Open Tools for Software Engineering: Validation of a Theory of Openness in the Automotive Industry

Hussan Munir
Lund University, Innovation
Engineering
Lund, Sweden
hussan.munir@design.lth.se

Per Runeson
Lund University, Department of
Computer Science
Lund, Sweden
per.runeson@cs.lth.se

Krzysztof Wnuk
Blekinge Institute of Technology,
Software Engineering Research Lab
Karlskrona, Sweden
krzysztof.wnuk@bth.se

ABSTRACT

Context: Open tools (e.g., Jenkins, Gerrit and Git) offer a lucrative alternative to commercial tools. Many companies and developers from OSS communities make a collaborative effort to improve the tools. Prior to this study, we developed an empirically based theory for companies' strategic choices on the development of these tools, based on empirical observations in the telecom domain. **Aim:** The aim of this study is to validate the theory of openness for tools in software engineering, in another domain, automotive. Specifically, we validated the theory propositions and mapped the case companies onto the model of openness. **Method:** We run focus groups in two automotive companies, collecting data in a survey and follow-up discussions. We used the repertory grid technique to analyze the survey responses, in combination with qualitative data from the focus group, to validate the propositions. **Results:** Openness of tools has the potential to reduce development costs and time, and may lead to process and product innovation. This study confirms three out of five theory propositions, on cost and time reduction, and the complementary role of open tools. One proposition was not possible to validate due to lack of investment in OSS tools communities by both companies. However, our findings extend the fifth proposition to require management being involved for both the proactive and reactive strategy. Further, we observe that the move towards open tools happen with a paradigm shift towards openness in the automotive domain, and lead to standardization of tools. Both companies confirm that they need legal procedures for the contribution, as well as an internal champion, driving the open tools strategy. **Conclusion:** We validated the theory, originating from the telecom domain, partially using two automotive companies. Both case companies are classified as *laggards* (reactive, cost saving) in the model of openness presented in the theory. Furthermore, we would like to have more validations studies to validate the remaining quadrants (e.g., leverage, lucrativeness and leaders).

CCS CONCEPTS

• Software and its engineering → Open source model.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

EASE '19, April 15–17, 2019, Copenhagen, Denmark

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-7145-2/19/04...\$15.00

<https://doi.org/10.1145/3319008.3319014>

KEYWORDS

Open Innovation, Theory of openness, Open tools

ACM Reference Format:

Hussan Munir, Per Runeson, and Krzysztof Wnuk. 2019. Open Tools for Software Engineering: Validation of a Theory of Openness in the Automotive Industry. In *Evaluation and Assessment in Software Engineering (EASE '19)*, April 15–17, 2019, Copenhagen, Denmark. ACM, New York, NY, USA, 10 pages. <https://doi.org/10.1145/3319008.3319014>

1 INTRODUCTION

Using and co-developing open tools with other organizations for proprietary product development is becoming an increasingly open process, thanks to the ever-growing size of Open Source Software (OSS). Open tools help companies to reduce the licensing costs for proprietary tools, and save development costs by offering flexibility in the development environment, increased turnaround speed, faster upgrades/releases and sharing the maintenance cost [22]. However, open tools require investment in the communities to gain influence on the development direction of these OSS tools. For example, the Gerrit code review tool was developed by Google and made open source in relation to Android development. Jenkins is another OSS tool co-developed by many companies (e.g., Sony Mobile, Ericsson, Intel, SAP etc.) to make the continuous integration process faster and more flexible.

Based on a literature study on OSS, an in depth case study in the telecom domain, and a follow-up cross domain survey, we derived the theory of openness [23]. To assess the validity of the theory, in this paper we apply it to a new industry domain. In focus group sessions, we validated the propositions of the theory, applying a multi-method analysis approach, using repertory grid analysis of survey data, combined with thematic analysis of qualitative data. The study validates three out of five propositions, extends one proposition, and leaves one unvalidated, as the case companies did not represent all the dimension of the theory. We also add to the knowledge, by comparing the adoption of OSS tools across industry domains.

In this paper, we give an introduction to the theory of openness in Section 2 which we aim to validate. Related work on open source tools, as well as the analysis methods used here, are presented in Section 3. Section 4 introduces the case companies, data collection procedures, validity threats, and analysis method to validate the propositions. Section 5 presents the analysis and discussion based on the repertory grid technique, using focus groups in the case companies. Finally, Section 6 wraps up the study with the conclusion.

Proactive strategy	Lucrativeness (Think tank) <i>Strategy:</i> Invest in existing communities to reduce time-to-market, spot business opportunities <i>Trigger(s):</i> Managers <i>Outcome(s):</i> Product and Process innovation <i>Level of Openness:</i> Open process – Closed outcome	Leaders (Growth through ecosystems) <i>Strategy:</i> Create new ecosystems to support brand proposition <i>Trigger(s):</i> Managers <i>Outcome(s):</i> Product Innovation <i>Level of Openness:</i> Open process – Closed outcome
	Laggards (Business as usual) <i>Strategy:</i> Reaction to paradigm shifts and cost reduction. <i>Trigger(s):</i> Managers and developers <i>Outcome(s):</i> Reduced licensing and patching cost <i>Level of Openness:</i> Open process – Open outcome	Leverage (Resource optimization) <i>Strategy:</i> Motivate developers through engaging in OSS communities i.e., look inside/outside for technological improvements <i>Trigger(s):</i> Managers and developers <i>Outcome(s):</i> Product and Process innovation <i>Level of Openness:</i> Open process – Open outcome
Reactive strategy	Cost Saving	Inspirational

Figure 1: Model of openness for tools

2 THEORY OF OPENNESS

This study is a validation step, proceeding our research efforts where we previously: 1) conducted a mapping study to identify the existing evidence on the use of open innovation in software engineering [24], 2) performed an exploratory case study at Sony about the use of open tools (e.g., Gerrit and Jenkins) in proprietary product development [22] and 3) conducted a survey in OSS tools communities (e.g., Gerrit, Jenkins, Git). Based on these findings we developed a theory of openness for tools in software engineering [23]. The scope of the theory of openness covers the use of OSS tools in the internal product development of the companies. The theory makes claims about the extraction of knowledge from OSS communities to facilitate company's internal process innovation, product innovation, reduced development time and cost. Figure 1 shows that the theory presents four categories of openness in companies with their respective focus:

- (1) Laggards – Business as usual
- (2) Leverage – Resource optimization
- (3) Lucrativeness – Acting as a think-tank
- (4) Leaders – Growth through ecosystems

Each category has different levels of openness, based on their *strategies* (proactive or reactive) in relation to *goals* (cost saving or inspirational). Typically, *laggards* respond to paradigm shifts and all strategies are reactive, aiming to reduce the development cost (i.e., for code integration). As for the *leverage* category, organizations utilize the external sources of innovation, by encouraging their internal developers to participate in various OSS tools communities, prior to internal R&D work. It adds to product and process innovation, and inspires developers to exchange ideas on discussion forums to develop competence. *Lucrativeness* refers to investing in existing OSS communities to be able to influence and steer these communities in the direction of the organizational interests. The purpose is to support internal innovation and reduce costs by investing in OSS tools communities. *Leaders* are the organizations that focus on creating new communities and ecosystems to strengthen their business model. As a part of defining the theory of openness for tools, we formulated five proposition shown in Table 1, related

Table 1: Propositions from the theory [23]

ID	Proposition
P1	Openness of tools provides opportunities to reduce development costs.
P2	Openness of tools provides opportunities to shorten the development time.
P3	Openness of tools complements internal processes and product innovation.
P4	The degree of investment in OSS communities has an affect on the outcome.
P5	Introducing a proactive strategy, in relation to openness of tools, requires conscious management involvement.

to cost reduction, development time, innovation, investment in OSS communities, and management involvement.

The main goal of this study is to validate the theory of openness for tools in software engineering, by conducting focus groups in two automotive companies, to assess if the theory is valid outside the scope of its inception. Propositions (see Table 1) from the theory of openness are validated, based on the data collected from the focus groups in the two automotive companies presented in Section 4. The validation focus is primarily on the *laggards* category, as case companies' work with tools fall in that category. The first two authors were directly involved in conducting the focus groups while the third author reviewed the collected data.

3 BACKGROUND

Open Innovation (OI) leverages internal and external knowledge exchange that may or may not be associated with monetary transactions. OI can be realized as inside-out, outside-in or coupled innovation processes [10]. OI has penetrated into several industries, as many companies discovered that their business may benefit by collaborating with OSS communities [7]. Companies apply OI using OSS tools for their internal product development. For example, our previous study [22] shows that Sony Mobile uses Jenkins and Gerrit ecosystems for knowledge flows between different companies' employees and OSS community developers, without monetary transactions. In contrast, companies like CloudBees monetize services related to the OSS tools. The stakeholders in the OSS tools communities not only share the innovation cost [6] and rewards, but also risks [22]. Companies may achieve that by revisiting their tool chains and look for open tool alternatives, as a natural and low-risk starting point in embracing openness.

The use of proprietary tools for software development leads to several challenges, e.g., delayed implementation of requirements, expensive licensing costs, inability of fixing things in-house, lack of customizability, and difficulty in finding solutions that meet current needs [22]. Companies use OSS tools communities to deal with several of these challenges [22]. However, in order to utilize open source tools communities for internal product development, companies need to invest their internal resources to use or contribute to OSS tools communities and have contribution strategies for it. The contribution strategies guides companies when to contribute and when to conceal in relation to their business model. The proposed

theory of openness helps companies to choose the right strategy while working with OSS tools communities.

Theories offer common conceptual frameworks to summarize, condense and accumulate knowledge. The software engineering research community have raised concerns on the lack of theories in the field [2, 4, 9, 14, 29–31] and pointed to the limited work about SE theories. Specifically, we found no theory building attempts in our literature review [24]. We used Sjøberg et al's method to design the theory of openness [30] and here aim to validate the theory by collecting data from focus group surveys and discussions, and applying the repertory grid technique to the data.

Repertory grid is a technique for identifying the ways that a person interprets or gives meaning to his or her experience, being used in software engineering already [8]. Moynihan's exploratory study used the repertory grid technique to identify the prevalent risks factors in software development projects [20, 21]. Lee and Truex [16] used repertory grid technique to explore whether or not training in information system development methods improves students' cognitive structures (e.g., focused, tight thinking). Baddoo's exploratory study [1] collected data from 13 collaborating software companies to investigate how groups of staff in three designated roles (i.e., developers, project managers and senior managers) saw themselves and the other roles as being involved in software process improvement. Ghazi et al. [11] proposed a decision support method, using the repertory grid technique, which aids practitioners to choose the right level of exploratory testing (i.e., freestyle testing, a high degree of exploration, medium degree of exploration, a low degree of exploration and scripted) in their context.

4 RESEARCH METHODOLOGY

In this section, we describe the case companies, data collection and analysis methods, and our actions taken to improve the validity of the study.

4.1 Case A

Company A is a major automotive industry manufacturer of heavy trucks and buses. It also manufactures diesel engines for heavy vehicles as well as marine and general industrial applications. The company develops its software services for fleet management and is considering transport as a service in their business model.

Albeit their internally developed software is not monetized yet, the company has started seeing the software as an important strategic area for its core business. Therefore, the company has a high-level strategy of utilizing OSS tools to build competence in this area. The company uses Jenkins in the development of their fleet management system.

The participants involved in the workshop were a tools manager, product owners and technical teams leads, and tools engineers.

4.2 Case B

Company B is one of the well-known global brands in automotive industry, heading towards an all-electric future. The increased use of software in the cars makes embedded software development a cutting-edge area for the company's business model. In order to facilitate the software development, the company buys tools from suppliers to facilitate the software development. However, it

is difficult to buy a solution and fit it into the existing tool chain. Therefore, the company is shifting towards OSS tools to achieve standardization in all teams.

Tools like Jenkins, Gerrit and Git are already utilized in the development process. The tool especially discussed in this study is an internally developed tool named *Awesome framework*. It is based on Django¹ which is a Python Web framework that encourages rapid development and clean, pragmatic design. It is an automated testing framework which enables test automation for hardware-in-the-loop, software-in-the-loop, and model-in-the-loop test environments to be able to fit these into a continuous integration chain. It provides an integrated development environment for developing such tests at component and system level.

The participants of the workshop in the case company include a tools manager, product owners, business analysts, technical teams leads, and tools engineers.

4.3 Data collection and analysis methods

We arranged focus groups [27] at the two companies. There were 12 and 10 participants in the focus groups at companies A and B, respectively. We firstly collected quantitative data in an online survey, and secondly qualitative data in the subsequent discussion of the survey outcome.

We used the repertory grid technique [15] to analyze and validate the theory of openness. Kelly proposed the personal construct theory (PCT) and the associated repertory grid technique in the 1950's to elicit and analyze personal constructs [15]. The idea behind the theory is that participants have their own views of the world, based on their observation of surroundings. Therefore, each participant builds his or her own conceptual framework which leads to different opinions about the same problem. Participants constantly observe and react to their understanding of the surroundings. Consequently, participants reform their personal theories and assumptions [8].

The *grid* is composed of three basic concepts: 1) elements, 2) constructs, and 3) ratings. *Elements* refer to individual aspects or objects of a topic, which participants try to understand. A *construct* is made of two contrasting concepts that are equally weighted on a bipolar scale. *Ratings* are the assessment of the participants on the bipolar scale.

There are two essential ways to select grid elements: a) elicit elements from participants, b) provide participants with elements. We provided the participants with elements and constructs derived from the theory of openness [23]. Each element was rated by the participants against each construct in the focus groups in an individual survey, and then discussed in plenum. To further support our choice of method, a number of studies have taken this approach [16, 25, 26, 32] by providing elements and constructs to participants for the ratings. The description of elements, constructs and their ratings are as follow:

Elements derived from theory of openness. This study uses four elements from the strategies and goals of the theory of openness, namely:

- (1) Inspiration
- (2) Reactive

¹<https://www.djangoproject.com/>

- (3) Cost saving
- (4) Proactive

The elements can be seen as columns in Figure 2 a and b for companies A and B, respectively.

Constructs derived from the theory of openness. We designed constructs from the theory of openness by creating the contrasting poles of each construct, see A1, A2 ... in Table 3. Each construct is used to underpin one of the propositions (P1–P5, see Table 1), as shown in the rightmost column of Table 3. We derived these constructs from the synthesis of literature in the theory of openness, during the theory development [23]. The constructs assess the company's perspective on creating new communities to facilitate internal product development by reducing the development, development cost, new creation of roles, approval from top management for OSS tools adoption and access to skilled workforce etc.

Ratings. Kelly's original grid used only a two-point scale, but the majority of the studies in software engineering use odd number scales to provide a clear reference point for the two poles. However, there is a concern that the participants cannot make a clear distinction between longer scales [8]. Therefore, we used a three-point Likert scale, see Table 2, for the participants to take a clear stance between the two poles (as defined in Table 3) while rating each element against each construct.

Table 2: Scale used for the ratings

Elements	Scale		
	1	2	3
<i>Inspiration</i>	Inspirational	Somewhat	Not Inspirational
<i>Reactive</i>	Reactive	Somewhat	Not Reactive
<i>Cost saving</i>	Cost saving	Somewhat	Not Cost saving
<i>Proactive</i>	Proactive	Somewhat	Not Proactive

In this step, focus group meetings were held, separately for each company, where elements were rated against each construct, using a survey². First, the focus group participants were given an introduction to constructs and elements to develop a common understanding. Second, participants from company A picked *Jenkins* and participants from company B selected their internal tool called *Awesome framework* for the focus groups. Third, a survey link was distributed among all the participants to rate each element against the constructs, based on the selected tools from their internal development environment. Fourth, each element was rated against each construct using the scale in Table 2.

The outcome of the ranking, after consolidating the rankings in plenum is shown in Figure 2a and 2b for company A and B, respectively. The outcomes show, for example, that the *construct* on adopting OSS tools need approval from top management (A8) is rated as not inspirational (3), reactive (1), not cost saving (3) and not proactive (3) by the participants in both companies. Similarly, the *construct* on use of OSS tools helping to reduce the development time (A11) is rated as a cost saving (1) and proactive (1). Lastly,

²<https://www.surveymonkey.com/r/Pickatool>

Figure 2: Ratings of elements in relation to constructs from all participants of focus group, using the three point Likert scale presented in Table 2.

we held a discussion among participants based on the ratings to further explore their ratings.

A construct could be reactive and proactive simultaneously. As an example, A4 is ranked 1 on all elements. We interpret this as the construct is valid for both reactive and proactive strategies. For example, the first company investing in an OSS tool is proactive, while subsequent companies can be reactive in their approach to adopt the OSS tool. Figure 2 a and b show ratings from company A and B, respectively. We used vote counting for ratings of elements and constructs in the scale shown in Table 2. The values shown in Figure 2 shows the value selected by a majority of the workshop participants. The borderline cases (i.e., no clear majority in vote counting) were discussed to reach a consensus value from all participants. The discussion part was recorded and transcribed to further explore the rationales for the participant's ratings.

After the session, the ratings were analyzed using Principal Component Analysis (PCA) to identify the orthogonal components in the data. We compare the resulting components with the original dimensions, defined by the elements (*inspiration* vs. *cost saving* and *reactive* vs. *proactive*). However, we perform that comparison qualitatively, as we cannot guarantee that the conditions for PCA are fulfilled. A repertory grid is usually two-set similarity data and PCA requires at least ordinal levels of measurement in the data to be applied [?]. On the other hand, the literature shows many instances of using PCA on Likert scale data from repertory grids [? ? ?]. Therefore, we used WebGrid³ based on Slater's work [? ?], which

³<http://webgrid.uvic.ca/>

Table 3: Constructs and contrasting poles (A1–A17) derived from theory of openness, mapped to the propositions (P1–P5)

ID	Similarity Pole (+)	Contrast Pole (-)	Prop.
A1	Creating new open source communities to facilitate internal product development	Creating new open source communities not required to facilitate internal product development	P5
A2	Creating new open source communities to explore emerging technologies	Creating new open source communities not required to explore emerging technologies	P4
A3	Leveraging OSS to increase market share	Leveraging OSS to increase market share not required	P3
A4	Using existing open source communities to identify the emerging technologies	Using existing open source communities to identify the emerging technologies not required	P4
A5	Using existing open source communities for technological improvements	Using existing open source communities for technological improvements not required	P4
A6	Investing in the open source communities is required to steer the community's development towards organizational benefits	Investing in the open source communities is not required to steer the community's development towards organizational benefits	P4
A7	Defining new work roles and teams to work with open source communities	Defining new work roles and teams not required to work with open source communities	P3
A8	Adopting OSS tools need approval from top management	Adopting OSS tools does not need approval from top management	P3
A9	Contributing to open source communities need approval from top management	Contributing to open source communities does not need approval from top management	P3
A10	Educating the organization about OSS culture and ways of working	Educating the organization about OSS culture and ways of working not required	P3
A11	Using OSS tools helps to reduce development time	Using OSS tools does not reduce development time	P2
A12	Working with open source communities gives access to free labor	Working with open source communities does not give access to free labor	P1
A13	Using OSS tools keeps developers updated with best tools practices	Using OSS tools does not keep developers updated with best tools practices	P3
A14	OSS tools reduces licensing cost in relation to proprietary tools	OSS tools increases licensing cost in relation to proprietary tools	P1
A15	The benefits of engaging in open source communities outweighs the risk of losing Intellectual property rights	The benefits of engaging in open source communities does not outweighs the risk of losing Intellectual property rights	P2
A16	Engaging in open source communities provides fun ways of working for developers	Engaging in open source communities provides boring ways of working for developers	P3
A17	Working with open source communities gives access to external knowledge from communities	Working with open source communities does not gives access to external knowledge from communities	P5

helps to perform the PCA on repertory grid data. Furthermore, the study uses qualitative data in combination with PCA to triangulate the analysis.

Outcome. The outcome of this step (ratings) is Figure 2, which shows the consensus rating of all participants in the focus groups from the two companies (A and B). We also perform a PCA to identify components in the data, as presented in Figure 3.

4.4 Validity threats

We discuss validity threats with respect to internal validity, external validity, construct validity, and conclusion validity [27].

4.4.1 Internal validity. Internal validity threats relate to confounding factors that may have influenced the outcome without the researcher's knowledge. One possible threat is that the participants

in companies A and B may misunderstand the constructs and elements. Therefore, an introduction presentation was given to participants before rating each element against constructs. Constructs and elements were exemplified during the introductory presentation session. Furthermore, we also had a follow-up discussion based on the ratings from the participants to clarify misunderstandings. Another threat to internal validity is that the participants might have difficulties in interpreting the scale for the ratings. To minimize the risk of misinterpretation, the examples were given in the introduction prior to rating of elements and constructs. Finally, the use of PCA has been questioned to be used on the Likert scale data. However, the repertory grid technique has made it possible to use PCA for Likert scale data and the literature shows many instances of using PCA on Likert scale data from repertory grid [? ? ?]. To

Figure 3: Principal component analysis (PCA) of ratings from focus groups.

further reduce this internal validity threat, we triangulated the PCA analysis with the qualitative data analysis.

4.4.2 External validity. The primary aim of the study is to validate the theory in another domain, where open software engineering tools may be applicable. Due to the advent of autonomous vehicles, both companies are experiencing a paradigm shift with the ever increasing adoption of software in vehicles to stay on the competitive edge, as well as using open tools in their engineering processes. Validating the theory in the automotive industry with software leading the cutting edge innovation, gives more weight to the external validity of the theory. More studies are intended to perform in the software industry to improve the external validity of the theory, as well as monitoring the development over time with respect to open software strategies and practices.

4.4.3 Construct validity. Both constructs and elements in the theory are relevant to discuss in relation to construct validity. Neither of the case companies comes from an OSS development and community participation background and therefore, does not have established procedures to work with open tools communities (i.e., they operate in the laggards category). As a consequence, neither company has a well-defined procedure to map all the constructs of the theory, i.e., those related to the leverage, lucrativeness and leaders categories. As a consequence, the theory validation is only partial.

4.4.4 Conclusion validity. First, participants in the focus groups may have tried guessing the propositions during the introduction presentation, which introduces a threat to the conclusion validity. This threat was mitigated by keeping elements and constructs as

discrete as possible. Second, a researcher's bias while interpreting and recording focus groups may threaten the conclusions. To reduce this threat, multiple researchers were involved in the focus groups (observer triangulation). Furthermore, focus groups were recorded, transcribed, and the results were validated by multiple researchers.

5 RESULTS AND DISCUSSION

The outcomes from the repertory grid technique are the grid and a principal component analysis (PCA). PCA is a data reduction technique, that transforms the dimensions of the data into orthogonal dimensions, thereby reducing the number of dimensions in the data. Further, we have the qualitative data from the focus group meetings, which are analyzed together with the quantitative data.

Figure 2 shows the formation of the repertory grid for companies A and B, based on the ratings from the participants in the focus group meetings. The columns show the contrasting poles of *constructs* derived from the theory of openness, mentioned in Table 3. The rows show the *elements* derived from the theory of openness. Each element was rated by the participants in the workshop in relation to constructs, using the scale presented in Table 2. The first row in Figure 2a, for example, means that the participants agreed on ranking construct A1 inspirational (1), reactive (1), somewhat cost saving (2), and proactive (1).

Figure 3 presents the spatial representation of the elements and constructs, mapped onto the two most dominating dimensions of the PCA. The PCA results in three major components, representing 70.0%, 24.4%, and 5.5% of the variance for company A, and 59.4%, 26.3%, and 14.3% of the variance for company B. The two first components correspond to axis 1 and 2 in Figures 3a and b, respectively.

Below we discuss the interpretation of the constructs' and elements' positions in this two-dimensional space.

5.1 Relation between strategies

The PCA shows that *proactive* and *inspiration* in companies A and B are spatially closer to each other than the *cost saving* and *reactive* poles. We interpret this as the companies connect the inspirational goal closer to a proactive strategy. However, the *cost saving* and *reactive* strategies in company A seem to have a higher variance in contrast to company B (larger distance between them). It can be explained by both companies' current strategy of buying OSS enterprise solutions instead of actively using or contributing to OSS tools communities, because they believe it gives them more control over the contracts as the companies think in the old way of subcontracting. The focus group discussion showed that company A was not able to distinguish between OSS tools, and tools provided by suppliers based on OSS. The reason for this may be the lack of OSS understanding and competence at company A, since the company does not come from an OSS development background. One of the participants stated, *"one of the major reasons for not using OSS tools is the lack of competence and understanding. We do not know how to protect our intellectual property rights. We tend to go for buying solutions when there are intellectual property rights involved."*

In summary, the PCA indicates that the companies distinguish between the reactive vs. proactive and inspirational dimension (close to component 1) and the cost saving dimension (close to dimension 2). As a consequence, cost saving and inspirational is not seen as contrasting poles, but can be complementary.

5.2 Validation of propositions

We validate the five propositions (see Table 1) from the theory of openness [23] in the light of the PCA and discussions in the focus groups. The constructs referred to (A1–A17) are presented in Table 3, mapped to the propositions (P1–P5).

For proposition (P1) about the reduction in development cost, the PCA shows that access to free labor (A12+) and reduced licensing cost (A14+) is closer to cost saving in case of company B as opposed to company A (see Figures 3a and 3b). One possible explanation is that Company A does not see faster automated build servers as a competitive edge for its internal developed software. Furthermore, none of its software is provided as a service to the customers yet. However, they started seeing value in services and connections, where licensing costs might come into play. One of the participants stated that *"the percentage of money spent on licenses is actually not that high at the moment. However, if we start producing hundreds of micro-services with each of them need their own database, buying Oracle licenses might become really expensive."* On the other hand, it is not uncommon for company B to buy tools and these tools can be expensive. One of the participants in company B stated that *"customized tools provided by big vendors like MathWorks for the company are really expensive. Therefore, we have a window of opportunity to update our environment with open tools."* In conclusion, openness of tools has the potential to reduce the development cost, which validates our proposition P1.

Regarding whether openness shortens the development time (P2), PCA for both companies in Figures 3a and 3b shows that the benefits of engaging in OSS tools communities outweigh the risk of losing IPRs (A15+), and reduced development time (A11+) is spatially closer to cost saving. This indicates that participants think that engaging in open tool communities outweighs the risk of losing intellectual property rights, although the lack of maintenance support in the company is a point of concern for developers. Software developers in company B think that using open tools speed up their development in relation to seeking permission to buy a tool. One participant stated that *"regarding speed as a developer, it takes a few minutes to download an open framework rather than seeking an approval from the manager to buy a tool, which takes forever."* However, both companies are careful in selecting and integrating these open tools to make sure that these tools won't die over time. There is a risk that software developers might not get the required funding and enough management backing to maintain these open tools internally if the community dies. A participant stated that *"we are dependent on some really large Python projects that we know won't die anytime soon."* Therefore, the proposition P2 about open tools may reduce the development is confirmed by the data.

In relation to process and product innovation (P3), it should be mentioned that we have earlier observed that process innovation leads to product innovation [17]. However, neither case company considers tools leading to innovation in the development of their core products. In particular, the tool development team of company B does not get enough funding and management support, although other departments in the company need tool support to facilitate product innovation in the core products. This is a general problem with project focused organizations, where the budget is allocated for the given project only, with limited consideration for long-term investment. The tools department does not have an allocated budget for the development of tools and the maintenance of these to achieve standardization. One of the participants in company B stated *"when we buy tools, it is part of the R&D budget and there is no designated budget for maintenance. For example, if we get the funding for developing an internal tool like Awesome framework, there is no allocation of budget for the maintenance of that tool."* On the other hand, both companies identify the need for process innovation by creating new roles [18] to deal with the challenges of open tools platform. Consequently, there have been initiatives in both companies to create new open innovation roles, although the companies score differently with respect to its value (A17+/-). For example, a participant of company A mentioned that *"the company has actually employed a new role of open source cloud manager to create procedures and guidelines regarding how to be active in an open space."* Furthermore, a participant in company B stated that *"there is a new initiative called Nordic foundation to spread OSS usage and with the already formed legal framework among Nordic companies. It could be a fast track for us but it needs to be championed by R&D in order to reap benefits."* Therefore, the proposition P3 related to the openness of tools complements internal process and product innovation is confirmed by the data.

In both companies, the PCA shows that adopting (A8+) and contributing to OSS tools (A9+) are spatially close to the *reactive* strategy, meaning that participants consider taking approval from top management in either adopting or contributing to open tools

communities as a reactive strategy. One possible explanation for seeking approval is that neither company has a legal framework to work with open tools communities. One of the participants in company A stated that *“we have a top-level statement saying Open Source First but what does that exactly mean? It does not state exactly what, why and how should we go open and how these decisions are supported.”* Similarly, a participant in company B also stated that *“we do not have any internal procedure or legal framework to create new or work with the existing open tools communities.”* Therefore, **P4** about the relation between investment in OSS tools communities and its impact, is neither confirmed nor rejected by the data, since both case companies lack internal procedures to invest their internal resources in open tools communities.

Finally, proposition **P5**, about the role of management in Table 1, needs to be extended since not only the proactive strategy but also the reactive strategy, requires management involvement, as software developers need a legal framework to work with open tools.

5.3 Comparison across industry domains

Actors in the automotive industry see software services as a new emerging area in terms of software-enabled transportation services [5, 12]. A modern, sophisticated car with the cloud connected infotainment systems can have up to 100 million lines of code [19]. This can be related to our previous case study in the telecom domain [22] in which a natural drive for saving money and reducing ownership costs was to create industry wide standard platforms (e.g., Andriod, Jenkins, Gerrit). Google capitalized on the first mover advantage by offering Android for free and it got widely adopted. Initially, Sony underestimated the value in services and apps. So everyone let Google develop Android and occupy services such as Gmail, drive, gaming, camera etc. Now Android is provided as an empty shell and vendors just provide devices to use services for money.

Similar risks may apply to the automotive industry, if the companies do not establish new OSS communities and ecosystems. Companies may provide vehicles to run the services and also make money on the services along with the selling of vehicles. Software is becoming more and more cutting edge component in the development of vehicles due to the advent of autonomous cars which leads to need of OSS tools for better continuous integration of code for these autonomous cars.

Both case companies are using OSS tools for development of their proprietary products. Therefore, we looked for the application of lessons learnt from our previous case [22]. That study explored Sony Mobile’s transition from closed tools to open tool platforms [22], which constitutes one of the cases for the empirical grounding of the theory of openness for tools. The two studied companies A and B have started looking into OSS tools due to their increased need of software in autonomous vehicles makes a case to compare this transition to Sony Mobile. In our analysis, we have found three similar patterns in companies A and B.

First, there was a paradigm shift when Sony Mobile moved from Windows to Linux (i.e., development tools), which in turn required Sony to move from proprietary tools (e.g. ElectricCommander) to open tools platforms (e.g., Gerrit, Jenkins, Git). The complex integration tool chain triggered the move towards open tools platforms

and it resulted in increased development speed and reduced time to market. Both companies A and B are experiencing a paradigm shift in the automotive industry, where software is becoming the key area for the development of vehicles, such as autonomous driving features and providing transportation as a service. Both companies are very reactive in their approach when opting for open tools. A participant in the company said, *“the change generally depends on how painful the situation is. A few years ago, we had 10 different teams working with an internal platform using different tool chains and had a great difficulty in glueing the tool chain together. We hired an external company that glued our tools chain together with massive scripts using an open tool solution.”* Therefore, the open tools chosen in a proactive way of standardizing the tool chain in all teams, may play an important role in the development of software-intensive vehicles.

Second, when Sony Mobile realized that they lacked legal procedures to work with open tools communities, they created an internal open tools department consisting of legal experts, developers and managers to create a well-established procedure for using or contributing to open tools platforms. Companies A and B may follow the same steps by creating an internal process to educate the developers and develop competence to work with open tools communities. As one of the participants in company A said, *“we lack competence and streamlined decisions on how to work with open tools communities.”* Another participant in company B said that *“we need improvements in the areas of governance, training and policies.”*

Third, Sony Mobile had an internal champion to derive the whole open tool-chain movement with the support of software developers and legal team. This played an important role in convincing the top management to support and fund open tools platforms, however, it took three years for Sony Mobile to make it. The current case companies may need a similar, internal champion to act as an interface between the legal team, software developers, and the top management.

Key takeaway. Companies A and B may choose a centralized proactive strategy similar to Sony Mobile (reactive in that case) to make internal legal policies and education for employees to meet the ever-growing size of software in vehicles. We explicitly took the knowledge from the telecommunications domain and apply it to automotive companies.

5.4 Implications for the Model of openness

Figure 4 shows the mapping of tools in companies A and B on the model of openness for tools [23]. Both tool-company pairs qualify as *laggards* since they have no internal procedures to facilitate developers to contribute to OSS tools communities. Both companies react when the integration of tools becomes painful for developers, rather than standardize the ways of working in the companies. Also for laggards, we noticed that it is important to have the management support and approval.

In case company B, the internally developed tool named *Awe-some framework* is meant to be shared among different partners of the company and it is used in core product development teams. However, the development does not have enough resources to support core product development teams who have requirements and need support. This suggests the importance of the development of

Proactive strategy	Lucrativeness (Think tank)	Leaders (Growth through ecosystems) Jenkins (Sony Mobile)
	Laggards (Business as usual) Jenkins (Company A) Awesome framework (Company B)	Leverage (Resource optimization)
Reactive strategy		
	Cost Saving	Inspirational

Figure 4: Mapping of tool-company pairs on the model of openness for tools.

the tools, while lack of funding and resources hampers the work of core product development teams who are dependent on the tool. An alternate way of gaining more resources is to create an OSS community around the Awesome framework and build an ecosystem around the community to share the development cost with other stakeholders in the community.

In our earlier work [22], Sony Mobile went through the same tools transition (e.g., Jenkins) as companies A and B are going through. Sony Mobile started as a *laggard* in relation to Jenkins with its complex continuous integration tools chain, before actively using and contributing to Jenkins community. However, this transition required Sony Mobile to create a tools department and a legal framework to utilize the OSS tools communities.

In the two studied case companies, company B is found to be more in need of this transition due to their complex integration tool chain, compared to company A. Therefore, both companies will need processes and legal frameworks for developers, to encourage them to take more initiative towards open tools in order to standardize the ways of working. Currently, in company B, all teams have their internally developed tools in different technologies (e.g., C#, Python, Java etc.) which leads to loss of resources on writing more glue code to integrate the tool chain. A participant from company B stated, “I think we have to change our way of thinking and we never share anything because there is always some kind of minor internal change that drives our testing and we do not want to share that. However, 80% of the rest is the same that can be shared. At the moment, it is all or nothing and we need to calm down a little bit and share the common parts at least.” This is well in line with the triggers of Sony Mobile when developers started looking for OSS tools like Jenkins, to stop relying on the tool suppliers and fix their internal code integration problems by using and contributing back Jenkins community. The contribution to the community allowed them to share the development cost with the community and also received bug fixes and new features from the community. Moreover, contributions to Jenkins community enabled to become the leaders to steer the community towards their own needs.

Once the processes for contribution and using OSS tools communities are in place, both companies may consider creating new

open tools ecosystems (*leaders*) by making the code open, to attract other manufactures in the industry with the same needs. One of the participants stated, “I think it is a question of survival in the near future and we do not have time to deal with the increase in the number of variants. It will be nice to replace the closed source tools with the open ones so that we can actually fix tools without glue scripts.”

It is to be noted that the data collected from the case companies does not address or validate the leverage and lucrativeness category in Fig 4. The studied companies only validates the laggards category, thus the theory required more validation studies to validate other categories as well.

Key takeaway. The model of openness can help companies to facilitate their internal product development by engaging their developers in OSS tools communities and eventually be able to build new communities (*leaders*) by choosing a reactive or proactive strategy. Furthermore, both companies may learn from Sony Mobile’s transition from *laggards* to *leaders* by innovating their process in terms of creating a legal framework for the contributions to OSS tools communities. However, *laggards* also require management to look at tools as a long term standardization of processes perspective rather than a project based approach.

6 CONCLUSIONS

This study aims to validate the propositions derived from the theory of openness for software engineering tools, mapping it from the telecom domain to the automotive. We use the repertory grid technique to validate the five propositions in the theory of openness and present a comparison with our prior work.

The data supports **P1** about the openness of tools may reduce the development cost, by relying on standard OSS tools instead of buying customized tool with expensive licenses from tool suppliers. Similarly, **P2** is supported, that open tools may reduce development time since they do not have to go through the formal budget approval process from top management to buy a tool from a supplier. For proposition **P3**, open tools may leads to process innovation in terms of creating the new work roles to work with the open communities and product innovation by giving a flexible and faster development environment to the development teams fore core product development. However, **P4** about the investment of financial and human resources on OSS communities and its impact on the development is neither confirmed nor rejected due to lack of investment in the OSS tools communities in the case companies. Finally, **P5** requires extension to P5, meaning that active management involvement not only required for proactive strategy but also for the reactive strategy.

Furthermore, the two studied companies only validate the laggards category. Both case companies lack internal procedures to work with the open tools communities. Furthermore, we extended our findings from telecommunications to the automotive domain. Both case companies are currently experiencing a paradigm shift in the automotive industry with the growing size of software due to the advent of autonomous driving. This may require the management to rethink and revise their strategy to utilize OSS tools and their communities. The strategy should address creating a new role which works as an interface between the legal department, software developers and management. Furthermore, it may entail

creating processes regarding how and when to use, contribute or create new open tools platforms for company's benefits. As for future work, more validation studies need to be conducted to validate the propositions derived from the theory of openness and what organizational changes are needed. Furthermore, the impact of openness on business model also needs to be investigated.

ACKNOWLEDGMENTS

We thank the company participants for sharing their insights in the focus groups. This work was partly funded by the EASE industrial excellence center.

REFERENCES

- [1] N. Baddoo and T. Hall. Practitioner roles in software process improvement: an analysis using grid technique. *Software Process: Improvement and Practice*, 7(1):17–31, 2002.
- [2] V. R. Basili, F. Shull, and F. Lanubile. Building knowledge through families of experiments. *IEEE Transactions on Software Engineering*, 25(4):456–473, 1999.
- [3] R. C. Bell. Analytic issues in the use of repertory grid technique. *Advances in personal construct psychology*, 1:25–48, 1990.
- [4] E. Bjarnason, K. Smolander, E. Engström, and P. Runeson. A theory of distances in software engineering. *Information and Software Technology*, 70:204–219, 2016.
- [5] M. Broy. Challenges in automotive software engineering. In *Proceedings of the 28th International Conference on Software Engineering*, ICSE '06, pages 33–42, New York, NY, USA, 2006. ACM.
- [6] H. Chesbrough. Why companies should have open business models. *MIT Sloan management review*, 48(2), 2012.
- [7] H. Chesbrough, W. Vanhaverbeke, and J. West, editors. *New Frontiers in Open Innovation*. Oxford University Press, Nov. 2014.
- [8] H. M. Edwards, S. McDonald, and S. M. Young. The repertory grid technique: Its place in empirical software engineering research. *Information and Software Technology*, 51(4):785–798, 2009.
- [9] A. Endres and H. D. Rombach. *A handbook of software and systems engineering: Empirical observations, laws, and theories*. Pearson Education, 2003.
- [10] E. Enkel, O. Gassmann, and H. Chesbrough. Open R&D and open innovation: exploring the phenomenon. *R&D Management*, 39(4):311–316, 2009.
- [11] A. N. Ghazi, K. Petersen, C. Wohlin, and E. Bjarnason. A decision support method for recommending degrees of exploration in exploratory testing. *arXiv preprint arXiv:1704.00994*, 2017.
- [12] A. Haghighatkhah, A. Banijamali, O.-P. Pakanen, M. Oivo, and P. Kuvaja. Automotive software engineering: A systematic mapping study. *Journal of Systems and Software*, 128:25–55, 2017.
- [13] T. Hall, D. Wilson, and N. Baddoo. Towards implementing successful software inspections. In *null*, page 127. IEEE, 2000.
- [14] J. D. Herbsleb and A. Mockus. Formulation and preliminary test of an empirical theory of coordination in software engineering. In *ACM SIGSOFT Software Engineering Notes*, volume 28, pages 138–137. ACM, 2003.
- [15] G. A. Kelly. The psychology of personal constructs. i. a theory of personality. ii. clinical diagnosis and psychotherapy. 1955.
- [16] J. Lee and D. P. Truex. Exploring the impact of formal training in isd methods on the cognitive structure of novice information systems developers. *Information Systems Journal*, 10(4):347–367, 2000.
- [17] J. Linäker, H. Munir, P. Runeson, B. Regnell, and C. Schrewelius. A Survey on the Perception of Innovation in a Large Product-focused Software Organization. *6th International Conference on Software Business - ICSOB*, 2015.
- [18] C.-E. Mols, K. Wnuk, and J. Linäker. The open source officer role – experiences. In F. Balaguer, R. Di Cosmo, A. Garrido, F. Kon, G. Robles, and S. Zacchioli, editors, *Open Source Systems: Towards Robust Practices*, pages 55–59, Cham, 2017. Springer International Publishing.
- [19] J. Mössinger. Software in automotive systems. *IEEE software*, 27(2), 2010.
- [20] T. Moynihan. An inventory of personal constructs for information systems project risk researchers. *Journal of information technology*, 11(4):359–371, 1996.
- [21] T. Moynihan. How experienced project managers assess risk. *IEEE software*, 14(3):35–41, 1997.
- [22] H. Munir, J. Linäker, K. Wnuk, P. Runeson, and B. Regnell. Open innovation using open source tools: a case study at Sony Mobile. *Empirical Software Engineering*, pages 1–38, 2017.
- [23] H. Munir, P. Runeson, and K. Wnuk. A theory of openness for software engineering tools in software organizations. *Information and Software Technology*, 97:26 – 45, 2018.
- [24] H. Munir, K. Wnuk, and P. Runeson. Open innovation in software engineering: a systematic mapping study. *Empirical Software Engineering*, 21(2):684–723, Apr 2016.
- [25] N. Niu and S. Easterbrook. Discovering aspects in requirements with repertory grid. In *Proceedings of the 2006 international workshop on Early aspects at ICSE*, pages 35–42. ACM, 2006.
- [26] G. J. Phythian and M. King. Developing an expert support system for tender enquiry evaluation: A case study. *European Journal of Operational Research*, 56(1):15–29, 1992.
- [27] P. Runeson, M. Höst, A. Rainer, and B. Regnell. *Case Study Research in Software Engineering: Guidelines and Examples*. John Wiley & Sons, Mar. 2012.
- [28] P. Runeson, H. Munir, and K. Wnuk. It is more blessed to give than to receive—open software tools enable open innovation. *Tiny Transactions on Computer Science*, 4, 2015.
- [29] C. Sauer, D. R. Jeffery, L. Land, and P. Yetton. The effectiveness of software development technical reviews: A behaviorally motivated program of research. *IEEE Transactions on Software Engineering*, 26(1):1–14, 2000.
- [30] D. I. Sjöberg, T. Dybå, B. C. Anda, and J. E. Hannay. Building theories in software engineering. In *Guide to advanced empirical software engineering*, pages 312–336. Springer, 2008.
- [31] W. F. Tichy. Should computer scientists experiment more? *Computer*, 31(5):32–40, 1998.
- [32] S. M. Young, H. M. Edwards, S. McDonald, and J. B. Thompson. Personality characteristics in an xp team: a repertory grid study. In *ACM SIGSOFT Software Engineering Notes*, volume 30, pages 1–7. ACM, 2005.