A Community Strategy Framework – How to obtain influence on requirements in meritocratic open source software communities?

J. Linåker [a],[*], B. Regnell [a], D. Damian [b]

[a] *Lund University, Box 118, Lund 221 00, Sweden*
[b] *University of Victoria, PO Box 1700, STN CSC, Victoria, BC V8W 2Y2, Canada*

A B S T R A C T

Context: In the Requirements Engineering (RE) process of an Open Source Software (OSS) community, an involved firm is a stakeholder among many. Conflicting agendas may create miss-alignment with the firm's internal requirements strategy. In communities with meritocratic governance or with aspects thereof, a firm has the opportunity to affect the RE process in line with their own agenda by gaining influence through active and symbiotic engagements.

Objective: The focus of this study has been to identify what aspects that firms should consider when they assess their need of influencing the RE process in an OSS community, as well as what engagement practices that should be considered in order to gain this influence.

Method: Using a design science approach, 21 interviews with 18 industry professionals from 12 different software-intensive firms were conducted to explore, design and validate an artifact for the problem context.

Results: A Community Strategy Framework (CSF) is presented to help firms create community strategies that describe if and why they need influence on the RE process in a specific (meritocratic) OSS community, and how the firm could gain it. The framework consists of aspects and engagement practices. The aspects help determine how important an OSS project and its community is from business and technical perspectives. A community perspective is used when considering the feasibility and potential in gaining influence. The engagement practices are intended as a tool-box for how a firm can engage with a community in order to build influence needed.

Conclusion: It is concluded from interview-based validation that the proposed CSF may provide support for firms in creating and tailoring community strategies and help them to focus resources on communities that matter and gain the influence needed on their respective RE processes.

1. Introduction

Open Source Software (OSS) is for many firms today a fundamental building block for creating, delivering and supporting their product and service offerings, or internal operations [1,2]. The development and maintenance of an OSS project are performed within a software ecosystem [3], often referred to as a community. The members of a community consist of stakeholders of the OSS project, i.e., *"...person[s] or organization[s] who influences a system's requirements or who [are] impacted by that system"* [4]. In this case, "a system" refers to the OSS project. To a firm involved in an OSS community, the Requirements Engineering (RE) process in the community is an external process where the firm is no longer the central authority, in contrast to traditional market-driven RE [5]. Instead, the firm is a stakeholder among many which may introduce conflicting agendas from other stakeholders [6–8], and a new type of power and politics than the firm might be used to [9]. Consequences

may include a lack of control over what requirements that are implemented, and miss-alignment with the firm's internal RE process [1,10]. A firm who wish to affect the RE process according to their agenda may, therefore, have to build up an influence within the community [7].

With influence, we refer to the Merriam–Webster dictionary [1] which defines it as *"the power to change or affect someone or something"*. In our context, this relates to the power of a firm to change or affect a requirement of interest in an OSS community, for example, how a requirement is specified, prioritized, and realized, both short-term in release-planning, and long-term on the road-map [11–13]. In OSS communities with a meritocratic governance structure [14,15], either in part or in full [16], influence is gained by proving merit and earning trust and status within the community [17]. What merit constitutes depends on the

---

[1] http://www.merriam-webster.com/dictionary/influence.

* Corresponding author.
 E-mail addresses: johan.linaker@cs.lth.se (J. Linåker), bjorn.regnell@cs.lth.se (B. Regnell), damian.daniela@gmail.com (D. Damian).

context [18,19], but is generally gained by building an active and symbiotic relationship with the community where a firm dedicates resources, contributes internal requirements and actively participates in the development of the OSS [2,7,20–23]. A meritocratic OSS community, therefore, offers an opportunity for the focal firm to influence the community's RE process according to the firm's own agenda while competing and collaborating with the other stakeholders in the community [23].

For a firm engaged in many communities, such investments may be costly if it is distributed over all communities. It may be that only a few communities are of such strategic importance to the firm, and are in a state where the firm needs to have an influence on their RE processes [1]. For a strategic community that is healthy, predictable and aligned with a firm's internal agenda, it may be that a high level of influence is not motivated [2]. Therefore, to optimize its resource utilization and investments where best needed, firms may have to assess how they could benefit from a specific OSS project and its community, and then if and how much influence that is required to reap these benefits [2]. To the best of our knowledge, there is no systematic approach to perform this kind of assessment, why we pose our first research question as:

RQ1 What aspects should a firm consider when assessing its need to influence the RE process in a meritocratic OSS community?

If a firm assesses that they need influence on the RE process in a meritocratic OSS community, the follow-up question is: what should their community engagement look like and how should they invest their resources to gain the influence needed? To the best of our knowledge, an overview on a software engineering level of what engagement practices that may be used to build influence in meritocratic OSS communities is absent (e.g., [6,24–26]). This gap leads us to pose our second research question:

1. What practices should a firm consider to gain influence on the RE process in a meritocratic OSS community?

To address these two research questions, this paper presents a Community Strategy Framework (CSF). A community strategy should describe if and why a firm needs influence on the RE process in a specific OSS community, and how the firm could gain it. Thus, the objective of CSF is to help firms create and tailor community strategies that enable them to focus resources on communities that matter and gain the influence needed on their respective RE processes.

Using a design science approach [27,28], we leverage a series of ten semi-structured interviews with industry professionals to explore the problem context. Interview transcripts were then inductively coded [29] which resulted in a first design of the CSF. To validate and refine the design, seven interviews were conducted where the interviewees were presented with the CSF and asked questions regarding its completeness and correctness. To evaluate the applicability and utility of CSF [27], in one of these interviews, the framework was also applied to a fictitious example based on an earlier reported case study [30]. As the last step, a case validation was conducted by interviewing four industry professionals from a software-intensive firm engaged in multiple OSS communities. Questions focused on the validity of CSF in the context of the firm's community engagements. In total, we conducted 21 interviews with 18 industry professionals from 12 different software-intensive firms.

The rest of the paper is structured as follows: in Section 2, we present related work, which this study builds upon. In Section 3, we present the research design of this study and how it was executed. In Section 4, we present the CSF, and in Section 5 the framework is applied to a fictitious example. In Section 6 we discuss our findings, followed by a discussion on threats to validity in Section 7. In Section 8, we conclude the paper.

2. Related work

In this section, we present the related work that provides a theoretical underpinning for the design of the artifact called the Community Strategy Framework (CSF). This theoretical basis is also used in the discussions on the validity of the proposed framework (see Section 6).

2.1. Requirements engineering in OSS communities

Compared to classic RE [31], OSS RE can be described as a collaborative, transparent and open process involving the stakeholders (both developers and users) in the community with interest in specific requirements [31,32]. Formal methods and processes, as well as documents or central repositories, are often absent [33,34]. Instead, a requirement may often be represented by multiple artifacts which are stored and managed in a series of interconnected and overlapping repositories, e.g., as an issue in an issue tracker and mail threads in a mailing list [35]. These repositories also function as communication channels for the stakeholders where the requirements are asserted (i.e., elicited from the OSS community perspective), analyzed, and specified informally, and often realized simultaneously [33,34,36,37]. This is an iterative process characterized as just-in-time RE [36,37] and where the social interactions between the stakeholders are often decentralized and dynamic [38]. However, these can on occasion also occur centralized in "off-line" events such as conferences, meet-ups, and hackathons [20,30,39].

Prioritization and selection of requirements are commonly performed by individuals in leadership positions of the OSS community, however, with consideration taken to expressed wishes of the community [11–13]. This hierarchy between the roles in OSS communities is often depicted with the help of an onion model [40]. In its multi-layered construction, central and leadership roles can be found among the core layers, while the passive users can be found in the outer ones (cf. Core-Periphery Model [41]). The structure implies that the further out a community member is, the less direct influence and knowledge the person has over the project's state and direction [42]. Furthermore, what roles that exist in a community, specifically regarding leadership, may differ between communities. Some may, for example, have a project lead as with Linus Torvalds in the Linux kernel community, while some may have a core team of entrusted members as in the PostgreSQL community [40].

Migration between layers can be fluid and agile depending on the project, e.g., community members can move between multiple layers, or be recruited into one, bypassing outer ones [42]. This migration further depends on the type of governance in the community.

2.2. Governance in OSS communities

de Laat [15] describes OSS governance as different configurations, primarily based on the authority structure, i.e., the way that authority is established, distributed, and exercised, either through autocratic or democratic principles. In the former, leadership is centralized and top-down, while in the latter it is decentralized and bottom-up. Building on this distinction, De Noni et al. [43] refines the two configurations further as presented in Fig 1. Concerning communities with autocratic tendencies, they differentiate between sponsor-based and tolerant dictator-based communities. In the former, leadership is centered around the sponsoring firm(s), while in the latter it is centered around a single project leader (tolerant dictator). In regards to communities with democratic tendencies, De Noni et al. [43] separates open-source-based and collective communities. In open-source based communities leadership is characterized as institutionalized, democratic, and distributed, often inside the walls of a foundation. In collective communities, leadership is seen as collective, meritocratic, and distributed.

Capra and Wasserman [44] makes a distinction between commercial and community OSS. In the former, the OSS project is owned and managed by a single firm [45], i.e., a special case of sponsor-based communities [43]. In the latter, the community is owned and managed by the community, which may include one or more firms, also aligning with the community-managed governance model as described by

Governance and Authority Structure Concepts and Relations				Reference
Autocratic communities		Democratic communities		de Laat (2007)
Sponsor-based communities	Tolerant dictator-based communities	Open-source-based communities	Collective communities	De Noni et al. (2013)
Commercial OSS projects	Community OSS projects			Capra and Wasserman (2008)
Single-vendor OSS projects	Multivendor OSS projects			Schaarschmidt et al. (2015)

Fig. 1. Overview of governance and authority structure concepts in OSS projects and their relations as presented in Section 2.

O'Mahony [46]. Schaarschmidt et al. [7] further label these types of projects as single-vendor projects and multivendor projects respectively.

Even with the categorizations of OSS governance models and their authority structures shown in Fig 1, other research shows that the picture can be more blurry. According to the literature review by Shaikh and Henfridsson [16], research has been consistent in describing how communities can only have one authority structure (with one notable exception [47]). Even though a community can evolve its authority structure in hybrid forms with time, a single authority structure will result in the end [19]. However, based on their view of a duality between governance and coordination, Shaikh and Henfridsson [16] move to suggest that multiple forms of authority structures can co-exist in parallel, each embedded in and operationalized by a coordination process. These coordination processes can integrate, and evolve together within a community, of which some may pass out with time and be replaced by others. In their longitudinal analysis of the Linux kernel community, they identified a varying mix of autocratic and oligarchic structures, but also semi-autonomous governing in terms of the different sub-modules. Meritocracy was continuously present through the analysis. i.e., even tolerant dictator-based communities can show traits of a community-managed [46] and meritocratic [15] governance model.

Although literature lists a number of them, meritocracy may be considered one of the more common authority structures, or type of governance in OSS communities (e.g., [2,17,23,35,40,48]). Based on merit and the earning of trust and status in the community, individuals are granted further responsibility and authority [17]. Merit correlates to the quality and quantity of the individual's contributions [11,22]. A common assumption is that these contributions are limited to technical code contributions, however, as is shown by Eckhardt et al.[18], this can be a simplification. Considering the onion model [40], several paths are depending on the type of role an individual possesses. Proven coordination and leadership skills are aspects that may be considered [19,42], but not obviously captured in code commits. As highlighted by O'Mahony and Ferraro [19] in their study of the Debian community, *"Any examination of meritocracy must develop a context-specific understanding of how merit is conceptualized"*.

2.3. Influencing the requirements engineering process in OSS communities

The members of the community all have their motives for participating, social or economic [49,50]. It may, therefore, be considered a challenge for firms to align their internal agenda with that of the community [1,7,51]. A decision to add functionality may require consensus in the community and approval by the community leadership depending on the type of governance. Being too aggressive with one's agenda may have an adverse effect and result in the functionality being blocked [32].

Dahlander et al. [20] differentiate how firms can adapt their relationship with an OSS community based on the level of influence needed. On a continuum scale, a relationship can be characterized as parasitic, commensalistic or symbiotic. In the parasitic approach, the firm takes without giving back, by some referred to as a "free-rider". In the com-

mensalistic approach, the firm contributes back when motivated, but focus on internal development. In the Symbiotic approach, the firm also sees to the best of the community, working to align internal and external development. The alignment is created through working as peers, and building status and recognition inside the community [21].

To build a symbiotic relationship, firms should first understand and learn to respect the needs, norms, and structure of the community [2,20,21,32,52,53], a form of "good citizenship" [51]. If there is a foundation encapsulating the OSS community, firms may have the option to gain influence through membership or sponsorship [48,51], or in other ways supporting the foundation, e.g., by supporting development with infrastructure [20], or general subject matter expertise [2]. In return, they may receive seats at relevant boards and committees through which they can make their voice heard [2,48]. Foundations, and similar boundary organizations between firms and an OSS community, are often limited to managing the technical direction of an OSS projects [51].

A more direct and general approach to the control of code contributions is by having "a man on the inside", letting employees engage with the community [7,21,23,51,52,54]. An alternative is to contract members of the community directly to have them work on matters of importance to the firm [1,7,11,51,55]. Through their engagement, these sponsored community members can take part in the RE processes by participating in discussions and providing both technical and non-technical contributions and support [2,30]. This work may take place both online and offline, because being visible and active on both ends is essential [7,19,30,39].

2.4. Determining the need for influence in OSS communities

As highlighted by Dahlander and Magnusson [1], it may be difficult to determine which OSS communities are of strategic importance to their operations. Firms should identify how they could benefit from an OSS project and its community, and then what kind of engagement is required to reap these potential benefits [2].

From a business model perspective, it may be considered how the OSS project helps to create, deliver, and capture value for a firm [56]. It may, for example, serve as a basis on which the firm builds complementary products or services, such as support and subscription offerings, or proprietary extensions [57]. The OSS project could also function as a product or service enabler, embedded in hardware products [58], or as tooling and infrastructure for development and service delivery [30]. From a more strategic perspective, the OSS project may provide value as a foundation for pooled R&D/product development, and as a mean for standardization of technology [59]. Furthermore, just as the community may serve as an external workforce, it may also serve as a marketing channel, both for customers and future employees [1,55,60]. Hence, the value should be viewed both from a monetary and a non-monetary perspective [57].

From a technical perspective, it is also essential to understand the strategic connection of the OSS project to a firm's business and how this is reflected in a developer's level [2]. There may be internal dependen-
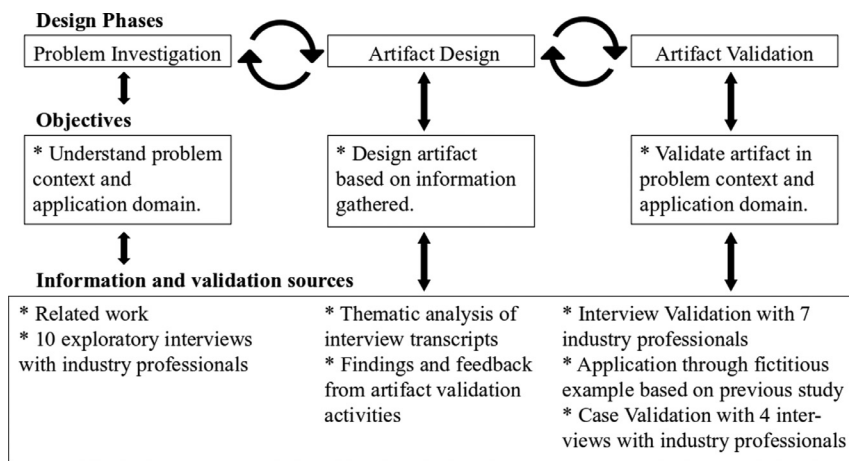
Fig. 2. Overview of the research process and context used in this study, using design science research [27,28]. Design steps includes problem investigation, artifact design and artifact validation, which are performed iteratively.

cies and integrations between the OSS project and internal software that are critical to maintain [30], as is specific functionality that is requested and expected by the firm's customers [58]. These two reasons both warrant a need for alignment between software development inside the firm and the community respectively [1,7]. If the direction of the community is predictable, both regarding road-map and release planning, then the need for an active community presence may be less urgent [2].

3. Research design

To develop the CSF, we used a design science research approach [27,28], in which research is performed and structured in the form of design cycles. A design cycle is comprised of three phases: problem investigation, artifact design, and artifact validation [28]. These phases are performed iteratively, as exemplified in Fig. 2. For example, as artifact validation renders feedback, this feedback is used for refinements in artifact design, resulting in a new artifact design that needs validation. Below, we use this structure to describe how we planned and executed the research behind this study.

3.1. Problem investigation phase

In the problem investigation phase, the problem context is analyzed. In our case, we conducted exploratory interviews to understand industry practice beyond what has been identified in the literature.

Ten individuals were interviewed (denoted I1-10, see Table 1) with a semi-structured approach where the interview instrument consisted of open-ended questions (see Section 9, Appendix A). The interviewees all held positions with responsibilities relevant to understanding how their respective firms work and engage with OSS communities. They were selected based on convenience sampling. All interviews lasted between 30 to 60 min and were conducted either in person or over video link by the first author of this study. All interviews were audio recorded and transcribed.

3.2. Artifact design phase

Drawing on the knowledge and understanding that is obtained during the problem investigation, an artifact is designed with the hypothesis that it will address the design problem. In this study, the design problem is stipulated by **RQ1** and **RQ2**, and the artifact is the CSF.

The interview transcripts were coded with an inductive approach by the first author with audit trails intact [29]. Sentences and paragraphs were first assigned descriptive topics. These topics were later collected under common codes, which could then be related and sorted under **RQ1** and **RQ2** respectively. Codes relating to **RQ1** are referred to as aspects and are divided into three categories; Business Aspects (BA),

Technical Aspects (TA) and Community Aspects (CA). Codes relating to **RQ2** are referred to as engagement practices and are collected in one single category. The CSF is presented in full detail in Section 4.

Below we provide an example with a subset of quotes rendering in engagement practice 5 (EP5) of the CSF:

- Engagement Practice (RQ2)
  - Offer the expertize and resources of the firm
    - Quote by I1: *"...contributing DevOps-kind of information and documentation and information, and it gives credebility"*.
    - Quote by I7: *"We don't have developers, but we send you these machines to do testing"*.

3.3. Artifact validation phase

In the artifact validation phase, the artifact is tested as a candidate solution to the defined design problem. In our study, this phase consisted of three steps. First, we conducted seven validation-focused interviews with four new industry professionals (I11-14), but also three from the problem investigation phase (I1, I5, I6), see Table 1. Second, to evaluate applicability and utility (i.e., descriptive validation [27]), the framework was applied through a fictitious example on a previously performed case study on how Sony Mobile evolved in their engagement with the communities of Jenkins and Gerrit [30]. Third, the CSF was validated in a similar way as in the first step, but within the context of a software-intensive firm (CaseOrg) and its Tools department.

3.3.1. Interview validation

The CSF was presented and discussed one element (aspect or practice) at a time to the interviewees. Discussions focused on whether something was redundant, missing, or could potentially be modified. Interviews were audio recorded and transcribed.

After verifying with transcripts, this step of the validation phase resulted in one TA being removed, TA2 being reformulated to also focus on the "fitness-of-use", and the explicit addition of TA4. A further consideration brought up in several of the validation interviews was that, while the business and technical aspects are relevant for determining the need for influence on the RE process, the community aspects are on the other hand used for determining the feasibility and potential of gaining influence in the community. I13, for example, describes it as, *"So your first two sets of criteria, the business and technical aspects, felt like you were deciding yes or no, we should care about this community. The community aspects don't feel like yes/no's, we should care, these feel much more like feasibility, can we do it or not"*. Furthermore, the validation interviews resulted in the validation of and more nuances to existing aspects and practices. I12 for example added to BA2 the perspective that standardization can be part of a strategy to build a software ecosystem. From

Table 1

Ten industry professionals (I1-I10) were interviewed in the problem investigation phase. Seven industry professionals (I1, I5, I6, I11-I14) were interviewed in the Interview Validation phase. Four industry professionals (I15-18) were interviewed in the Case Validation. Small-sized firms (S): <50 employees, Medium-sized firms (M): 50 <>250 employees, Large-sized firms (L): >251 employees).

ID	Title	Firm	Business	Size	Use of OSS
I1	OSS Program Officer	A	Telecom	L	Infrastructure
I2	Community Manager	A	Telecom	L	Infrastructure
I3	OSS Program Officer	B	Software products	L	Infrastructure & Products
I4	OSS Strategist	C	Software products	L	Infrastructure & Products
I5	Community Manager	D	Software products	S	Products
I6	Community Manager	E	Software products	S	Products
I7	OSS Strategist	F	Software products	L	Products
I8	OSS Program Officer	G	Software products	L	Infrastructure & Products
I9	OSS Program Officer	H	Software products	L	Infrastructure & Products
I10	OSS Strategist	I	Consumer electronics	L	Products
I11	OSS Strategist	J	Consultancy	S	Strategy services
I12	Community Manager	F	Software products	L	OSS products
I13	Community Manager	F	Software products	L	Products
I14	OSS Program Officer	K	Consumer electronics	L	Products
I15	Team Manager	L	Embedded systems	L	Infrastructure
I16	Project Manager	L	Embedded systems	L	Infrastructure
I17	Senior Developer	L	Embedded systems	L	Infrastructure
I18	Junior Developer	L	Embedded systems	L	Infrastructure

a design science perspective, findings and feedback from the validation phase were used to refine the artifact design.

3.3.2. Framework application example

The analysis was performed by the first author of this study, who was also one of the authors behind the previous case study [30]. Using interview transcripts and codings from the original study, the CSF was applied by considering each aspect against the Jenkins and Gerrit communities. Traces and support for the different engagement practices were then searched for. Findings were then summarized and verified for correctness with the OSS Program manager at Sony Mobile. The program manager was presented with the results from each of the applied practices, and the support gathered for each of the engagement practices. The program manager was asked to verify the interpretation and clarify any misunderstandings of the first author's analysis. It should be noted that the program manager was also one of the interviewees from the previous case study [30].

3.3.3. Case validation

The Tools department has a similar organization and purpose as that described in earlier work of Sony Mobile, which is the foundation for the application example as described in Section 3.3.2. CaseOrgs's Tools department develops and maintains multiple OSS tools and infrastructure projects, including Jenkins and Gerrit, to support its product development organization. All OSS communities that were discussed during interviews were characterized as community-managed and meritocratic.

Four interviews were conducted with I15-I18 (see Table 1) who all held various positions but were all engaged in different OSS communities, some with maintainership positions. As in the previous step (see Section 3.3.1), the CSF was presented and discussed one element (aspect or practice) at a time to the interviewees. Discussions focused on whether something was redundant, missing, or could potentially be modified, specifically in the context of the communities that CaseOrg's Tools department is engaged in. Interviews were audio recorded and transcribed. Findings and feedback were used to refine the artifact design of CSF. No aspects or practices were removed or added. Existing ones were however given more nuances as EP5 where the importance of attending and arranging hackathons was added.

4. Community Strategy Framework

Here we describe the Community Strategy Framework (CSF) as presented in Table 2, which consists of two parts. The first of these, contain

aspects a firm should consider when assessing its need to influence the RE process in an OSS community (**RQ1**). The second part of the CSF consists of practices a firm should consider to gain influence on the RE process in an OSS community with meritocratic governance or aspects thereof [16] (**RQ2**). Fig. 3 shows an overview of the CSF. A firm constructs a community strategy by firstly assessing the community of interest based on four Business Aspects (BA1-4) and four Technical Aspects (TA1-4), and secondly determine the actual need for and feasibility of gaining influence using the four Community Aspects (CA1-4). It may be that not all aspects are applicable or relevant. It may also be that one aspect may indicate a need for influence, while another may not. With this in mind, it is up to the user to consider the different aspects in relation to the community of interest, and weigh these against each other. The CSF should, therefore, be viewed as a support for the user to arrive at a decision on if and how much influence is needed by the firm on the RE process in the OSS community.

Once such a decision has been made, the firm then formulates important engagement goals and selects which Engagement Practices (EP1-8) to apply, and finally determine how to apply them. Below we present the respective aspects and engagement practices in detail.

For further guidance on how to apply the CSF, please see Section 5 where it is applied in a case example based on earlier work [30].

4.1. Aspects

The aspects are divided into three categories: business, technical, and community aspects. Aspects from the two former categories are used to reflect on the OSS and its importance to the firm from a business and technical perspective. The latter, community aspects, are used to reflect on the feasibility and potential to gain influence, as well as the need for it.

4.1.1. Business Aspects (BAs)

BA1 - Connection between the OSS project and the value proposition and revenue streams of the firm's business model. As expressed by I10, *"It comes down to the bottom-line, and making sure where [the firm] is making money, we want to have as much impact on those areas as possible"*. I11 emphasizes *"I think the sticky point is understanding how the value of the open source matches to the value of the business they're trying to build"*. A firm should, therefore, recognize how the OSS project is leveraged in its business model. It can be a complement of the core

Table 2

Overview of the Community Strategy Framework. Business, Technical and Community Aspects relate to **RQ1** and Engagement Practices to **RQ2**.

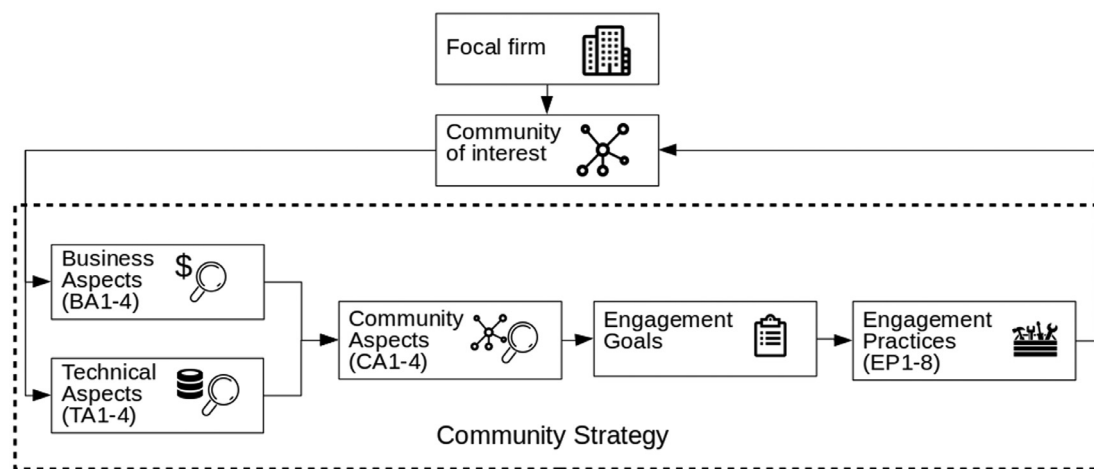|	Business Aspects (BA)
BA1	Connection between the OSS project and the value proposition and revenue streams of the firm's business model.
BA2	Connection between the OSS project and the business strategy of the firm.
BA3	Importance of the OSS community as a pool for recruitment
BA4	Need of the OSS community-related visibility and credibility towards the firm's customers
	Technical Aspects (TA)
TA1	Internal dependency of the OSS project inside the firm
TA2	Fitness-of-use and road-map alignment of the OSS project
TA3	Dependency on the OSS community's release planning
TA4	Need for competence and resources of the OSS community
	Community Aspects (CA)
CA1	Presence, influence and agenda of other stakeholders in the OSS community
CA2	Diversity and activity in the OSS community's stakeholder population
CA3	Openness in Culture and Governance of the OSS community
CA4	Ownership and management of the OSS project
	Engagement Practices (EP)
EP1	Understand the governance structure and have seats in right groups, committees and boards
EP2	Become a member or sponsor of the foundation or governing community body
EP3	Sponsor, contract or hire developers and maintainers to engineer contributions and mentor internal engineers
EP4	Contribute to the development of the OSS project through internal engineers
EP5	Offer the expertize and resources of the firm
EP6	Have an active on-line and off-line community presence
EP7	Be open and humble to the OSS community
EP8	Build an inner source culture and practice inside the firm

Fig. 3. Overview of the Community Strategy Framework's related process. A firm first values the community of interest with the business and technical aspects and then uses the community aspects to determine the feasibility of gaining influence and potential engagement goals. Engagement goals are then decided and engagement practices chosen.

value proposition as for Red Hat and their distribution Red Hat Enterprise Linux which is based on Fedora. It could also be an enabler for the value proposition as for Sony Mobile and Android which is used in their mobile handsets. Or it could play a more indirect role as part of an infrastructure or a tool-chain that can be used to develop and deliver the main value proposition. In the latter case, there may be a limited amount of competitive edge connected to how the OSS project is used internally, as described by I16, *"We're so far out of core business that we have our own contribution process"*. In other cases, *"if you're building a product offering around an open source project in the core, there's not even a question. If you're committing to customers to support the project, then you need to have an influence on that project"* (I12).

BA2 - Connection between the OSS project and the business strategy of the firm. The business strategy specifies how a firm should navigate a changing environment and as a consequence construct and adapt its business model [61]. In this context, an OSS project and its community can play a pivotal part, e.g., to commoditize a market, or

change the default technology being used by industry. I9 reflected on one of their experiences, *"So we wanted to change the industry conversation, and we wanted to have a substantial impact in that"*. This type of standardization can further be part of a strategy where the intent is to build a software ecosystem, as explained by I12, *"Driving standardization enables the market to potentially develop and that is what gives business opportunity, if you're running an infrastructure project and all of a sudden you have a lot of third-party vendors, whether monitoring and logging, or storage or network, you know there's an entire ecosystem that comes along, not to mention all of the developer toolings that you need to develop container-native applications. So there's a lot of opportunities that come from having a de facto technology base in the platform. And that creates that opportunity to create the commercial ecosystem around the platform"*.

BA3 - Importance of the OSS community as a pool for recruitment. Being active and influential in an OSS community can be important to attract and maintain a skilled workforce. This concerns both specific technologies where skilled people are scarce and attracting de-

velopers in general. The latter is emphasized by I7, *"It's also about reputation - [firm] created a big open source office, and a huge open source initiative, because no one wanted to work with them"*. I11 adds, *"This is something a lot more are starting to realize, particularly large companies with aging populations, that people don't want to sit in a stodgy old company in cubicles"*. I16 continues, *"We need to show that we don't just consume, but also contribute to attracting good developers. The community becomes a channel for new employees"*.

BA4 - Need of the OSS community-related visibility and credibility towards the firm's customers. As for recruiting talent, being active and influential in an OSS community can be essential to attract and maintain customers. It can be to prove technical competence, but also the ability to push features upstream. As put by I1, *"[The OSS project] was the selling point of the product. We needed to demonstrate to customers that we were one of the core contributors of [the OSS project]"*. I11 gives the example of IBM and how they, *"...back in the 2000s, invested a billion dollars in Linux and they wanted to make a big deal of it because they saw that as an emerging market that they wanted to get into"*. I1 adds how this aspect is particularly important for firms using OSS in their products, such as *"Red Hat, or any commercial open source project. Like Cloudera would need to do that, that they have influence in the Hadoop community, and DataStack for Cassandra"*.

4.1.2. Technical Aspects (TAs)

TA1 - Internal dependency of the OSS project inside the firm. Technical dependencies between an OSS project and a firm's internal software can be considered an architectural reflection of how an OSS project connects to a firm's value proposition (BA1). Certain features in a product or parts in an infrastructure may be dependent on the project. I3 phrases the question as *"Do you depend on this or do you not? And how much do you depend on it? How much functionality goes into your product that's based on upstream software? How heavily are these integrated into your product?"*. I17 exemplifies, *"We are extremely dependent on [OSS project], we have based our whole infrastructure chain on it. This requires us to be active so that we can affect in what directions the tools head"*.

TA2 - Fitness-of-use and road-map alignment of the OSS project. Deviance between a firm's internal and a community's requirements and road-maps may be essential to address in order to avoid or minimize technical debt. I12 refers to an OSS project's fitness-of-use and explains it as *"How many things that we need it to do does this project do today? And if you feel like there is a delta between what it does today, and what you need it to do, and this is a strategically important component of your plan, then it would be important to be involved. You need to have influence so that you can affect the change that you need in that project"*. I1 adds that in *"...some cases, it may not be necessary for you to be as actively involved because you are happy with the direction it is going, and it's a mature and stable project. And in some cases, you really need to be there and watch it and make sure it goes in the right direction"*.

TA3 - Dependency on the OSS community's release planning. A firm can be more or less dependent on the release planning of an OSS community, and have various needs to synchronize it with that of any internal development. Getting features upstream quickly and running the latest release may be an essential factor for firms whose customers may expect quick access to the latest functionality, as some buyers do of Android-based mobile handset manufacturers. For others, it may be less of a concern, as for Red Hat who focuses on offering a stable and secure version of Fedora. I4 explains it as *"How much do we care if they are changing it rapidly? Are we living on a fork and are willing to eat a little bit of fit and finish? Or do we really want to be on the latest bits all the time?"*. I12 sees it from a risk analysis perspective, *"Is there is a risk that a feature will not go into a project, or is there a risk that a project that you depend on will miss its release date?"*.

TA4 - Need for competence and resources of the OSS community. Firms can be limited, both in terms of *"...resources, time or people"* as highlighted by I16, or in terms of specific competencies that are internally available. By engaging in an OSS community, firms may have

an opportunity to gain these resources through collaboration and "co-opetition". By growing influence in such a community, a firm can better exploit and steer these resources to best match the firm's agenda. I1 explains it as, *"Sometimes we may not have the competency inside the company, but yet we want to draw on the competency of the community to help us use something correctly. So, the link to the community may be important because of that, to just improve our own competency in handling that project"*.

4.1.3. Community Aspects (CAs)

CA1 - Presence, influence, and agenda of other stakeholders in the OSS community. Knowing whom the stakeholders are, where they focus their resources, with whom they collaborate and how much influence they hold, can signal how a firm should consider its relationship with the stakeholder, but also overall community engagement. As expressed by I3, *"If it's a company that wields a big influence, and they are a competitor to you, there's a much different way to approach that than if they were a partner or one you're not a competitor with"*. I3 continues, *"If you understand why those companies are contributing it potentially makes your strategy why you should be contributing"*. I1 explains that for projects which are important to a firm, *"You work on it, you contribute to it, and you make sure your competitors are not influencing it differently than you would"*. Hence, the presence of competitors may indicate *"how strongly [a firm] need to be present"*, as further highlighted by I1. However, as explained by I13, OSS communities provide a *"...forum for competitors to cooperate in a way that doesn't upset their shareholders, a form of co-opetition"*. Presence of competitors may also be *"a signal that the OSS project we should be engaged in, maybe it is becoming an industry standard"*, as suggested by I2.

CA2 - Diversity and activity in the OSS community's stakeholder population. A community maintained by only a few individuals or companies could be vulnerable if they were to leave for any reason. As asked by I3, *"What would be the case if there were shortages in supply, i.e., the project would no longer be available?"*. I12 compares a community to an external vendor and asks, *"Is this company going to be in business in five years? Is there someone who can take up the mantle if they shut down?"*. A low level of diversity and activity in a community can, therefore, be a warning sign if a firm is to engage in the first place. However, if a firm is dependent on a community or sees potential, then it indicates that the firm should invest, *"...not for influence, but for health"* as emphasized by I1. I9 further adds, *"We want to make sure it is not just totally dependent on one or two parties only because vibrant community to me means that it has a broad spectrum of contributions and that it is not just totally dependent on one party"*. From a sourcing perspective, it may be relevant to also consider other alternatives and weigh these against the cost of investing in the concerned community.

CA3 - Openness in Culture and Governance of the OSS community. To be attractive, the culture and governance of an OSS community should have meritocratic influences, i.e., be open to new members joining and gaining in rank, but also for discussions regarding road-maps and ways of working to be open. This is further explained by I9, *'If there are communities that are uninterested in changing and learning then that is a community in my opinion that will stagnate and contract, they will have a hard time growing new leadership, they will have a hard time evolving as the needs of users evolve, as the needs of community evolve"*. I9 continues, *'An openness to constant improvement and to input needs to be a core value, or at least be demonstrated in a community in order to consider putting any substantial investment"*. If the project is important for the firm, a low level of "openness" could motivate a high investment and active engagement to be able to affect the culture and governance of the community if deemed possible.

CA4 - Ownership and management of the OSS project. A criterion before engaging in an OSS community is to determine whether there is potential for the firm to gain influence and extract the expected value. As highlighted by I10, *"If we see that it's a project that is controlled by one company, and it doesn't look like we'll be able to influence it in a way we*

want, we may not get involved in that project". I.e., if the OSS project is now owned and managed by the community, or a legal entity representing it (e.g., a foundation), gaining influence through active contributions and engagement may prove hard. If a firm's strategy is to hire a maintainer to get influence and there is no one available, *"...the project becomes much less attractive"*, as stated by I10.

4.2. Engagement Practices (EPs)

The engagement practices presented in this section should be seen as a tool-box of ways in how a firm can engage with a meritocratic community to build the influence needed.

EP1 - Understand the governance structure and have seats in right groups, committees, and boards. Depending on the complexity of the community governance structure, there can be many groups and committees where decisions are made. As described by I13, *"It's very dependent on the community, some have large foundations, while others may have less"*. Hence, a firm should first *"...understand where decisions get made and what kinds of decisions [they] need to influence. Is it a technical decision? Is it a positioning decision? Is it a communication decision? And hence, which body do you need to be on, or what level of membership do you need?"* as stated by I1. Once understanding the governance structure of the community, a firm may need to build a certain level of influence to be able to join the identified groups. This need can also concern groups and committees that may not be a direct part of the community, but part of a greater ecosystem affecting the OSS community. As expressed by I1, *"It's an influence game making sure you have people in all the right places, joined all the right foundations"*. I2 provides an example, *"I think with [OSS community] they did that, ok, we need to sit at this group, this group, this group, we need to get a seat at the table of the user committee, we need to be on this committee, and they actually mapped it, and they put people there"*.

EP2 - Become a member or sponsor of the foundation or governing community body. Once a firm understands the community and its governance structure, they can start to consider whether they should become a member or sponsor if possible. If the community is run under a foundation, a membership can give a firm visibility and marketing to show community, customers, and potential newly-hires that they are both competent and committed in regards to an OSS community. However, it does not have to imply a direct influence on the OSS community automatically. As explained by I3, *"You can potentially buy yourself into the business side of the governance, but you don't get any technical influence unless you do any work"*. I13 gives the example of GNOME, *"You pay to be part of the advisory board, but it has very little power. You have to be a contributing member to be elected to the board of directors, and that's where the power is"*. A membership or sponsorship should instead be seen as a long-term investment that can help build a sustainable influence through growing and attracting influential community members and maintainers. Sometimes membership may be unnecessary, as explained by I1, *"A lot of these bodies have end-user boards which do not require any pay-to-play, it just requires you to be a big user of that technology. Because a lot of projects are very eager to get feedback on how you are using it, what are the challenges that you face at scale? So they see that as currency and value. So we've kind of been reexamining our presence in some of these bodies and asking why are we spending 40K when we can get the same influence through being on the end-user committee?"*.

EP3 - Sponsor, contract or hire developers and maintainers to engineer contributions and mentor internal engineers. To build influence organically by on-ramping new developers into a community can be time-consuming, why a firm may consider hiring existing maintainers and developers in leadership positions. As explained by I3, *"If your willing to do a longer-term play, then you get people already in your development team starting to make upstream contributions, then it may take a year or two years depending on what kind of community it is, to have the influence long term. But if you needed that influence yesterday, the only way is to hire someone that is a very strong contributor or maintainer"*. I10 adds,

"We like to hire people in leadership positions. And once we get to two or three people that are in those sort of positions, then we can get started introducing some junior developers". This kind of mentoring is further endorsed by I12, *"I would hire the contractor to teach how to do the work. So it's kind of on-the-job-training"*

EP4 - Contribute to the development of the OSS project through internal engineers. Long-term and sustainable influence is built by directly contributing to the development of the OSS project. These contributions are not limited to code, but may also include *"...writing documentation, testing, answering questions, doing the mud work, doing a lot of the things that no one wants to do"*, as explained by I3. Developers need to be enabled to actively engage in the community development process without being hindered by internal contribution processes. I17 adds, *"Principally all open source communities are run as meritocracies so we need to be active. If we want to be able to change the direction in [OSS project], we need to produce code and plugins to show that we are part of the community"*.

EP5 - Offer the expertize and resources of the firm. If a firm holds specific resources, these can also provide valuable contributions to the community. These resources can, for example, be infrastructure-related, but also include soft factors. I1 exemplifies how they provide large-scale testing capabilities, as well as credibility to an OSS project that they run in production, *"if you have big companies like us using [OSS project], it says that it is a viable product"*. I9 adds another example where they provide server space for the community to run compute, test and build processes, enabling the active development in the OSS community.

EP6 - Have an active on-line and off-line community presence. Community discussions regarding the development of an OSS project take place in on-line mediums such as issue-trackers, chats and social media, but also off-line at events and social gatherings such as meetups, conferences, and hackathons. For a firm to grow and leverage its influence, it needs to be present and take an active part in these discussions, and also help to facilitate them, e.g., by arranging their own events. I17 exemplifies, *"Concerning [OSS project], we are extremely active at hackathons... We travel a lot to get and know the people... Recently we hosted a hackathon where we gathered basically all maintainers of the project"*.

These activities should be coordinated internally as highlighted by I1, *"[The Community Manager] ran community activities internally and externally, created awareness of what we were doing in [OSS community], making sure that people contributed to the right projects, submitted abstracts to the right projects, were elected to the right bodies, showed up at the right conferences"*. Having dedicated developer advocates and community managers was a generally recommended practice. This person should be able to mediate and be a spokesperson both of the community and the firm.

EP7 - Be open and humble to the OSS community. When joining a community, I9 explains, a firm should adapt to the culture and way of working in the community. They should *"...come in humbly and offer to help in things where they have expertize as opposed to 'We need to do a thing, it's gotta be done this way', then you are going to get an immune reaction if you start that way"*. I12 gives the comparison, *"Joining a new open source community is like moving into a new neighborhood. There is a way of doing things, some of these things are going to be built up during time, and there is going to be inertia. So there are things that are obviously better, that people are going to agree is obviously better, but they are used to the way things are done. So you kind of have to figure out how to bring change gradually. And at the same time is that you figure out how things work. And so, first figure out how the community works before you come in and propose a lot of changes, so don't be excessively critical of the way things are done in a specific neighborhood, then no one is going to listen to you when you propose changes"*. Furthermore, the firm should be *"...transparent and open about the intentions and the agenda with the community and project, e.g., road-map, what you are keeping closed and for what reason"* as highlighted by I7. This is further emphasized by I13, *"You need to be open and completely honest about the why even if it's for profit because otherwise you just*

look suspicious". The firm should hence differentiate between the communication they use towards the community and that which they use towards for example their employees and customers.

EP8 - Build an inner source culture and practice inside the firm. By introducing inner source culture and development practices internally, a firm can help its developers to learn better how to work with external OSS communities, and simplify on-ramps. I13 explains the importance of teaching internal engineers about OSS development practices, such as working distributed and decentralized, *"A lot of companies where everyone sits in the same office all talking among themselves and have meetings just them, and they need to learn how to do everything online, and include the people who are not there"*. In effect, this can create more contributors for the firm and in a longer perspective help raise its influence in OSS communities in general.

5. Framework application example: Jenkins and Gerrit

In this section, we illustrate through a fictitious example of how the CSF could be applied (cf. descriptive validation [27]), based on a previously studied case which describes how Sony Mobile and its Tools department evolved in their engagement with the communities of the two OSS projects, Jenkins, and Gerrit [30].

Initially, Sony Mobile had a restrictive view of what they shared with the two communities and how they engaged. They focused mainly on doing bug-fixes, general knowledge-sharing and had a community presence limited to online channels, such as mailing lists and issue trackers. The engineers in the Tools department focused on internal work and tailoring of the two OSS projects to internal needs. They further saw that they could create a competitive advantage by keeping internally developed features closed. However, with time, the attitude towards the two communities evolved into a more symbiotic relationship. Sony Mobile and the engineers at the Tools department saw increased benefits with having an active engagement and being more open. As then highlighted by Sony Mobile's Director of OSS operations (I5) - *"...not only should [the tool-chain] be based on OSS, but we should behave like an active committer in the ways we can control, understand and even steer it up to the way we want to have it"*. It is in this context that the aspects of the CSF are analyzed and discussed for Jenkins and Gerrit.

5.1. Defining the need for influence in the Jenkins and Gerrit communities

Below we investigate the need for influence in the Jenkins and Gerrit communities by considering the business, technical and community aspects of the CSF from Sony Mobile's point of view.

5.1.1. Business Aspects

The connection between the two OSS projects and the business model of Sony Mobile (**BA1**) was indirect in the sense that the projects were used in the development infrastructure that engineers leverage in the product development inside Sony Mobile. Perceived benefits from the use of the two OSS projects include improved quality of Sony Mobile's end-products, as well as shorter time-to-release and market. Both OSS projects were seen as a commodity and a non-competitive advantage. There were alternative solutions available, but most were proprietary, and the primary motivation for an OSS option was that Sony Mobile could customize the OSS projects based on internal needs much more easily.

The adoption of Jenkins and Gerrit was part of a broader strategy (**BA2**) of moving Sony Mobile more towards usage of OSS, as well as the adoption of the same tool-chain used by Google in the Android development.

Both communities made up important pools for finding and attracting new and talented employees that could help in adapting the two OSS projects to the preference of Sony Mobile (**B3**). However, as neither Jenkins or Gerrit was a part of the product or any marketing, there was no need to establish a certain level of visibility or credibility towards the customers (**B4**).

5.1.2. Technical Aspects

Both Jenkins and Gerrit made up pivotal parts of the continuous integration tool-chain inside Sony Mobile. Therefore, there were many interactions and dependencies between the two OSS projects and as well as to other tools in the tool-chain (**TA1**). They had been tailored to internal requirements and supported the development process defined internally.

Sony Mobile was dependent on a stable and secure infrastructure, why they did not need to use the latest or experimental releases (**TA3**). In general, however, there was an expressed goal to avoid too many patches, and adaptations as the Tools department was limited in resources and had to rely on the community for much of the development (**TA4**). Further, there was a need to introduce a heavier focus on scalability in the two OSS projects, as they at the time were not optimal in large-scale setups as that used by Sony Mobile (**TA2**).

5.1.3. Community Aspects

Both the Jenkins and Gerrit communities had several firms involved, including direct competitors to Sony Mobile. However, as both OSS projects were seen as non-competitive by Sony Mobile, this presence was not considered as an issue. Few of the existing stakeholder had the equivalent or larger size of installations, which made Sony mobile somewhat unique in its need for improved scalability (**CA1**).

In general, both communities were very active and diverse concerning contributors and users (**CA2**). Also, the culture and governance structure was very open for new contributors to join in discussions and rise in rank (**CA2**). Due to the healthy activity, and meritocratic culture and governance of the communities, it was also deemed easy to increase influence, both organically by introducing employees, but also through hiring new talent as both communities are community-managed(**CA4**).

5.1.4. Summary and goals for engagement

Even though classified as non-competitive, there was a defined need to be able to influence the road-maps of the OSS projects, and be able to contribute larger features (e.g., related to improving scalability). Due to the limited size of the Tools department, there was also an expressed goal to be able to find and create collaborations when possible, even with competitors.

5.2. Defining the engagement activities in the Jenkins and Gerrit communities

Based on the determined need for influence and goals that were defined, Sony Mobile and its Tools department became more active and open in their engagement with the two communities.

No foundations were surrounding the two projects, why there was no need to attain a specific membership or sponsorship (**EP1**). However, there were committer groups (**EP2**), i.e., central parts in the communities' governance [40], that Sony Mobile wanted to join.

Sony Mobile did not see a need to rush and hire engineers from the communities directly (**EP3**). Instead, they grew their influence organically by introducing their engineers to the communities to the point where they managed to get to positions in the Gerrit committer group. The engineers were given frame agreements for the two communities where they were allowed to contribute freely, both in regards to features and bug-fixes (**EP4**). With their large set-up and testing infrastructure, Sony Mobile could also contribute to improving the quality of the two tools (**EP5**).

The engineers at the Tools department were active and visible in both online and offline communication channels (**EP6**). Their online presence included active participation in discussion and knowledge sharing through mailing lists, issue trackers, chat channels, and webinars. Offline presence included attending conferences, meet-ups, and

hackathons. The latter was seen as an essential forum to do quick implementations (cf. just-in-time RE [37]) as often many of the more influential persons in the communities were gathered in the same room. Alongside this active engagement, Sony Mobile had an open attitude towards the community and was transparent with its agenda. Engineers presented how the two projects were setup internally, as well as best practices and know problems when possible. They even talked to and engaged in knowledge-sharing with direct competitors (**EP7**).

The engagement and internal development of Jenkins and Gerrit were further seen as a seed to create an inner source initiative inside Sony Mobile, with the ambition to spread into others corners of the Sony Corporation (**EP8**). The goal was to grow more contributors and active users to Jenkins and Gerrit, but also in other projects, and maybe even create new ones where motivated.

6. Discussion

Below we discuss the validity of the CSF and contrast it to related work.

6.1. Determining the need for Influence in OSS communities

From a business perspective, as highlighted by I11, the *"…sticky point is understanding how the value of the open source matches to the value of the business [a firm is] trying to build"*. In this sense, the business model concept provides a useful lens to frame how the OSS helps to create, deliver, and capture value for a firm [56], and more specifically, through the OSS project's connection to the value proposition and revenue streams as pointed out in BA1 of the CSF. As indicated by the diversity of the firms that the interviewees represent, this connection can be made in several ways, as is reported in literature [30,49,57,62]. This is also true on the business strategy level where the firm chooses and configures its business model to compete in its business environment [61]. Creating or supporting a competing standard, or commoditizing a technology or market are two ways in how a firm can disrupt their competition and pave the way for their own business model, both reported on in the CSF (BA2) and in literature [59]. This alignment between the CSF and literature is further repeated in regards to the importance of an OSS community as a pool for recruitment (BA3), as well as a marketing tool towards customers (BA4) [1,60].

On an implementation level, it is also important to understand the reflections between how the OSS project is used in the internal development and it's strategic importance to a firm [2,7]. In the example of Sony Mobile and the communities of Jenkins and Gerrit (see Section 5 and [30]), the two OSS projects played a less direct part in the firm's value proposition, but a much more significant from a technical perspective. They constituted core parts in the internal development infrastructure (TA1), and the communities were key partners to adapt and maintain the software (TA4). As the fitness-of-use and road-map alignment were not satisfactory, a high level of influence was required (TA2). If the case was otherwise, and the direction being predictable, the need for an active community presence may be less urgent [2].

Supportive evidence and alignment can hence be found between literature and many of the aspects identified in the interviews and presented in the CSF. However, aspects that need consideration may be different depending on the firm and community. For example, Sony Mobile had in regards to their customers, no need to prove credibility or visibility in the Jenkins and Gerrit communities, as these OSS projects were used internally and not part of any marketing or key selling points [30]. Hence, the business aspect BA4 is not relevant in this case, while in other cases it may.

Other aspects though can be considered more general such as business aspect BA1. An OSS project can, depending on the case, have a more direct or indirect connection with the value proposition and revenue streams of a firms business model. For Red Hat, the connection may most often be direct as they base many of their products on them [62].

Conversely, returning to the example of Sony Mobile, Jenkins and Gerrit had a more indirect connection as they enabled a customized development process. As reported, Sony Mobile experienced these community engagements as having a positive impact on time-to-market and quality of their products [30].

6.2. Influencing the requirements engineering process in OSS communities

As reported in the literature (see Section 2), the type of governance in OSS communities can vary [15,16,43,44]. There is also variation in the possibilities and ways how firms can gain influence on the RE processes in a community.

Among the cases researched, meritocracy seems to be among the more common authority structures [2,17,35,40,48]. In a meritocratic community, influence on the RE process is gained by proving merit, and as highlighted in the literature, this does not have to be limited to technical contributions [18,19]. In essence, it is about earning the trust and status among one's peers in a community [21,23,52]. Considering the differentiation by De Noni et al. [43] between open-source based or collective communities, this characteristic can be assigned to both, even though the former is described as institutionalized and democratic, and the later as collective and meritocratic. In a purely democratic community, an individual still needs to earn trust, respect, and recognition among its peers to gain responsibilities and authority. In Apache communities, for example, both democratic and meritocratic traits can be found as individuals are voted into leadership positions even though Apache communities are profiled mainly as meritocracies [17]. Further, as De Noni et al. classifies Apache communities as open-source based, rather than collective, one can view the two categories of authority structures as closely related, as is the way in how influence can be gained on their communities' RE processes.

In communities with a centralized and autocratic authority structure [15], i.e., firm-sponsored or tolerant dictator-based [43], project leadership is often centered to a single (or limited number of) firm(s) (e.g., Android Open Source Project) or person(s) (e.g., the Linux kernel project). In firm-sponsored communities [44,46], specifically those centered around a single firm [7,45], where the focus is more on transparency than accessibility [63], communities may often be viewed more as user communities and a less open type of software ecosystem [64,65]. To gain influence in these types of communities, firms may focus more on direct business relationships (cf. [66,67]). However, this does not prevent meritocratic governance aspects to be present why the community engagement practices as proposed by the CSF may still be relevant. In tolerant dictator-based communities, as shown by Shaikh and Henfridsson [16], there can still be mixes of meritocracy and democracy implemented through different coordination processes. Even if such coordination practices would not be present in an autocratic community, there is still some possibility to influence by earning trust and respect in the community. If a firm can create enough traction among their peers in the community, the project leadership will commonly consider it [11–13,30]. If not, and an opposing will is strong, part of the community may in worst case move to create their own fork of the project [68], as was the case with OpenOffice and LibreOffice [69].

Findings from the interviews regarding engagement practices align with Dahlander and Magnusson [20], in that influence in a meritocratic OSS community is built through creating a symbiotic relationship with the community. Trust and status are gained through active involvement and respecting its norms and values [20,22,23,32]. As pointed out by S9, gaining influence may be done through different types of engagements and with varying types of resources, *"It's bringing code, bringing people into influence, in most projects, buying influence is not as easy to do, but you can still spend sponsorship money and port money to make sure that a project is happier or healthier for example"*. I.e., influence may, for example, be gained through providing code contributions as well as more general resources, including financial, aligning with practices reported in literature [2,19,21,23,30,39,51,52,54]. When comparing the commu-

nities inside the Linux Foundation and the Apache Software Foundation, S9 describes it as, *"...you need to show contribution, activity, commitment, leadership, and then you grow through contributions that you take in both foundations"*. Hence, the engagement practices in the CSF are primarily intended for OSS communities where there is a presence of meritocratic coordination processes [16].

7. Threats to validity

As presented in Section 4, the CSF covers a broad spectrum of aspects, some more general and applicable than others. One reason for this may be that the 18 interviewees each have extensive personal experience in the field but with different backgrounds, e.g., business or developer-oriented. Another reason may be that they represent 12 different firms (see Table 1) which in turn may have different use cases and needs. From an external validity perspective [29], this is positive and indicates a potential of transferability to other firms who are engaged (or are aspiring to) in meritocratic OSS communities. However, as the CSF is based on qualitative data from a limited set of interviewees, quantitative conclusions on generalization will require further validation using statistics based on a population of real-world firms and OSS communities.

Another area regarding external validity is to what extent the practices presented by the CSF actually leads to a gain in influence, and in what contexts. As discussed in Section 6.2, we believe that there has to be meritocratic coordination processes present [16] as the engagement practices proposed in the CSF present make up different ways in how a firm can contribute to and engage with a community to build a symbiotic relationship based on trust, respect, and recognition among its peers. Interviews from the case validation (see Section 3.3.3) supports these arguments as the communities that CaseOrg is engaged in, and in the context which CSF was discussed, were all community-managed and meritocratic. However, external validity is still a limitation in regards to CSF why further empirical validation is needed in future studies, e.g., through the use of case studies and cross-case synthesis.

Regarding the completeness on the aspects and practices presented in the CSF, we again acknowledge that CSF is based on qualitative data from a limited set of interviewees. When performing the interviews in the Interview Validation step (see Section 3.3.1) of the validation phase we did reach a point of saturation where we observed a tendency of maturity in terms of declining number of emerged codes. This observation was further supported in the Case Validation (see Section 3.3.3) as no aspects or practices were added or removed, only further refined. This may point to some level of completeness. However, as presented in Section 2, there are numerous variations in the characteristics of OSS communities, e.g., in regards to governance structure, demographics, and RE process. Hence, further research and design cycles are needed to validate the CSF and to improve its level of completeness.

8. Conclusions

The focus of this study has been to identify what aspects that firms should consider when they assess their need of influencing the RE process in a meritocratic OSS community (**RQ1**), as well as what practices that should be considered in order to gain this influence (**RQ2**). To address these questions we used a design science approach [27,28]. We developed a questionnaire used in ten semi-structured interviews with industry professionals. Inductive coding of interview transcripts [29], an initial version of a Contribution Strategy Framework was developed. The framework was then validated and refined through seven new interviews and by applying it on a fictitious example of an earlier reported study [30]. Finally, a case validation was performed by interviewing four industry professionals from a software-intensive firm engaged in multiple OSS communities. Questions focused on the validity of CSF in the context of the firm's community engagements. In total, 21 inter-

views were conducted with 18 industry professionals from 12 different software-intensive firms.

The framework consists of aspects and engagement practices. The aspects address **RQ1** and are divided into business, technical, and community aspects. The two former may be considered to help determine how important an OSS project and its community is from the business and technical perspectives, while the community aspects add the perspective of feasibility and potential in gaining influence in a community. The engagement practices address **RQ2** and should be seen as a tool-box of ways in how a firm can engage with a community to build influence needed in the community.

As this study uses a qualitative survey approach with a limited sampling of interviewees, further research is needed to validate the CSF through case studies and additional empirical work, both qualitative and quantitative. Along with such research, more theory-grounding work should be performed to further formalize the concept of influence in OSS communities, and how it can be gained, as exemplified by the CSF. Inspiration may be gathered from Valença and Alves [67] in how they generated a theory of power for emerging software ecosystems formed by small-to-medium sized firms.

Conflicts of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Interview questionnaire

- Do you, in any way, consider or plan how you engage with a community, and where you spend your resource, and to what extent? If yes, how? Is it formalized in any way? How could this be improved/otherwise done
- In what ways can you contribute to an OSS community (code, knowledge, socializing, sponsorship)? What roles would you say are involved in these contributions?
- How can you gain the power to change or affect (influence) a community in terms of what features gets implemented, and how they are prioritized? (Short- and long-term)
- Do you see any connection or consideration between how you engage and invest in a community and the level of influence you need to have in it? How would you describe it?
- How can you consider how an OSS and its community creates value for your company? Is there any relation or consequence between this value and how you engage and invest in the community, and the influence you need in it?
- Are you aware of the product planning and development in the OSS communities you are involved in? Are your internal product planning and development aligned with the OSS communities'? Do you consider possible dependencies? Do you see a need for it? How would it affect how you engage and invest in the community and the influence you need in it?

- Do you consider the motivation and underlying drivers for why you engage and spend your resources in a community? If yes, how? Do these align with how the community creates value for you and its role in your product strategy? Do they align with what you contribute to the communities? What do you see as the main drivers of your company?

References

[1] L. Dahlander, M. Magnusson, How do firms make use of open source communities? Long Range Planning 41 (6) (2008) 629–649.

[2] S. Butler, J. Gamalielsson, B. Lundell, P. Jonsson, J. Sjöberg, A. Mattsson, N. Rickö, T. Gustavsson, J. Feist, S. Landemoo, et al., An investigation of work practices used by companies making contributions to established OSS projects, in: Proceedings of the 40th International Conference on Software Engineering: Software Engineering in Practice, ACM, 2018, pp. 201–210.

[3] S. Jansen, M.A. Cusumano, Software ecosystems: analyzing and managing business networks in the software industry, Edward Elgar Publishing, pp. 13–28.

[4] M. Glinz, R.J. Wieringa, Guest editors' introduction: stakeholders in requirements engineering, IEEE Softw. 24 (2) (2007) 18–20.

[5] B. Regnell, S. Brinkkemper, Market-driven requirements engineering for software products, in: Engineering and Managing Software Requirements, Springer, 2005, pp. 287–308.

[6] H. Munir, K. Wnuk, P. Runeson, Open innovation in software engineering: a systematic mapping study, Empir. Softw. Eng. 21 (2) (2016) 684–723.

[7] M. Schaarschmidt, G. Walsh, H.F. von Kortzfleisch, How do firms influence open source software communities? a framework and empirical analysis of different governance modes, Inf. Org. 25 (2) (2015) 99–114.

[8] H. Mäenpää, S. Mäkinen, T. Kilamo, T. Mikkonen, T. Männistö, P. Ritala, Organizing for openness: six models for developer involvement in hybrid oss projects, J. Internet Serv. Appl. 9 (1) (2018) 17.

[9] A. Milne, N. Maiden, Power and politics in requirements engineering: embracing the dark side? Requir. Eng 17 (2) (2012) 83–98.

[10] K. Wnuk, D. Pfahl, D. Callele, E.-A. Karlsson, How can open source software development help requirements management gain the potential of open innovation: an exploratory study, in: Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, ACM, 2012, pp. 271–280.

[11] D.M. German, The gnome project: a case study of open source, global software development, Softw. Process 8 (4) (2003) 201–215.

[12] P. Laurent, J. Cleland-Huang, Lessons learned from open source projects for facilitating online requirements processes, in: Requirements Engineering: Foundation for Software Quality, Springer, 2009, pp. 240–255.

[13] J. Noll, Innovation in open source software development: atale of two features, in: J. Feller, B. Fitzgerald, W. Scacchi, A. Sillitti (Eds.), Open Source Development, Adoption and Innovation, Springer US, Boston, MA, 2007, pp. 109–120.

[14] M.L. Markus, The governance of free/open source software projects: monolithic, multidimensional, or configurational? J. Manag. Gov. 11 (2) (2007) 151–163.

[15] P.B. De Laat, Governance of open source software: state of the art, J. Manag. Gov. 11 (2) (2007) 165–177.

[16] M. Shaikh, O. Henfridsson, Governing open source software through coordination processes, Inf. Org. 27 (2) (2017) 116–135.

[17] R.T. Fielding, Shared leadership in the apache project, Commun. ACM 42 (4) (1999) 42–43.

[18] E. Eckhardt, E. Kaats, S. Jansen, C. Alves, The merits of a meritocracy in open source software ecosystems, in: Proceedings of the 2014 European Conference on Software Architecture Workshops, ACM, 2014, p. 7.

[19] S. O'Mahony, F. Ferraro, The emergence of governance in an open source community, Acad. Manag. J. 50 (5) (2007) 1079–1106.

[20] L. Dahlander, M.G. Magnusson, Relationships between open source software companies and communities: observations from nordic firms, Res. Policy 34 (4) (2005) 481–493.

[21] L. Dahlander, M.W. Wallin, A man on the inside: unlocking communities as complementary assets, Res. Policy 35 (8) (2006) 1243–1259.

[22] M. Syeed, J. Lindman, I. Hammouda, Measuring perceived trust in open source software communities, in: F. Balaguer, R. Di Cosmo, A. Garrido, F. Kon, G. Robles, S. Zacchiroli (Eds.), Open Source Systems: Towards Robust Practices, Springer International Publishing, Cham, 2017, pp. 49–54.

[23] A. Nguyen-Duc, D.S. Cruzes, S. Terje, P. Abrahamsson, Do software firms collaborate or compete? a model of coopetition in community-initiated oss projects, e-Inf. Softw. Eng. J. 13 (1) (2019) 37–62.

[24] J. Linåker, B. Regnell, H. Munir, Requirements engineering in open innovation: a research agenda, in: Proceedings of the 2015 International Conference on Software and System Process, ACM, 2015, pp. 208–212.

[25] C. Alves, J. Oliveira, S. Jansen, Software ecosystems governance-a systematic literature review and research agenda, in: Proceedings of the 19th International Conference on Enterprise Information Systems 3, 2017, pp. 26–29.

[26] O. Franco-Bedoya, D. Ameller, D. Costal, X. Franch, Open source software ecosystems: a systematic mapping, Inf. Softw. Technol. 91 (2017) 160–185.

[27] A.R. Hevner, S.T. March, J. Park, S. Ram, Design science in information systems research, MIS Q. 28 (1) (2004) 75–105.

[28] R.J. Wieringa, Design Science Methodology for Information Systems and Software Engineering, Springer, 2014.

[29] P. Runeson, M. Höst, A. Rainer, B. Regnell, Case Study Research in Software Engineering - Guidelines and Examples, Wiley, 2012.

[30] H. Munir, J. Linåker, K. Wnuk, P. Runeson, B. Regnell, Open innovation using open source tools: a case study at sony mobile, Empir. Softw. Eng. 23 (1) (2018) 186–223.

[31] T. Alspaugh, W. Scacchi, et al., Ongoing software development without classical requirements, in: 21st International Requirements Engineering Conference, IEEE, 2013, pp. 165–174.

[32] P.J. Ågerfalk, B. Fitzgerald, Outsourcing to an unknown workforce: exploring opensourcing as a global sourcing strategy, MIS Q. (2008) 385–409.

[33] J. Kuriakose, J. Parsons, How do open source software (oss) developers practice and perceive requirements engineering? an empirical study, in: 2015 IEEE Fifth International Workshop on Empirical Requirements Engineering (EmpiRE), 2015, pp. 49–56.

[34] J.W. Castro Llanos, S.T. Acuña Castillo, Differences between traditional and open source development activities, in: O. Dieste, A. Jedlitschka, N. Juristo (Eds.), Product-Focused Software Process Improvement, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 131–144.

[35] W. Scacchi, Collaboration Practices and Affordances in Free/Open Source Software Development, in: Collaborative software engineering, Springer, 2010, pp. 307–327.

[36] T. Bhowmik, A.Q. Do, Refinement and resolution of just-in-time requirements in open source software and a closer look into non-functional requirements, J. Ind. Inf. Integr. (2018).

[37] N. Ernst, G.C. Murphy, Case studies in just-in-time requirements analysis, in: 2nd International Workshop on Empirical Requirements Engineering, IEEE, 2012, pp. 25–32.

[38] T. Bhowmik, N. Niu, P. Singhania, W. Wang, On the role of structural holes in requirements identification: an exploratory study on open-source software development, ACM Trans. Manage. Inf. Syst. 6 (3) (2015) 10:1–10:30.

[39] W. Stam, When does community participation enhance the performance of open source software companies? Res. Policy 38 (8) (2009) 1288–1299.

[40] K. Nakakoji, Y. Yamamoto, Y. Nishinaka, K. Kishida, Y. Ye, Evolution patterns of open-source software systems and communities, in: Proceedings of the International Workshop on Principles of Software Evolution, in: IWPSE '02, ACM, New York, NY, USA, 2002, pp. 76–85.

[41] M. Joblin, S. Apel, C. Hunsen, W. Mauerer, Classifying developers into core and peripheral: an empirical study on count and network metrics, in: 39th International Conference on Software Engineering, 2017, pp. 164–174.

[42] C. Jensen, W. Scacchi, Role migration and advancement processes in ossd projects: a comparative case study, in: 29th International Conference on Software Engineering, 2007, IEEE, 2007, pp. 364–374.

[43] I. De Noni, A. Ganzaroli, L. Orsi, The evolution of oss governance: a dimensional comparative analysis, Scand. J. Manag. 29 (3) (2013) 247–263.

[44] E. Capra, A.I. Wasserman, A framework for evaluating managerial styles in open source projects, in: B. Russo, E. Damiani, S. Hissam, B. Lundell, G. Succi (Eds.), Open Source Development, Communities and Quality, Springer US, Boston, MA, 2008, pp. 1–14.

[45] D. Riehle, The single-vendor commercial open course business model, Inf. Syst. e-Business Manag. 10 (1) (2012) 5–17.

[46] S. O'Mahony, The governance of open source initiatives: what does it mean to be community managed? J. Manag. Gov. 11 (2) (2007) 139–150.

[47] M. Germonprez, J.E. Kendall, K.E. Kendall, B. Young, Collectivism, creativity, competition, and control in open source software development: reflections on the emergent governance of the spdx working group, Int. J. Inf. Syst.Manag. 1 (1–2) (2014) 125–145.

[48] S. O'Mahony, Nonprofit Foundations and their Role in Community-firm Software Collaboration, in: Perspectives on Free and Open Source Software, MIT Press, Cambridge, MA, 2005, pp. 393–414.

[49] D. Riehle, The economic motivation of open source software: stakeholder perspectives, Computer 40 (4) (2007) 25–32.

[50] J. Lerner, J. Tirole, Some simple economics of open source, J. Ind. Econ. 50 (2) (2002) 197–234.

[51] S. O'Mahony, B.A. Bechky, Boundary organizations: enabling collaboration among unexpected allies, Adm. Sci. Q. 53 (3) (2008) 422–459.

[52] A. Nguyen Duc, D.S. Cruzes, G.K. Hanssen, T. Snarby, P. Abrahamsson, Coopetition of software firms in open source software ecosystems, in: A. Ojala, H. Holmström Olsson, K. Werder (Eds.), Software Business, Springer International Publishing, Cham, 2017, pp. 146–160.

[53] B. Lundell, B. Lings, E. Lindqvist, Open source in swedish companies: where are we? Inf. Syst. J. 20 (6) (2010) 519–535.

[54] J. Henkel, Champions of revealing-the role of open source developers in commercial firms, Ind. Corp. Change 18 (3) (2008) 435–471.

[55] D. Riehle, Controlling and steering open source projects, Computer 44 (7) (2011) 93–96.

[56] D.J. Teece, Business models, business strategy and innovation, Long Range Planning 43 (2–3) (2010) 172–194.

[57] S. Shahrivar, S. Elahi, A. Hassanzadeh, G. Montazer, A business model for commercial open source software: a systematic literature review, Inf. Softw. Technol. 103 (2018) 202–214.

[58] J. Linåker, H. Munir, K. Wnuk, C.E. Mols, Motivating the contributions: an open innovation perspective on what to share as open source software, J. Syst. Softw. 135 (2018) 17–36.

[59] J. West, S. Gallagher, Challenges of open innovation: the paradox of firm investment in open-source software, R&D Manag. 36 (3) (2006) 319–331.

[60] J. Henkel, Selective revealing in open innovation processes: the case of embedded linux, Res. Policy 35 (7) (2006) 953–969.

[61] C.M. DaSilva, P. Trkman, Business model: what it is and what it is not, Long Range Planning 47 (6) (2014) 379–389.

[62] H. Chesbrough, M. Appleyard, Open innovation and strategy, California Manag.Rev. 50 (1) (2007) 57–76.

[63] J. West, S. O'Mahony, The role of participation architecture in growing sponsored open source communities, Ind.Innov. 15 (2) (2008) 145–168.

[64] A. Gawer, Bridging differing perspectives on technological platforms: toward an integrative framework, Res. Policy 43 (7) (2014) 1239–1249.

[65] S. Jansen, S. Brinkkemper, J. Souer, L. Luinenburg, Shades of gray: opening up a software producing organization with the open software enterprise model, J. Syst. Softw. 85 (7) (2012) 1495–1510.

[66] A. Baars, S. Jansen, A framework for software ecosystem governance, in: M.A. Cusumano, B. Iyer, N. Venkatraman (Eds.), Software Business, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, pp. 168–180.

[67] G. Valença, C. Alves, A theory of power in emerging software ecosystems formed by small-to-medium enterprises, J. Syst. Softw. 134 (2017) 76–104.

[68] L. Nyman, J. Lindman, Code forking, governance, and sustainability in open source software, Technol. Innov. Manag. Rev. 3 (1) (2013) 7–12.

[69] J. Gamalielsson, B. Lundell, Sustainability of open source software communities beyond a fork: how and why has the libreoffice project evolved? J. Syst. Softw. 89 (2014) 128–145.