The Netherlands was hand-picked by the first author, as it was a company in which he first encountered DevOps being used. For each organization, we asked a representative of the IT department to recommend an expert on DevOps within their organization. The interviewee at UtilCom was recruited through a personal connection. Finally, the interviewee at CommunitySoft was part of the staff of an online course through which the first author met him.

We interviewed specialists with various roles: 2 managers of IT departments, a senior analyst, a project management coach, a DevOps team member, a senior software engineer, and a senior manager (a cofounder of a company). The interviewees reported their experiences in adopting DevOps within their organization. The unit of analysis was the department in which the interviewee was active. Table 2 gives an overview of the collected data.

The number of employees that were working at each organization at the time of the interviews ranged from less than 100 to over 50 000. At most organizations, one employee was interviewed for a one-hour session, and further communication took place via e-mail afterwards. This differs in the case of the interview with FinCom2, as here, 2 employees were interviewed in a single session. At PortalCom, the interview took place over the phone. At CommunitySoft the interview took place entirely over e-mail due to time and location constraints.

2.2.3 | Data analysis

The researcher conducting the interviews (the first author) took notes during the interviews. Furthermore, the interviews were recorded using a voice recorder. After the interview, the interviewer digitized the notes and transcribed the recordings. For the interviews conducted at the com-

TABLE 1 Organization details

Organization	No. of Employees	Type of Systems Developed	Existing Processes
FinCom1	50 000+	Finance	Scrum, ITIL, CMMI
FinCom2	3000+	Finance	Scrum, ITIL, CMMI
SupportCom	300+	Customer relations	Scrum
PortalCom	1000+	Content management	Mixed
UtilCom	1000+	Cloud computing	Mixed
CommunitySoft	1800+	Software for nonprofits	Scrum

TABLE 2 Interview statistics

Organization (interviewees)	Interview Setting	Duration	Interviewee Role
FinCom1 (1)	On location, e-mail	00:54:53	DevOps team manager
FinCom2 (2)	On location, e-mail	01:02:37	1: Business analyst
			2: Agile coach
SupportCom (1)	On location, e-mail	00:44:32	Project manager
PortalCom (1)	Phone, e-mail	00:43:04	DevOps team member
UtilCom (1)	On location, e-mail	00:49:27	Software engineer
CommunitySoft (1)	E-mail	n/a	Cofounder, Project manager

panies in The Netherlands, the interviewer used Dutch as the language of conversation and then translated the transcriptions to English. The interviews with the representatives of the organization in the United States and the charity in the United Kingdom were conducted in English. We performed open coding on the translated interviews to create a set of theoretical codes.[19] Then, we organized the codes in a hierarchical map using the XMind mind mapping software.[20] By coding the interview transcripts and grouping similar codes, we identified 11 top level codes, which enable us to summarize the cases: (1) Overall definition of *DevOps*, (2) role of *individuals*, (3) role of *teams*, (4) role of *departments*, (5) realization of a *culture of collaboration*, (6) *automation* of processes, (7) *measurement* of process effectiveness, (8) *monitoring* of product effectiveness, (9) implementation of practices from *lean* software development, (10) *sharing* of information between development and operations, and finally, (11) how DevOps relates with *(micro)services*. We summarize the cases based on these codes. These summaries can be found in the tables referred to in the individual sections about each organization. Finally, we assign a weight to the applicability of each code to each case, which we use to create an overall summary of the interviews.

3 | LITERATURE REVIEW RESULTS

We started our research by doing an SLR, executed in the spring of 2015. Table AI describes the quality of the studies, gives a short summary, and relates each paper to the research question it contributes to.

We labeled each paper and then grouped similar labels. This resulted in 7 top labels:

Culture of collaboration (8 papers) The culture within an organization. Organizations practicing DevOps try to remove the cultural barrier between development and operations personnel.

Automation (8 papers) Automation within the software process. Organizations practicing DevOps aim for a high degree of automation.

Measurement (3 papers) Metrics within the software process. Organizations practicing DevOps try to use metrics involving both development and operations disciplines, instead of separate metrics for both.

Sharing (5 papers) Sharing of information within the software process. Organizations practicing DevOps try to facilitate between development and operations by having shared systems for recording knowledge.

Services (4 papers) Structuring the organization around services instead of around disciplines. Organizations practicing DevOps for example use cloud services and the microservices architecture.

Quality Assurance (5 papers) The party involved with ensuring products and services have adequate quality. Organizations practicing DevOps sometimes try to incorporate QA into the software process.

Governance (7 papers) This label refers to the way in which organizations practicing DevOps are governed. One relevant issue is the integration of DevOps with standards such as ITIL.

We noticed that there are various gaps in the study of DevOps:

1. There is no consensus of what concepts DevOps covers, nor how DevOps is defined.
2. There is a lack of evidence on the effectiveness of DevOps.
3. DevOps has not been adequately separated from other practices such as ASD/APM, making it confusing to understand whether a principle or practice discussed in the literature is part of a DevOps implementation or part of an implementation of an ASD/APM methodology.
4. Principles and practices, which are suggested to be part of DevOps, have not always been clearly operationalized. The authors of the articles do not define the principles and practices clearly, and they have not assigned metrics to measure their effectiveness.
5. According to the research quality assessment criteria of Kitchenham (Table AII in the Appendix) the quality of the studies is quite low, with 1 rank 3 paper, 5 rank 4 papers, and the other papers being rank 5.

The studied literature defines DevOps in multiple ways:

1. Merging of development and operations:
   (a) the combination of development and operations[21-23]
   (b) effective collaboration between development and operations personnel through integration of processes, tools and data[24]
   (c) a setting in which the development and operations teams work together closely[7,25,26]
2. Development that supports operations:
   (a) developers support the operational use of software[27]
   (b) applying agile techniques to operational activities[28]
3. Centralized group: a centralized group with 2 aims. First, achieving consistency, efficiency, visibility, and predictability in development and operations. Second, to support the transition to iterative / agile methods.[29]
4. Incentive alignment between different roles including development and operations: Aligning incentives of every party involved in delivering software, emphasizing developers, testers, and operations[30]

5. Cultural movement: a cultural movement combined with software development practices enabling rapid development[31]
6. Either a job description or an area of expertise combining development and operations skills[32]

There is a lack of consensus on the relevant characteristics of DevOps, and hence, there is no agreement on the possible range and types of effects that an organization could expect to observe when applying DevOps. One effect that could be observed when DevOps is being applied is that organizations acquire a higher capability of bringing software into production.[4] Some of the papers discuss cases of DevOps, eg, at Advance Internet,[33] an unnamed organization,[29] and at Facebook.[27] The Facebook case is the most elaborate but does not present any measurements of DevOps effectiveness, while the other two cases are based on anecdotal evidence. The other studies report on DevOps from the experience of a single person or focus on a specific technique considered part of DevOps. In the case of papers discussing specific techniques, the evidence provided is not supported by experiment (for example, a collection of DevOps patterns[23] for which no evidence was provided, and a toolkit[22] for which no practical results were discussed).

We grouped our findings into 4 classes: Findings about the organization as a whole, to the development and operations staff, to the development staff only, and to the involvement of other personnel beyond development and operation specialists. The following list summarizes the findings of the SLR:

- Organizations
  a. modeling processes using System Dynamics to measure what level of Continuous Delivery is reached,[34]
  b. using the Knowledge, Skills, and Abilities framework to evaluate new hires and to identify how to train employees,[21]
  c. using three-tier knowledge management to support system innovation,[35]
  d. trusting and assigning more responsibilities to employees to create a culture of empowerment,[27]
  e. setting up a group answering questions and ensure top management support to support the DevOps implementation,[29]
  f. focusing on people instead of tools and processes when implementing DevOps,[36]
  g. applying various practices for supporting Continuous Delivery to increase customer feedback,[37,38]
  h. applying various techniques for integrating DevOps with existing standards such as CMMI and ITIL,[26]
  i. implementing a framework of metrics across development and operations to measure velocity across the development and operations boundary,[39]
  j. considering the challenges posed by the Software-as-a-Service business model on the development and operations processes to address these,[40]
  k. transforming their organizational culture to create a DevOps culture.[31]
- Development and operations personnel
  a. managing configurations as source code to be able to keep track of changes to configuration,[25,41]
  b. together deciding what a system should log and how it should log this to increase the value of logs,[42]
  c. applying practices from Behavior Driven Development in operations to test solutions,[43]
  d. cooperating regularly to increase work efficiency,[7]
  e. using an application life cycle toolkit to support software mass customization,[22]
  f. considering issues from the other discipline's perspective to avoid conflicts,[44]
  g. using web application scalability patterns to increase capacity while avoiding having to reinvent the wheel.[23]
- Developers studying operational artifacts such as standards to get a better understanding of operational requirements without increasing communication.[45]
- Quality Assurance personnel working together with development and operations personnel to apply techniques and practices to improve system monitoring.[32]

The review did not enable us to reach the main research objective of the SLR, as we have not found enough evidence to determine whether DevOps is indeed beneficial for Software Engineering. As we could not obtain satisfactory answers to our research questions through the review, and also because the answers we obtained came from publications with relatively low quality of evidence provided, we decided to conduct an exploratory interview study in 6 different IT organizations to study the practice of DevOps.

4 | INTERVIEW RESULTS

We first discuss each separate organization studied and then compare and contrast how the various organizations implemented DevOps.

4.1 | FinCom1

The interviewee at FinCom1 saw DevOps as the formalization of the Agile concept that a team is responsible for every aspect of a product, which includes operations. FinCom1 is one of the first large organizations that started to implement DevOps in The Netherlands and had been using it

TABLE 3 Summary of DevOps at FinCom1

Code	FinCom1:
DevOps	Defines DevOps as the formalization of the Agile concept that a team is responsible for every aspect of a product, which includes operations.
Individual	Uses the title DevOps Engineer for all development and operations personnel working in DevOps Teams.
Team	Uses DevOps Teams who are responsible for both developing and operating systems and services.
Department	Merged departments of development and operations into DevOps Departments.
Culture of collaboration	Has development and operations personnel working together daily. The organization is exploring how infrastructure personnel can work in DevOps Teams. A DevOps Team shares responsibilities towards delivering new features and functionalities in a stable manner.
Automation	Automates the process of releasing software through automated tests and is implementing Continuous Delivery.
Measurement	Measure effectiveness of systems and teams using an inspect-and-adapt-loop. Measures the time between inception of an idea and the idea actually being available as a service to customers. Has made usage by and satisfaction of customers the primary measure of system performance.
Monitoring	Has operations personnel monitoring systems while being in close contact with the users, so they can correlate problems reported by users with changes in the system conditions reported by the monitoring instruments.
Lean	Applies lean principles to improve both the effectiveness and efficiency of processes but only considers lean to be weakly related to DevOps.
Sharing	Encourages employees to speak their minds regarding process and product improvements and uses openness as one measure of employee performance.
(Micro)services	Sees DevOps as a way of organizing processes, separate from architecture.

for a few years at the time of the interview. FinCom1 has been mentioned by various other interviewees (eg, at FinCom2 and SupportCom) as a source of inspiration for their own adoption of DevOps. The interviewee had participated in the DevOps adoption from the start. The interview is summarized in Table 3.

4.1.1 | Why did FinCom1 decide to implement DevOps?

FinCom1 had 3 main goals that they wanted to achieve by implementing DevOps: Reduce lead-time, improve problem-solving, and improve feedback. Starting new projects took a very long time at the organization, as teams had problems obtaining development resources such as servers and software. During the project itself, it was hard to solve problems in which close collaboration between development and operations personnel was needed. By using DevOps, FinCom1 wanted to decrease the time required to solve these problems. The organization also wanted to increase the knowledge available to development personnel regarding customer satisfaction. Before adopting DevOps, information about customer satisfaction was collected by operations personnel, but seldom shared with development personnel.

4.1.2 | How did FinCom1 implement DevOps?

FinCom1 adopted Scrum, ITIL, and CMMI before adopting DevOps. FinCom1 introduced DevOps Teams, in which development and operations personnel work together on a daily basis. At the same time, the organization introduced a new employee role called DevOps Engineer. Employees having this role should have skills in both development and operations. Eventually, the organization wanted to use only DevOps Engineers within DevOps departments, instead of separate development and operations personnel in separate departments. FinCom1 also retrained their management personnel so that they can manage both development and operations personnel. To allow proper evaluation of the personnel active in a DevOps Team, their Human Resources department also changed the evaluation procedures. The organization is also currently automating its infrastructure processes.

4.1.3 | What problems did FinCom1 encounter when implementing DevOps?

FinCom1 considers openness to be a main component of a DevOps culture. With openness they refer to employees sharing what is on their mind and being given the freedom to do so. The organizational culture has to allow for this. Not all the employees were comfortable with this level of openness. For example, employees did not want to give negative feedback about projects and colleagues.

Development personnel also resisted the increase in responsibility. In the past, operations personnel were responsible for the availability and performance of systems. FinCom1 made the DevOps Teams, which included both development and operations personnel, responsible for the availability and performance of systems. Hence, development personnel had to carry more responsibilities than before, including participating in on-call rotations.

In DevOps, the team is responsible for the availability of their service or system under development. Teams organize on-call rotations when a service or system is expected to be available at all times. These are time periods to which people are assigned as being responsible for the system. Personnel is expected to be reachable during the time periods to which they are assigned.

Another problem with DevOps was friction between development and operations personnel, caused by development personnel perceiving the operations personnel's way of working to be ad hoc and chaotic. Development personnel argued that development work is easier to predict than operations work, and hence, managing both under the same process was difficult. Development personnel also considered the presence of operations personnel as distracting.

4.1.4 | What results did FinCom1 expect to achieve by implementing DevOps and how far have these results been achieved?

FinCom1 reported an improvement in the lead time of projects. Before adopting DevOps, there was no process for starting new projects. Because of this, it took a long time for projects to get the required infrastructure for development and operations. By introducing an automated process for this, the lead time was reduced from roughly 9 months to 9 weeks. The interviewee also reported an increase in software quality.

4.2 | FinCom2

FinCom2 sees DevOps as a way of collaboration in which processes are automated as much as possible. It has hundreds of employees working in a DevOps setting. At the time of the interview, the organization had been using DevOps for around half a year. The interviewees have been involved with the DevOps adoption since the start. The interview is summarized in Table 4.

4.2.1 | Why did FinCom2 decide to implement DevOps?

The interviewees at FinCom2 defined DevOps as a combination of Agile and Lean Software Development. They used Formula-1 as metaphor to describe DevOps. In the past, systems could be down for days at a time for maintenance activities to be performed. As systems at FinCom2 became more customer-facing, such long periods of downtime were no longer accepted. So the organization wanted to increase the availability of their systems. According to the interviewees, DevOps at FinCom2 is about solving problems that occur when implementing high levels of automation. Another motivation for increasing the level of automation was reduction in workforce size.

To measure availability, FinCom2 uses a concept called straight through processing (STP). Straight through processing is the execution of a process without manual intervention by an employee. When a system is unavailable, this might reduce the STP grade of business processes, which use that system. The STP grade is a percentage representing how much time is saved by automated processing. When systems are unavailable, employees perform the tasks the system was supposed to perform manually, or the tasks might be blocked until the system becomes available again. This negatively affects the STP grade of the process that the tasks are part of.

TABLE 4 Summary of DevOps at FinCom2

Code	FinCom2:
DevOps	Defines DevOps as a way of collaboration in which processes are automated as much as possible.
Individual	Believes it is important for employees to have overlapping skill in development and operations; however, they do not carry a special title. Describes a DevOps team has having both generalists and specialists.
Team	Thinks DevOps teams are a core element of adopting DevOps.
Department	Is reorganizing its IT departments to work in service lines hosting both development and operations personnel.
Culture of collaboration	Has development and operations personnel working together daily.
Automation	Has the policy to automate as much as possible. Gives employees time to spend on automation activities. Is implementing Continuous Delivery and automated testing with quality gates.
Measurement	Uses frameworks from UrbanCode and Xebia to measure at what level the organization is in various disciplines. Applies chain monitoring to control software performance.
Monitoring	Monitors both IT processes and business processes using instruments such as chain monitoring and STP metrics, describing its strategy as "monitor everything that moves."
Lean	Sees DevOps as a combination of agile and lean.
Sharing	Encourages employees to share their opinion regarding bottlenecks and problems and share important discoveries.
(Micro)services	Has each team working in a particular service line.

There were a lot of mistakes made when releasing new versions of software, for which most teams did not have an automated process. Release process automation is part of continuous deployment, and the organization considered DevOps to be a supporting factor in realizing Continuous Deployment. Continuous deployment is a practice in which a system automatically deploys new versions of a system whenever a change has been made. To implement this practice, a system like Jenkins can be used, alongside deployment automation tooling such as Puppet and Chef. By automating the release process, FinCom2 reduced the amount of mistakes being made when releasing new versions of software.

4.2.2 | How did FinCom2 implement DevOps?

FinCom2 uses Scrum and tries to adhere to principles from Lean Software Development, such as aiming to reduce waste. It has also adopted ITIL and CMMI. It sees DevOps as an extension to both Scrum and Lean. The driving principles of DevOps at FinCom2 are daily collaboration and workspace sharing between development and operations personnel.

Part of the DevOps approach is that software should go to production (the environment which end-users access) during every iteration. In the Agile, development methodology software might be shown to the customer at the end of every iteration; however, this did not mean software would be available for actual users. After adopting DevOps, FinCom2 now releases their software to production in every iteration.

The organization used frameworks from commercial parties such as Xebia (Table AIII in the Appendix) and UrbanCode[46] to guide the implementation of Continuous Delivery. FinCom2 combined these 2 frameworks to create a matrix to evaluate the implementation of continuous delivery. The matrix consisted of questions about the software process and a scale of 4 levels, ranging from basic to extreme. Part of the framework consisted of questions related to DevOps. Hence, in this framework, DevOps is considered to be related to continuous delivery.

The interviewees mentioned how they want employees to apply systems thinking skills. Besides that, they also considered encouraging employees to be open and willing to "hang out your dirty laundry" as part of DevOps. This means that personnel should be open in their communication.

4.2.3 | What problems did FinCom2 encounter when implementing DevOps?

As part of implementing DevOps, FinCom2 requested its employees in a DevOps Team to spend more time on process improvements. According to the interviewee, employees were resistant to do this, valuing new feature development over improving the process. The interviewee also mentioned that management was skeptical about implementing DevOps, as its costs, benefits and risks had not been sufficiently explored.

4.2.4 | What results did FinCom2 expect to achieve by implementing DevOps and how far have these results been achieved?

FinCom2 used an assessment tool. At the time of the interview, an initial assessment was taken by all of the teams, and some teams had performed a follow-up assessment. The assessment showed that one of the DevOps Teams climbed from base level to beginner level in testing according to the Xebia model, meaning that they integrated automated testing in their continuous integration pipeline.

4.3 | SupportCom

SupportCom sees DevOps as the process of removing barriers between development and operations personnel. It has one DevOps Team. The adoption of DevOps started a few months before the interview. The interviewee had been involved from the start with adopting DevOps in this team through his role as Project Manager. The interview is summarized in Table 5.

4.3.1 | Why did SupportCom decide to implement DevOps?

The interviewee at SupportCom defined DevOps as being development and operations personnel working closely together, by for example sharing the same physical space and being members of the same team. SupportCom develops a product that can either be hosted on the servers of SupportCom or the servers of their customers. The former model is called Software as a Service (SaaS) while the latter is called on-premise. SupportCom wanted to reduce the release time of its SaaS-based software by using DevOps. They believe that miscommunication between development and operations personnel is a major factor in the high release time. Due to the increase in the complexity of their software, they expected more communication between development and operations personnel was needed. SupportCom believes that DevOps improves the communication between development and operations personnel.

4.3.2 | How did SupportCom implement DevOps?

SupportCom used Scrum before adopting DevOps. They had multiple teams working on separate features of its software and decided to turn one of these teams into a DevOps Team. According to the interviewee, the DevOps Team naturally evolved from a team which was working on components specific to its SaaS based software. The organization started to experiment with continuous delivery, but at the time of the interview was still at a very early stage of adopting it.

TABLE 5 Summary of DevOps at SupportCom

Code	SupportCom:
DevOps	Defines DevOps as a process of removing barriers between development and operations personnel.
Individual	Does not use a title resembling DevOps Engineer, but do want to work towards merging the skills of development and operations personnel.
Team	Has one DevOps team.
Department	Still has separate development and operations departments and has no plans to merge these due to the size of the DevOps adoption.
Culture of collaboration	Has members of the DevOps team working together at a central location.
Automation	Allows development and operations personnel to use 20% of their time for automating the software process, and is planning to implement Continuous Delivery.
Measurement	Is experimenting with phased rollouts and feature switches to allow them to measure software quality.
Monitoring	Currently only has few monitoring systems in place.
Lean	Is familiar with lean but does not see lean as a core part of DevOps.
Sharing	Believes that information sharing is increased within the DevOps Team, but that this decreases information sharing with management personnel.
(Micro)services	Was motivated to adopt DevOps to support the offering of a SaaS based version of its product.

4.3.3 | What problems did SupportCom encounter when implementing DevOps?

The interviewee mentioned how development personnel felt like their work was more structural than that of operations personnel. This leads to some friction between development and operations personnel.

Management personnel at SupportCom believed that DevOps made it harder to monitor the communication between development and operations personnel. Before creating the DevOps team, development and operations personnel communicated via official channels, whereas a DevOps Team used more face-to-face communication.

4.3.4 | What results did SupportCom expect to achieve by implementing DevOps and how far have these results been achieved?

The interviewee at SupportCom mentioned that implementing DevOps increased the speed and effectiveness of problem-solving. The interviewee attributed this to the improved communication between development and operations personnel, as problems were raised earlier, and as a result, fewer mistakes were being made. The organization observed fewer escalations of problems between development and operations personnel, as DevOps Teams were resolving problems internally, instead of using a formalized processes. The interviewee thought that by communicating about problems over more informal communication channels, the problems could be resolved more efficiently and more effectively.

4.4 | PortalCom

PortalCom sees DevOps as an aspect of organizational culture, in which development and operations personnel work together closely. The organization had started to adopt DevOps in a single team a few months before the interview took place. The interviewee has been involved with adopting DevOps from the start and was one of the key evangelists in bringing DevOps to the organization. The interview is summarized in Table 6.

4.4.1 | Why did PortalCom decide to implement DevOps?

The interviewee at PortalCom defined DevOps as being a cultural intervention to increase the understanding between development and operations personnel, and mentioned how DevOps is part of a movement towards integration of all phases in the software process. PortalCom started to offer cloud software and has implemented DevOps to increase software process velocity and quality of the software product. The organization hoped to increase customer satisfaction by releasing software more often. They also hoped to free more resources to spend on new features, instead of spending resources on maintenance of existing features. A future plan of PortalCom is to extend DevOps principles and practices to software testing and quality assurance.

4.4.2 | How did PortalCom implement DevOps?

PortalCom has created a single DevOps Team that experiments with DevOps tools, principles, and practices. Employees within this team have multidisciplinary skills. PortalCom started using tools such as version control and also developed their own tools to easily create cloud environments.

TABLE 6 Summary of DevOps at PortalCom

Code	PortalCom:
DevOps	Defines DevOps as an aspect of organizational culture in which development and operations personnel work together closely.
Individual	Does not use the title of DevOps Engineer, but does consider members of a DevOps team to need a wider set of skills than in a regular software development team.
Team	Has one DevOps team.
Department	Has not established a DevOps Department because the size of the DevOps adoption is still too small to profit from this.
Culture of collaboration	By having development and operations collaborate, they will start to speak each other's language and gain understanding.
Automation	Uses various tools for realizing automation, such as version control and automated testing, and is implementing Continuous Delivery.
Measurement	Did not introduce any process or performance measurement instruments yet.
Monitoring	Has no strong focus on monitoring.
Lean	Does not consider lean as part of DevOps.
Sharing	Has not established special systems for information sharing but relies on the closer proximity of development and operations personnel. ρ
(Micro)services	Sees DevOps as a vital element of a transition to services.

4.4.3 | What problems did PortalCom encounter when implementing DevOps?

For PortalCom, the main problem in implementing DevOps was the need for restructuring. According to the interviewee, members of a DevOps Team needed to have a wider array of skills and were hence in high demand. As a result of this, it was not always possible to assign people with enough DevOps experience to every team.

4.4.4 | What results did PortalCom expect to achieve by implementing DevOps and how far have these results been achieved?

The interviewee at PortalCom mentioned that by using DevOps they had increased their speed of software development. This was because DevOps enabled earlier and more frequent communication between development and operations personnel. This reduced the time needed to set up development environments. The interviewee however also mentioned the lack of metrics to determine the effectiveness of the DevOps implementation.

4.5 | UtilCom

The interviewee at UtilCom sees DevOps as the principles and practices that are needed to create a scalable service infrastructure. The interviewee joined UtilCom a few years before the interview took place, while the DevOps adoption had already started before that. At UtilCom, thousands of employees worked in a DevOps setting. The interview is summarized in Table 7.

4.5.1 | Why did UtilCom decide to implement DevOps?

UtilCom started implementing principles and practices, which are today often associated with DevOps, before the term DevOps was introduced. There was hence no explicit decision to implement DevOps. According to the interviewee, DevOps principles and practices grew organically to deal with challenges in developing and operating a complex scalable service architecture. After DevOps was considered a success, the leadership of the organization started to explicitly ask teams to implement DevOps principles and practices.

4.5.2 | How did UtilCom implement DevOps?

At UtilCom teams are largely free to choose which methodology they want to use, and most teams do not use a standard methodology. Teams typically use practices from various Agile Software Development methodologies such as Extreme Programming and Scrum, including daily stand-up meetings, a Scrum board, and sprints.

According to the interviewee, DevOps principles and practices should be included in every team responsible for offering a service. The interviewee mentioned that these principles and practices are related to software deployment and the management of infrastructure. DevOps could be seen as a role responsible for incident management, capacity management, risk management, and supporting the build process. The people who have this role would in the past be called server script engineers and would sometimes even be described using a derogatory term such as "script junkies." One of the goals of implementing DevOps was to give more respect to this role. The rising complexity of operations due to the introduction of cloud computing led to a more systematic approach of performing operations, requiring operations personnel to learn various software development techniques.

TABLE 7 Summary of DevOps at UtilCom

Code	UtilCom:
DevOps	Defines DevOps as "the principles and practices which are needed to create a scalable service infrastructure."
Individual	Sees DevOps both as a skill set every employee should have experience with and also as a specialized role.
Team	Believes that every team should have some people with DevOps skills and also has a team specifically called the DevOps team.
Department	Has both development and operations personnel in their engineering department, without having a separate operations department.
Culture of collaboration	Tries to change its organizational culture so that more respect is given to people doing server engineering or scripting.
Automation	Sees automation as the first subject every person joining the DevOps team should learn about.
Measurement	Sees a difference in the way team members of software development and DevOps teams get measured, ie, devs in terms of features delivered and ops in terms of how many on-call operations participated in, and tries to implement measurements spanning both development and operations.
Monitoring	Created a DevOps team, which is responsible for monitoring the infrastructure which other teams and clients use, so that there always is enough spare capacity.
Lean	Does not consider lean to be a central element of DevOps.
Sharing	Sees documentation as the primary means of information sharing. Tries to reduce tribal knowledge (knowledge only held by a small group of people). Focuses on creating clear software interfaces and enforces usage of public interfaces.
(Micro)services	Sees DevOps as an important ingredient of building scalable service infrastructures.

The interviewee at UtilCom worked in a large, geographically distributed team of around 100 members. The team was divided into subteams, of which one was focused on DevOps. This team would alternatively be described as the infrastructure team. The team was mainly responsible for supporting other teams, by for example orchestrating capacity. Even though there was one team that was explicitly called the DevOps team, every team within UtilCom used the principles and practices of DevOps.

The interviewee also mentioned how being geographically distributed was beneficial for the DevOps team, as having people on multiple locations all over the world contributed to the redundancy of monitoring and support facilities. If for some reason personnel working in a facility in the United States could not be reached, due to for example a power outage, employees working from remote locations could still provide support.

The principles and practices mentioned by the interviewee were continuous integration (a practice in which several members working on separate parts of a system regularly merge their parts into a bigger whole), continuous testing (automatically checking a system for correctness by executing an automated test suite whenever a change is made), continuous deployment, feature switches/toggles, staged deployments, upgrade/downgrade testing (testing whether a software upgrade works properly, but testing whether the software can be downgraded if necessary), infrastructure as code (defining the infrastructure of a project in the form of code), and infrastructure as a service (being able to request infrastructure using an automated system and then paying for the infrastructure based on the amount of usage).

The interviewee mentioned how at UtilCom there are 3 kinds of deployments: deployment of software, deployment of configuration, and deployment of both software and configuration. This separation is necessary because in DevOps, feature switches/toggles are used. This means that code paths can be deactivated through configuration. This is useful when for example working on a new feature, if the feature is not yet ready to be used, it can be disabled through a configuration setting.

4.5.3 | What problems did UtilCom encounter when implementing DevOps?

DevOps teams at UtilCom had problems with evaluating their progress. Traditionally, the progress of development personnel would be evaluated based on the amount and complexity of features implemented, while operations would be evaluated based on the availability of service. The problem was caused by the terminology that was used, as according to the interviewee, work from both development and operations personnel could now be considered as output. Performance metrics on both team and individual level had to be changed. For development personnel, an example metric was the number of issues that were planned to be fixed in an iteration, compared to the number of issues actually fixed. For operations personnel, an example metric was the amount of on-call rotations they participated in.

4.5.4 | What results did UtilCom expect to achieve by implementing DevOps and how far have these results been achieved?

According to the interviewee, DevOps infrastructure automation at UtilCom had repeatedly reduced on-call escalations, false alarms, and duplicate alarms over the course of time. UtilCom verified the decrease in escalations by comparing historical escalation patterns before and after implementing DevOps automation steps.

TABLE 8 Summary of DevOps at CommunitySoft

Code	CommunitySoft:
DevOps	Defines DevOps as "getting development done and into operations".
Individual	Does not separate between development and operations personnel, meaning everyone has to have experience both in developing and operating the software.
Team	Has development teams who are also responsible for operating the software. For each project, there is a team; however, the membership of teams is flexible, people can easily join and leave a team and can be member of multiple teams at the same time.
Department	Is not divided into departments.
Culture of collaboration	Does not divide personnel as being in development and operations, creating a culture in which collaboration is natural.
Automation	Has a high degree of automation of the development and deployment processes, only releases to staging and production require a manual sanity check.
Measurement	Has a culture of experimentation with new techniques, but the evaluation is mostly done qualitatively.
Monitoring	Uses no monitoring systems.
Lean	Uses a Kanban style project tracker.
Sharing	Has teams primarily communicating using a messaging app and holds Scrum-style daily meetings using Google Hangouts.
(Micro)services	Has no focus on services: Most products are web applications primarily controlled using a graphical front-end.

4.6 | CommunitySoft

The interviewee at CommunitySoft saw DevOps as "getting development done and into operations." They had been using DevOps principles and practices for a few years. The interview is summarized in Table 8.

4.6.1 | Why did CommunitySoft decide to implement DevOps?

To get direct feedback from stakeholders (customers or clients), CommunitySoft tried to adhere to the Agile Manifesto and also followed the "release early, release often" philosophy.[47] On a frequent basis (eg, weekly) the stakeholders got to see and use a working prototype of the system. The community preferred to collaborate as much as possible with the customer and to do this a completely functioning version of the latest system under development was needed. DevOps practices supported the agile approach of the community and helped in delivering solutions to stakeholder problems.

4.6.2 | How did CommunitySoft implement DevOps?

Teams at CommunitySoft are largely self-organizing, and the teams decided which tools, principles, and practices they used. The following describes a typical architecture in terms of processes and technology.

CommunitySoft used continuous integration. Developers made a fork of the repository of the system under development and worked locally on their fork. If they wanted to integrate their code (because they for example implemented a new feature or fixed a bug) into the original repository, they submitted a pull request. Automated CI systems automatically performed an experiment in which the code from the pull request was merged, and a test suite was run. Project managers then reviewed both the test results and other metadata associated with the pull request. If the pull request got accepted, the code would become part of the original repository. Then automated continuous deployment systems proceeded with the new version of the system in the development pipeline. CommunitySoft used cloud-hosted servers in which software was hosted in 3 environments: development, staging, and production. The development environment was always aligned with the latest integrated development version of the system under development. The staging environment represented the production environment. The production environment was accessed by end-users. Project managers at CommunitySoft manually performed sanity checks to decide whether the system could be deployed to staging and to production once the stakeholder approved it.

CommunitySoft used pivotal tracker and Waffle.io to create a kanban, which is an overview of the work that has been done, will be done, and currently is being done. Git and GitHub were used as version control systems, and they used Heroku for managing their cloud-hosted development pipeline. They also used various continuous integration tools such as Semaphore, Travis, and CodeShip.

To teach community members about how to use all the tools, principles, and practices, the community recommended participants to take 2 online courses, one about engineering software as a service and one about managing distributed teams.

As part of the Scrum methodology, every team had a weekly retrospective in which the members discussed what went right and what went wrong during the previous iteration. Based on this, the team chose a small number of adjustments that it can start implementing and testing in the following

iteration. The interviewee described this as a process of constant cautious experimentation. Some of the experiments performed were for example testing new process support tools, adjustments to deployment strategies, and further automating the deployment pipeline.

4.6.3 | What problems did CommunitySoft encounter when implementing DevOps?

CommunitySoft performed manual sanity checks, which cost up to 2 days. This was still adequate for a weekly release cycle and hence was not a large problem. The members of the community had different levels of skills, some, for example, combined their membership with a job as a Software Engineer while others had taken only 1 or 2 online courses on programming. With people of so many different backgrounds, there were sometimes discussions about nuances of implementing Agile or DevOps. This offered a learning opportunity for the members of CommunitySoft. The interviewee mentioned how it is hard for members of CommunitySoft to find a proper balance between discussing the process and working on developing and operating products. The interviewee stressed that sometimes the best approach is not to discuss directly but to just try something for a week or two and then, with some experience gained, to revisit the discussion.

4.6.4 | What results did CommunitySoft expect to achieve by implementing DevOps and how far have these results been achieved?

CommunitySoft had adopted DevOps practices because it wanted to continuously deliver working software to stakeholders, get their feedback, and then iterate and improve. Their results were anecdotal, with working software being delivered to stakeholders for over 3 years, and feedback having played an important role in improving the software.

The interviewee mentioned how it was difficult to say whether DevOps principles and practices had delivered the expected results. The interviewee argued from experience that without DevOps, the principles and practices would not have worked as well and would not have produced solutions that adequately addressed the real concerns of stakeholders. Hence, even though a company could ask its developers to implement practices such as continuous integration and continuous delivery, they would not be as effective without having interaction between both development and operations personnel.

4.7 | Comparison of DevOps adoptions

We compared the DevOps adoption at the 6 organizations through 3 means. First, Table AIV summarizes the interviews in the form a partially ordered meta-matrix.[48] Second, we explicitly answer the level 3 questions. Third, we weigh the applicability of the top level codes of the interviews to each individual interview and summarize the assigned weights in Table 9.

4.7.1 | What problems do organizations try to solve by implementing DevOps?

In the experiences of our participants, organizations are trying to solve a wide variety of issues by implementing DevOps. FinCom1 wanted to reduce lead-time, improve problem solving, and improve feedback. FinCom2 wanted to support automation. SupportCom and PortalCom wanted to reduce release time. PortalCom furthermore wanted to increase velocity and quality, as well as free resources to work on implementing new features. UtilCom also wanted to increase velocity and quality. CommunitySoft also wanted to improve feedback.

Implementing DevOps has led to the achievement of a few results. FinCom1 claimed that their lead time had improved. FinCom2 mentioned how one team had improved its automated testing capabilities. SupportCom noticed better problem-solving and fewer escalations caused by conflicts between development and operations. PortalCom mentioned how its velocity had increased. UtilCom had been able to turn most of its services into public facing services. CommunitySoft had deployed software to customers regularly and because of this, was able to receive and incorporate feedback rapidly.

TABLE 9 Applicability of top level codes. 0 = not applicable, 1 = applicable, 2 = strongly applicable

Code	FinCom 1	FinCom2	SupportCom	PortalCom	UtilCom	CommunitySoft
Individual	2	1	1	1	2	2
Team	2	2	2	2	2	2
Department	2	2	0	0	2	0
Culture of collaboration	2	2	2	2	2	2
Automation	2	2	2	2	2	2
Measurement	2	2	2	0	2	0
Monitoring	2	2	1	0	2	0
Lean	2	2	0	0	0	1
Sharing	2	2	1	0	2	2
(Micro)services	1	1	1	1	2	0

4.7.2 | What problems are encountered when implementing DevOps?

Employees working in a DevOps environment need to have a wide variety of technical and interpersonal skills. Software architecture is no longer limited to the deployed system running at the customer infrastructure, and also includes the systems being deployed for development and staging as well as the architecture required to support the development and operations pipeline. This new architecture is supported by a wide variety of tools, including systems for version control (a practice in which different versions of artifacts produced or consumed by a team are stored at a central location) such as SVN and Git, Continuous Integration systems such as Jenkins and Travis, and Infrastructure Automation tools such as Puppet and Chef.

The development and operations pipeline not only consists of technical components but also includes people from various disciplines who are contributing to the artifacts processed by the pipeline. These people need to be open to collaboration between different roles.

Because DevOps is not a precisely defined concept, organizations have to find their own way of implementing it. This might slow down the rate of adoption and possibly cause a lot of discussion among employees.

The interviewee at FinCom1 mentioned how some employees are concerned with the sharing aspect of DevOps, which requires a high level of openness. Developers are worried about the increase in responsibilities, including being within reach when problems occur. Development personnel also raised concerns about differences in work style between development and operations personnel, the latter being considered to be ad hoc or chaotic according to developers. The interviewees at FinCom2 mentioned that team members felt like they did not have the opportunity to work on process improvements due to the pressure to work on product improvements. According to the interviewees at FinCom2, management was skeptical about DevOps due to a lack of quantitative proof of its effectiveness. The interviewee at SupportCom mentioned that management raised concerns that closer collaboration between development and operations personnel decreased management's capability of performing their tasks. The interviewee at PortalCom mentioned that employees who have both development and operations skills are rare, and hence, having such a person in every team is considered impractical. The interviewee at UtilCom mentioned that it required a change in mindset to align tracking the progress of development activities with operations activities, and that teams required new metrics. CommunitySoft has not been able to eliminate all manual checks and has had a lot of discussion about how to implement DevOps principles and practices.

4.7.3 | What practices are considered part of DevOps?

In DevOps, there is a high degree of interaction between development and operations personnel, up to a level where some organizations no longer differentiate between both types of personnel. There are various ways in which the interviewed organizations have formalized this interaction. Some organizations have introduced official titles, eg, DevOps Engineers, DevOps Teams, and DevOps Departments. They have thus introduced organizational roles to support the DevOps transition. Other organizations do not think these formal structures are needed. They therefore think that DevOps should be an implicit way of working for every team member and leave it up to the teams themselves to decide how to implement DevOps in their software process. In any case, the roles of development and operations merge, sometimes partially, and sometimes completely. According to the interviewees, continuous delivery is an important component associated with the adoption of DevOps.

5 | DISCUSSION

We see DevOps as interaction between development and operations personnel. In the literature and interviews, DevOps interaction takes place primarily on 3 levels: individuals, teams, and departments.

To develop the tables used to give an overview of DevOps at each organization interviewed, we included the findings of the SLR in the following ways:

- Culture, automation, measurement, and sharing were all mentioned in multiple interviews.
- Governance was a top label in the literature review. We discussed governance with the interviewees; however, we found it to be implicitly part of how the organizations managed personnel, teams, and departments.
- We found little proof of Quality Assurance (QA) to be an explicit part of DevOps and hence did not use it in the tables. Including QA in DevOps seems to be part of a larger movement towards applying the DevOps concept of interaction between multiple disciplines, to other areas. The interviewees at FinCom2, for example, also discussed their idea of including the business side in DevOps, calling their approach *BizDevOps*.
- Services were mentioned in the practical literature (the microservice architecture is considered to be a central part of DevOps adoption[5]), the academic literature (DevOps is argued to be a strategy to manage cross-functional teams working on systems consisting of services[30]) and by practitioners (eg, the interviewee at UtilCom described a transformation to a more service oriented organization). We grouped these concerns regarding service orientation under the term *(Micro) services*.

The literature and interviewees mention the role of DevOps as a personnel trait. The usage of DevOps to describe an employee role (eg, a DevOps Engineer) is disputed.[32] In our interviews, we noticed the same thing, as FinCom1 uses the term DevOps Engineer, while the interviewees at FinCom2 said that DevOps Engineers *do not exist*. However, the existence of personnel with skills in both development and operations was acknowledged by all the interviewees.

The team level is the next level at which DevOps was applied. Five out of the 6 organizations that we interviewed had one or more teams that the interviewees described as a *DevOps team*. Some of the interviewed organizations were transforming all of their teams to become DevOps teams (FinCom1 and FinCom2). Three organizations interviewed thought that DevOps teams were only needed in special cases (SupportCom, PortalCom, and UtilCom). One organization (CommunitySoft) had teams that were responsible for both development or operations, but they did not use the term *DevOps team*.

The next level on which DevOps can be applied is the departmental level. One paper from the SLR discussed DevOps as if it was a separate group or department within an organization.[29] The organizations studied did not share this definition of DevOps. The interviewee at FinCom1 mentioned how development and operations departments were merged into DevOps departments. At FinCom2, development and operations teams worked together in business lines, but this predated the implementation of DevOps. SupportCom and PortalCom still considered development and operations personnel to work for separate departments. UtilCom and CommunitySoft did separate development and operations personnel into separate departments, but they did not make this decision as part of a DevOps adoption.

Practitioners have defined various mnemonics to describe DevOps: CAMS (Culture, Automation, Measurement, Sharing),[49] CALMS (CAMS + Lean),[50] and CAMM (Culture, Automation, Measurement, Monitoring).[51] In the interviews, culture and automation are universally considered to be core parts of DevOps adoption. Measurement, monitoring, and sharing were considered important by most but not all interviewees. Lean is only considered to be part of DevOps by a small subset of the organizations. An orientation towards (micro)services is correlated to DevOps adoption in most cases but is rarely considered to be a core element of DevOps adoption.

Our results show that there is no consensus in the literature on the desired effects of DevOps, and that the various cases presented in the publications relied mostly on anecdotal descriptions of DevOps. Also, DevOps was not operationalized properly in the literature and the researchers did not perform any experiments to verify the effectiveness of DevOps.

We do not see DevOps as a discrete concept, ie, organizations do not either "possess or not possess DevOps." We believe DevOps is something that is present in every organization that uses development and operations personnel. But the amount of interaction differs and depends on the particular organization.

We hoped that by studying actual organizations using DevOps, we could move towards operationalizing DevOps and finding some quantitative data to support the effectiveness of DevOps. DevOps is however rarely operationalized in the organizations interviewed. One metric discussed in the literature is the mean time between development of a feature and the release of the feature into operations.[5] None of the organizations interviewed mentioned that they used this metric.

Some frameworks exist for assessing a DevOps implementation. These are often qualitative that ask closed questions such as "Do development and operation personnel work together as part of a multidisciplinary team with shared responsibilities?" (from the Xebia framework), while neither specifying concretely what is meant with the construct, nor offering a metric for measuring the construct.

We believe that the effectiveness of implementing principles and practices should not be only measured subjectively. Based on this, we recommend researchers and practitioners to spend more time on operationalizing DevOps adoption, performing experiments, and measuring the results of experiments to show that DevOps adoption actually brings concrete quantitative as well as qualitative benefits.

6 | CONCLUSION

The main goal of our research was to determine whether DevOps has a beneficial effect on software organizations and their processes. We were hoping to find quantitative measures for the effectiveness of DevOps. We found that in practice, multiple definitions of DevOps exist. These definitions of DevOps seem to actually represent different perspectives of DevOps. We first performed an SLR and found that most publications report anecdotal experiences that the term DevOps is ill-defined and that little evidence has been offered showing the actual benefit of DevOps. For example, we could not find literature in which an organization states that they have implemented DevOps while providing quantitative data on observed benefits. We then studied 6 organizations to discover how DevOps is being implemented in practice. The interviewed organizations were positive about their experiences with DevOps and all of them described relatively smooth transitions to using DevOps. They however did not share any quantitative data to show the effectiveness of DevOps. Future work could focus on such research.

6.1 | Validity

We considered the 4 types of validity pertaining to empirical research: construct validity, internal validity, external validity, and reliability.[18]

Construct validity can be ensured using these 3 techniques[18]: using multiple sources of evidence, establishing a chain of evidence, and having key informants review the report. During this research, we have applied all 3 techniques. First, we used multiple sources of evidence as we interviewed 6 organizations and combined those findings with our findings from the literature review. Second, we established a chain of evidence by following a strict process consisting of preparation, performance, transcription, and coding. Third, all informants were given the opportunity to provide feedback on the report concerning their organization. Three of the 6 organizations provided feedback which we then incorporated in the final report.

Internal validity can be ensured using these 4 techniques[18]: doing pattern matching, explanation building, rival explanation exploration, and logic model construction. We have applied 3 out of 4 techniques. First, we applied pattern matching by categorizing the data using axial coding while retroactively applying codes to earlier studied cases. Second, we have explored rival explanations by relating DevOps principles and practices to those from Agile, Scrum, and Lean Software Development. Third, we have developed logic models for the cases by constructing concept maps. We did not formally apply explanation building; however, we have spent significant time on analyzing the literature and the cases by writing and discussing about the concept maps and attached notes.

External validity can be ensured using these 2 techniques[18]: using theory in single-case research and using replication logic in multicase studies. As this is an exploratory study, we did not start by selecting a theory. However, we setup a multicase research project in which we have applied replication logic, in the form of using a set of high-level questions that we used for every case studied. A critical external validity question that is important for exploratory studies concerns the extent to which our findings could be generalizable to other organizations embarking on DevOps.[52] The question is to what extent we expect to find similar findings if we had interviewed more organizations. While we could not claim universal generalizability based on our interviews at 6 organizations, the diversity of the interviewees (working for organizations active in different industries, in different countries), and the interviewees explaining DevOps from different perspectives, increases the generalizability of our results.

Reliability can be ensured using these 2 techniques[18]: using an interview protocol and developing a database of interviewed organizations. The first step of the research was to create an interview protocol. The protocol consists of questions (levels 2 and 3) related to the top labels from the SLR and practical details such as the sites to be visited, how data would be collected, and how we should prepare for the visits. We used this protocol throughout the project. We also established a database, in which we stored the audio recording, interview transcripts, coding results, research notes, and case descriptions. Due to reasons of confidentiality, this database is not publicly available.

6.2 | Implications for research

This paper contributes to the research of DevOps in 2 ways. First, it summarizes what has been written about DevOps in academic literature. Researchers can use this paper to get an overview of the academic literature and to find literature to explore further. Second, the paper summarizes the results of our interview research at 6 organizations that described their usage of DevOps. Researchers can use this as a starting point for researching DevOps in practice. Organizations can compare their approach to DevOps with the approaches of the 6 organizations that we interviewed.

6.3 | Implications for practice

We have presented a definition of DevOps being the interaction between development and operations at the individual level, team level, and departmental level. Organizations can clarify their communication and thinking about DevOps using this definition. We have listed the top level codes originating from the literature and interviews, which organizations can use as a tool to evaluate their own adoption of DevOps.

REFERENCES

1. Susarla A, Barua A, Whinston AB. A transaction cost perspective of the" software as a service" business model. *J Manage Inf Syst*. 2009;26(2):205-240.
2. Fox A, Patterson D. *Engineering software as a service: An agile approach using cloud computing*. India: Strawberry Canyon LLC; 2014.
3. Dyck A, Penners R, Lichter H. Towards Definitions for Release Engineering and Devops. In: Proceedings of the IEEE/ACM 3rd International Workshop on Release Engineering, Florence: 2015:3.
4. Debois. Opening Statement. *Cutter IT J*. 2011;24(8):3-5.
5. Bass L, Weber I, Zhu L. *Devops: A software architect's perspective*. New Jersey: Addison-Wesley Professional; 2015.
6. Rowe M, Marshall P. The business case for collaborative devops. Available from: https://www.ibm.com/developerworks/mydeveloperworks/blogs/invisiblethread/entry/the_business_case_for_collaborative_devops; 2012; Accessed June 13, 2017.
7. Loukides M. *What is DevOps?*. Sebastopol, CA: O'Reilly Media; 2012.
8. Allspaw J. 10+ Deploys Per Day: Dev and Ops Cooperation at Flickr. http://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-&LWx0FB02;ickr/; 2009; Accessed February 21, 2016.
9. Rembetsy M, McDonnell P. Continuously deploying culture: Scaling culture at etsy. http://www.slideshare.net/mcdonnps/continuously-deploying-culture-scaling-culture-at-etsy-14588485; 2012; Accessed March 11, 2016.
10. Erich F, Amrit C, Daneva M. Cooperation between information system development and operations: A literature review. In: Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement; Torino 2014:69.
11. Erich F, Amrit C, Daneva M. A mapping study on cooperation between information system development and operations. In: Proceedings of the 15th International Conference on Product-Focused Software Process Improvement, Lecture Notes in Computer Science, Helsinki, vol. 8892. Berlin: Springer; 2014:277-280.
12. Erich F, Amrit C, Daneva M. Technical Report of the University of Twente. Available from: http://www.utwente.nl/mb/iebis/staff/amrit/devopsreport.pdf; 2014; Accessed June 13, 2017.
13. Cruzes DS, Dyba T. Recommended steps for thematic synthesis in software engineering. In: Proceedings of the 5th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement, Banff; 2011:275-284.

14. Petersen K, Feldt R, Mujtaba S, Mattsson M. Systematic mapping studies in software engineering. In: Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering, Bari; 2008:68-77.
15. Charmaz K. *Constructing Grounded Theory (Introducing Qualitative Methods Series)*. 2nd ed. Thousand Oaks: SAGE Publications; 2014.
16. Merton RK, Kendall PL. The Focused Interview. *Am J Sociology*. 1946;51(6):541-557.
17. Kvale S. *Doing Interviews*. Thousand Oaks: SAGE Publications, Inc; 2008.
18. Yin RK. *Case Study Research: Design and Methods (Applied Social Research Methods)*. 4th ed. Thousand Oaks: Sage Publications; 2008.
19. Glaser BG, Strauss AL. *The Discovery of Grounded Theory: Strategies for Qualitative Research*. New York, NY: Aldine de Gruyter; 1967.
20. XMind. Xmind mind mapping software. Available from: http://www.xmind.net/. Accessed June 13, 2017
21. Bang SK, Chung S, Choh Y, Dupuis M. A grounded theory analysis of modern web applications: Knowledge, skills, and abilities for DevOps. In: Proceedings of the 2nd Annual Conference on Research in Information Technology; 2013; Orlando, FL:61-62.
22. Hosono S, Shimomura Y. Application lifecycle kit for mass customization on PaaS platforms. In: Proceedings - 2012 IEEE 8th World Congress on Services 2012; 2012; Honolulu, HI:397-398.
23. Cukier D. DevOps patterns to scale web applications using cloud services. In: Proceedings of the 2013 Companion Publication for Conference on Systems, Programming, & Applications: Software for Humanity, Proceedings - SPLASH '13; 2013; Indianapolis, Indiana, USA:143-152.
24. Ambler SW. Disciplined agile delivery and collaborative DevOps. *Cutter IT J*. 2011;24(12):18-23.
25. Spinellis D. Don't install software by hand. *IEEE Softw*. 2012;29(4):86-87.
26. Phifer B. Next-generation process integration: CMMI and ITIL do DevOps. *Cutter IT J*. 2011;24(8):28-33.
27. Feitelson DG, Frachtenberg E, Beck KL. Development and deployment at Facebook. *IEEE Internet Comput*. 2013;17(4):8-17.
28. DeGrandis D. Devops: So you say you want a revolution?. *Cutter IT J*. 2011;24(8):34-39.
29. Fitzpatrick L, Dillon M. The business case for DevOps: A five-year retrospective. *Cutter IT J*. 2011;24(8):19-27.
30. Humble J, Molesky J. Why enterprises must adopt DevOps to enable continuous delivery. *Cutter IT J*. 2011;24(8):6-12.
31. Walls M. *Building a DevOps Culture*. Sebastopol, CA: O'Reilly Media; 2013.
32. Roche J. Adopting DevOps practices in Quality Assurance. *Commun ACM*. 2013;56(11):38-43.
33. Shamow E. DevOps at advance internet: How we got in the door. *Cutter IT J*. 2011;24(8):13-18.
34. Akerele O, Ramachandran M, Dixon M. System dynamics modelling of agile continuous delivery process. In: Proceedings of the 2013 Agile Conference, Nashville; 2013:60-63.
35. Corbin RD, Dunbar CB, Zhu Q. A three-tier knowledge management scheme for software engineering support and innovation. *J Syst Softw*. 2007;80(9):1494-1505.
36. Mueller E. DevOps and the people who practice it: Winning their hearts and minds. *Cutter IT J*. 2011;24(12):6-11.
37. Neely S, Stolt S. Continuous delivery? easy! just change everything (well, maybe it is not that easy). In: Proceedings of the 2013 Agile Conference, Nashville; 2013:121-128.
38. Tessem B, Iden J. Cooperation between developers and operations in software engineering projects. In: Proceedings of the 2008 International Conference on Software Engineering, Leipzig; 2008:105-108.
39. Le-Quoc A. Metrics-driven DevOps. *Cutter IT J*. 2011;24(12):24-29.
40. Stuckenberg S, Fielt E, Loser T. The impact of software-as-a-service on business models of leading software vendors: Experiences from three exploratory case studies. In: Proceedings of the 15th Pacific Asia Conference on Information Systems: Quality Research in Pacific, Brisbane; 2011:184.
41. Schaefer A, Reichenbach M, Fey D. Continuous integration and automation for DevOps. *Transactions on Engineering Technologies, Vol.170 of the series Lecture Notes in Electrical Engineering*. 2012;345-358.
42. Shang W. Bridging the divide between software developers and operators using logs. In: Proceedings of the 2013 International Conference on Software Engineering, Zuerich; 2013:1583-1586.
43. Gohil K, Alapati N, Joglekar S. Towards behavior driven operations (BDOps). In: Proceedings of the 2011 International Conference on Advances in Recent Technologies in Communication and Computing; 2011; Bangalore:262-264.
44. Keyworth B. Where is IT operations within DevOps?. *Cutter IT J*. 2011;24(12):12-17.
45. Bass L, Jeffery R, Wada H, Weber I, Zhu L. Eliciting operations requirements for applications. In: Proceedings of the 1st International Workshop on Release Engineering, RELENG 2013; 2013; San Francisco, CA:5-8.
46. Bahrs P. Adopting the IBM DevOps approach for continuous software delivery; 2013 https://www.ibm.com/developerworks/library/d-adoption-paths/index.html; Accessed on 22nd February 2016
47. Raymond ES. *The Cathedral & the Bazaar*. Sebastopol: O'Reilly Media; 1999.
48. Miles MB, Huberman AM. *Qualitative Data Analysis*. 2nd ed.Thousand Oaks: SAGE Publications, Inc; 1994.
49. Willis J. What devops means to me. http://www.slideshare.net/\ignorespacesjezhumble/devops-and-agile-release-management; 2010. Accessed February 22, 2016
50. Humble J. DevOps and Agile Release Management. http://www.slideshare.net/\ignorespacesjezhumble/devops-and-agile-release-management; 2010. Accessed February 22, 2016
51. Lwakatare L, Kuvaja P, Oivo M. Dimensions of devops. In: Proceedings of the 2015 XP Conference Helsinki; 2015:212–217.
52. Wieringa R. *Design Science Methodology for Information Systems and Software Engineering*. Berlin: Springer; 2014.
53. Kitchenham B, Charters S. Guidelines for performing Systematic Literature Reviews in Software Engineering; Jul 2007.
54. van Steenis M, Verschure R, Zschuschen E. *Radicale versnelling van het software voortbrengingsproces door middel van continuous delivery*; 2014. Available from: https://xebia.com/downloads/\ignorespacescontinuous-delivery-andagile-transformations-dutch-only.pdf. Accessed February 22, 2016

How to cite this article: Erich FMA, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. *J Softw Evol Proc*. 2017;29:e1885. https://doi.org/10.1002/smr.1885

APPENDIX

TABLE AI Details about included studies. Column with header Q shows research quality based on Kitchenham. Column with header V shows venue: journal, conference, and (technical) report

Ref.	Q	V	Description	RQ contribution			
				RQ1	RQ2	RQ3	RQ4
34	5	C	The construction of a System Dynamics model to achieve a "repetitive, risk-free, and effortless continuous delivery process" is described. Simulation is used to verify the results. The article explains how validation could take place, but does not concretely define how it will take place.			✓	
24	5	J	Disciplined Agile Delivery is presented as an enterprise process for developing software, which includes DevOps. No validation takes place.	✓		✓	✓
21	5	C	The usage of Knowledge, Skills and Abilities as a framework for finding employees with DevOps skills is described. No validation is done and the process followed is quite unclear from the article.	✓			
45	5	C	Ways for developers to elicit operations requirements from documents instead of face to face communication are described. No validation takes place.	✓		✓	
35	5	J	How to support system engineering using knowledge management is described. No validation takes place.			✓	
23	5	C	A case study of using patterns, which cross development and operations, is described. The results are limited to a single case.	✓		✓	
28	5	J	An experience report of using techniques argued to be part of DevOps, such as system thinking, is described. No concrete cases are described to validate the results.	✓		✓	
27	5	J	A case study of the functioning of a company, which has a culture sharing characteristics with DevOps, is described. The results are limited to a single case.	✓			✓
29	4	J	A case study of the business implications on implementing DevOps is described. The results are limited to a single case.	✓	✓	✓	✓
43	4	C	An approach for applying testing techniques used in software development on operations is described. An experiment trying to prove that this is possible is described, but the experiment lacks rigor.			✓	
22	5	C	A platform for supporting development based on DevOps is described. No results of validation are presented.	✓		✓	
30	5	J	It is described how DevOps supports continuous delivery. No concrete cases are described to validate the results.	✓		✓	
44	5	J	The lack of involvement of operations in DevOps initiatives is described. No concrete cases are described to validate the results.	✓		✓	
39	5	J	The importance of metrics in a DevOps initiative is argued. No concrete cases are described to validate the results.	✓		✓	
7	5	R	The history of DevOps is described. Evidence is limited to industry examples.	✓		✓	
36	5	J	The importance of human factors in a DevOps initiative is described. No concrete cases are described to validate the results.	✓		✓	✓
37	4	C	Experience implementing Continuous Delivery is reported. The results are limited to a single case.			✓	
26	5	J	It is argued that DevOps can be used together with the CMMI and ITIL standards. No concrete cases are described to validate the results.	✓	✓	✓	
32	5	J	The importance of using DevOps practices in quality assurance is described. No concrete cases are described to validate the results.	✓	✓	✓	
41	5	J	The use of DevOps practices in an academic setting is described. The results are limited to a single case.	✓			
33	4	J	The experience using DevOps at a small organization is described. The results are limited to a single case.	✓	✓	✓	✓
42	5	C	Logging is presented as a way to improve cooperation between development and operations. To validate some preliminary results, a study is described on high level and its results are presented. There is no validation of the final solution.			✓	
25	5	J	It is argued that software installation should be handled by automated processes. No concrete cases are described to validate the results.	✓	✓		
40	4	C	The influence of Software-as-a-Service architecture on the business model is described. Validation was done by studying three cases.			✓	
38	3	C	Problems related to the cooperation between development and operations are explored. The research is validated by studying a focus group and two cases.	✓		✓	
31	5	R	A method for building a DevOps culture is described. No concrete cases are described to validate the results.	✓		✓	✓

TABLE AII Literature quality assessment by Kitchenham[53]

Rank	Description
1	Evidence obtained from at least one properly-designed randomized controlled trial
2	Evidence obtained from well-designed pseudo-randomized controlled trials (ie, non-random allocation to treatment)
3-1	Evidence obtained from comparative studies with concurrent controls and allocation not randomized, cohort studies, case-control studies, or interrupted time series with a control group.
3-2	Evidence obtained from comparative studies with historical control, two or more single arm studies, or interrupted time series without a parallel control group
4-1	Evidence obtained from a randomized experiment performed in an artificial setting
4-2	Evidence obtained from case series, either post-test or pre-test/post-test
4-3	Evidence obtained from a quasi-random experiment performed in an artificial setting
5	Evidence obtained from expert opinion based on theory or consensus

TABLE AIII DevOps integration characteristics according to the Xebia model[54]

Level	Characteristics
Level 1 (base)	Operations are actively involved in the final phase of the project.
Level 2 (beginner)	Code is released with release notes, which allow operations to perform installation and management.
Level 3 (average)	Development and operations work together when it is necessary.
Level 4 (advanced)	A representative from operations and a representative from development work together on projects.
Level 5 (complete)	Development and operations personnel work together as part of a multi-disciplinary team with shared responsibilities.

TABLE AIV Case level display for partially ordered meta-matrix: Adopting DevOps

	Motivation	Implementation	Problems	Results
FC1	Reduce lead time for new projects	Introduced DevOps teams	Employee discomfort with openness	Improved lead time
	Improve problem solving	Adopted DevOps Engineer job title	Resistance against increased Dev responsibilities	Increased software quality
	Increase feedback	Put Dev and Ops under same management	Devs considering Ops work as ad hoc and chaotic	
FC2	Reduce system downtime	Using Xebia and UrbanCode frameworks for measuring progress	Employees focusing on production rather than improving production capacity	Improved testing for one team
	Reduce workforce size	Introduced policy of making software available to actual users each iteration	Management skepticism due to lack of evidence of effectiveness	
	Support automation	Introduced time and location overlap between Dev and Ops		
		Trained employees in System Thinking		
SC	Reduce miscommunication between Dev and Ops	Introduced a specialized DevOps team	Devs considering Ops work as ad hoc and chaotic	Increased problem solving capabilities
	Reduce release time of SaaS product		Reduced capabilities for management	Improved Dev and Ops communication

TABLE AIV Continued

	Motivation	Implementation	Problems	Results
			oversight	
				Fewer escalations of issues
PC	Release software more often	Experiment with one team as DevOps team	Restructuring needed	Increased process velocity
	Free up resources to work on new features instead of problem solving	Automation (version control and environment provisioning tooling)	Team members needed to have a wide skillset	Less time spend on setting up environments
	Increase product quality			
	Increase process velocity			
UC	Deal with challenges in developing and operating complex scalable service architecture	Operations personnel trained in software development techniques	No terminology for evaluating progress duplicate alarms	Reduced escalations, false alarms and
		More respect given to people in a DevOps role than before	New metrics were needed for evaluating progress	
		Adopted systematic approach to operations		
		DevOps is considered as a role held by select people		
		Single team explicitly called DevOps team		
		Implicit focus on DevOps by all dev and ops		
CS	Get direct feedback from stakeholders	Introduced Continuous Integration as coordination tool	Hard to find balance between producing and improving production capability	Working software is regularly being delivered to stakeholders
		No separation between dev and ops roles	Not every member has sufficient skills for DevOps approach	DevOps as supporting instrument of adopting technical practices such as CI and CD
		Constant cautious experimentation	Difficulty automating everything	