Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Problems, Solutions, and Success Factors in the openMDM User-Led Open Source Consortium

Elçin Yenişen Yavuz

Computer Science Department,

Friedrich-Alexander-University Erlangen-Nürnberg

elcin.yenisen@fau.de

Ann Barcomb

Schulich School of Engineering,

University of Calgary

Dirk Riehle

Computer Science Department,

Friedrich-Alexander-University Erlangen-Nürnberg

Abstract:

Open-source software (OSS) development offers organizations an alternative to purchasing proprietary software or commissioning custom software. In one form of OSS development, organizations develop the software they need in collaboration with other organizations. If the software is used by the organizations to operate their business, such collaborations can lead to what we call “user-led open-source consortia” or “user-led OSS consortia”. Although this concept is not new, there have been few studies of user-led OSS consortia. The studies that examined user-led OSS consortia did so through the lens of OSS, but not from the inter-company collaboration perspective. User-led OSS consortia are a distinct phenomenon which


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

share elements of inter-company collaboration, outsourcing software development, and vendor-led OSS development and cannot be understood by using only a single lens. To close this gap, we present problems and solutions in inter-company collaboration, outsourcing, and OSS literature, and present the results of a single-case study. We focus on problems in the early phases of a user-led open-source consortium, the openMDM consortium, and the solutions applied to these problems. Furthermore, we present the factors which lead this consortium to sustained growth.

Keywords: Open source software, collaborative software development, open source user-led consortia, open source foundations, community source, Eclipse Foundation, success factors, outsourcing

This manuscript underwent [editorial/peer] review. It was received xx/xx/20xx and was with the authors for XX months for XX revisions. [firstname lastname] served as Associate Editor. **or** The Associate Editor chose to remain anonymous.]


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

1 Introduction

Organizations are involved in open-source software (OSS) in different ways. Some organizations open source their internally developed code and create a community around it (West & O'Mahony, 2005; Dahlander & Magnusson, 2005; Dahlander, 2007; Harutyunyan et al., 2020), and some contribute to the development of OSS projects through sponsorship, such as providing infrastructure, marketing support, financial support, or developer support (Berdou, 2006; Zhou et al., 2016).

Some organizations contribute or lead OSS projects with the aim of profiting from the software commercially (Dahlander, 2007). This model, with exactly one stakeholder company, is called single-vendor open source (Riehle, 2010; Schaarschmidt et. al., 2011). If multiple organizations collaborate, these collaborations often lead to open-source consortia or foundations. One type of these foundations is the vendor-led foundation, in which software vendors collaborate to develop the software they base their products on (Schaarschmidt et. al., 2011; Riehle & Berschneider, 2012). In another type of OSS foundation, organizations collaborate to develop software for the organizations' own use; the intent is firm-internal use of the software. We call these types of foundations "user-led open-source consortia", or "user-led OSS consortia". Consortium here means a formally organized community of organizations with a defined governance structure and processes. The specific incorporation, if any, does not matter. We therefore use the terms foundation and consortium synonymously. We present the hierarchy and categorization of open-source foundations' terms regarding the leading roles in Table and in Figure 1, respectively. Our area of focus, user-led open-source consortia, is shown with a gray background.

Communications of the Association for Information Systems

Research Article ISSN: 1529-3181

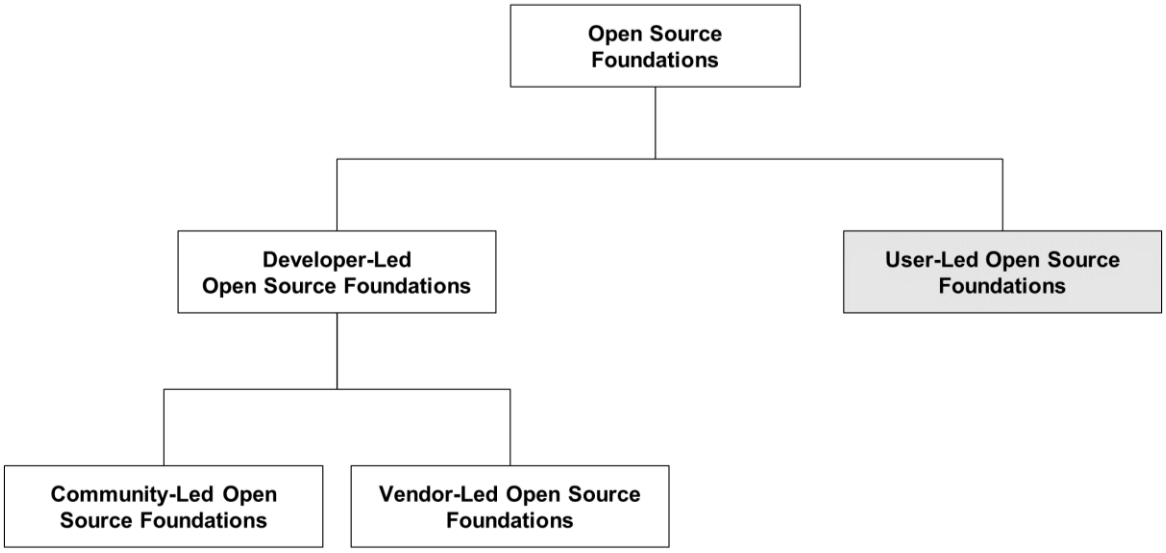

Figure 1. Hierarchy of Types of Open Source Foundations Regarding the Leading Roles


Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Table 1. Categories of Open Source Foundations

Category	Common definition	Example
Open Source Foundations	General term for all types of foundations which host OSS development projects.	Linux Foundation
Developer-Led Open Source Foundations	General term for open source foundations which host OSS development projects steered by software developers (individual or organizational)	Mozilla Foundation
Community-Led Open Source Foundations	Developer-led open source foundations which host OSS development projects which are initiated and managed by individuals or groups of individuals.	The Apache Software Foundation
Vendor-Led Open Source Foundations	Developer-led open source foundations which host OSS development projects which involve software development companies and individual developers (either volunteer or paid).	The Open Infrastructure Foundation (Project: OpenStack)
User-Led Foundations	Foundations which host OSS development projects which are initiated and steered by software user organizations with the purpose of their own use.	Apereo Foundation

A user-led open-source consortium is a consortium of organizations who sponsor and steer the development of OSS that they need to operate their business. Organizations who are not generally engaged in selling software products or services are key stakeholders and often founders. Software the organizations develop collaboratively doesn't provide competitive differentiation among members. Sponsorship involves paying for development in any combination of outsourcing and allocating employees to work on the software. Those who lead the consortium are user organizations: software consumers. Individuals are not the intended beneficiaries of the consortium, and might be permitted to contribute, depending on how the consortium is structured.


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

Developing OSS helps organizations avoid vendor lock-in, establish de-facto standards, and reduce costs for customization and training (Wheeler, 2004; West & Gallagher, 2006; Liu et al., 2008). As an alternative to using proprietary software or developing the software alone, user-led open source consortia reduce costs while allowing organizations to acquire software that meets their needs (Liu et al., 2008; Liu et al., 2014).

Early examples of user-led OSS consortia were seen in higher education (e.g., Kuali, Sakai). Recently, interest in user-led OSS consortia has increased among commercial organizations. In our ongoing subsequent research, we counted over 56 distinct user-led OSS consortia in different industries. Some of these user-led open-source consortia established their own foundations to govern their collaboration (e.g., GENIVI Alliance), and others joined already established umbrella foundations such as the Linux Foundation (LF) or Eclipse Foundations (EF).

There are two focal features of user-led OSS consortia. First, the development is driven by user organizations, not by individual volunteers or software vendors. Being a software consumer (user) and being a software vendor (producer) lead to differences in governance, project management and maintenance processes. Second, software is developed primarily for the organizations' own use instead of being part of a commercial software end-product for the leading organizations of the consortium (Liu et al., 2007; Wheeler, 2007; Baldwin & von Hippel, 2011). These characteristics are–what distinguish user-led open-source consortia from vendor-led open-source foundations, which are driven by vendor organizations, and typically for the purpose of developing non-differentiating software which can be used as a base for or in commercial software offerings (West & O'Mahony, 2005; Riehle, 2010; Germonprez et al., 2013).

To date, research about user-led OSS consortia is limited, and primarily focused on early examples in higher education, such as the Kuali and Sakai projects (e.g., Liu et al., 2012), with a focus on describing the phenomenon, rather than developing an overview of the problems and solutions which differentiate user-led OSS consortia from other collaborations. Later work examined other industries, using the OSS


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

literature to understand user-led OSS consortia (e.g., Schwab et al., 2020). In this paper, we drew on the literature from user-led OSS consortia, inter-company collaboration, open source software (OSS) development, and outsourcing literature, and anticipated the possibility of finding additional problems arising from the lack of familiarity with OSS development methods, which can exist in both consortium members as well as the vendors they hire.

This study addresses this gap in how user-led OSS consortia can be best guided, through an exploratory single-case study research. Our goal was to understand problems faced and solutions to these problems found by investigating a successful consortium. We chose the openMDM consortium, which is an automotive industry working group. openMDM is not incorporated as its own legal entity, but rather takes the form of an Eclipse Working Group (EWG), which is an Eclipse Foundation (EF) concept that provides a working group with all the independence of a stand-alone non-profit organization, including its own bylaws, without incorporation. Our research was guided by the following questions:

> **RQ1:** What problems occur in a user-led open source consortium?
>
> **RQ2:** What are solutions to the problems which occur in a user-led open source consortium, and which factors lead to success?

To address these questions, we opted to conduct a single-case study research (Eisenhardt, 1985; Yin, 2018). Our work involved collecting multiple sources of evidence (public documents, meeting minutes, interviews with the key informants of the consortium), and qualitatively analyzing the data. In case study research, unlike in action research, the researchers do not actively engage or intervene in the case. We subsequently confirmed the accuracy of our findings by presenting them at the annual meeting of openMDM in 2019.

The contributions of this paper are:

- We perform case study research on a novel industry phenomenon where little work has been presented before.

Research Article ISSN: 1529-3181

- We precisely identify and present the 13 problems and 22 solutions which are faced during the starting and growing phases of a user-led open-source consortium.

- We relate these problems and solutions and support the results with findings from the literature.

The rest of this paper is structured as follows: In Section 2 related work is reviewed. Section 3 describes the research method and provides background information about the case of openMDM. Section 4 presents the results of our research. Discussion about our findings and suggestions for future work are presented in Section 5. Limitations are discussed in Section 6. Finally, Section 7 concludes this paper.

2 Related Work

There are four main areas of literature relevant for the topic: existing work on user-led OSS consortia, problems and solutions of company collaborations, problems and solutions of OSS development projects, and problems associated with outsourcing software development.

2.1 User-Led Open Source Consortia

The first examples of user-led OSS consortia came from higher education. uPortal, Sakai, Kuali, Open Source Portfolio Initiative are early examples of user-led OSS consortia, which have been also called “community source” (Wheeler, 2004). Since the term “community source” can cause confusion with the community-led OSS projects—that is, OSS projects which are led by a community of developers, —we use the term “user-led OSS consortia” in our research. Other consortia which have been studied include openKonsequenz (Schwab et al., 2020), and openMAMA (Levy & Germonprez, 2015). Schwab et al. (2020) investigate the ecosystem and motivation of actors in the involvement of openKonsequenz user-led OSS consortium. Levy & Germonprez (2015) present the innovation intermediaries in openMAMA community.

Liu et al. performed studies specifically around the “Kuali” case. They have focused on the technical features of the developed product, partners’ motivations and decision criteria applied for joining

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

consortium, labor changes over the project life-cycle, and evolution of the consortium (e.g., Liu et al., 2007; Liu et al., 2008; Liu et al., 2010; Liu et al., 2012; Liu et al., 2014; Liu et al. 2020).

Several problems were identified in the Kuali studies. One concerns the development process management aspects. Liu et al. (2010) suggest that when a user consortium has a large number of partner institutions which have developers with unbalanced competencies, the consortium faces problems of coordination and development management. A proposed solution is outsourcing software development (Liu et al., 2010). Another problem is differing expectations among partner organizations, meaning software development teams receive requirements from multiple stakeholders. Flexible software architecture is a possible solution (Liu et al., 2012). Other problems are seen during the expansion phase of the consortium with an increased number of projects, participants and commercial affiliates are explained by Liu et al. (2020). These are the difficulties in community governance and coordination, which is addressed by overseeing governance by a single unit (foundation); balancing the competition between the involved commercial affiliates, maintaining the family atmosphere among the participants of the consortium, and lack of knowledge sharing between different development projects. These problems, which are related to growth of a user-led OSS consortium, are addressed through modular organizational structure.

2.2 Problems and Solutions of Inter-Company Collaborations

In user-led OSS consortia, two or more companies collaborate to develop OSS to operate their internal processes. In order to understand the collaboration dynamics in user-led OSS consortia better, we examined the inter-company collaboration literature.

Inter-company collaborations can suffer from a number of problems. When the partners are lacking technological capability, financial resources, and due diligence, it can lead to friction, particularly if the partners differ in these aspects (Fortuin & Omta, 2008; Kelly et al., 2002). Asymmetry between partners in other aspects also negatively affects collaboration success. When the companies are very different in size, the smaller companies may fear losing independence (Fortuin & Omta, 2008). Fear and distrust can


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181
also arise from cultural differences, due to misunderstandings, lack of openness, and untimely information flow (Dacin et al., 1997; Fortuin & Omta, 2008). When ground rules, roles, and responsibilities are not defined clearly, and the potential benefits of the collaboration are not explicit, problems are more likely to occur (Kelly et al., 2002; Fortuin & Omta, 2008).

Solutions depend on the specific problems, but the literature has identified numerous success factors in collaboration. In order to provide a general overview of the topic of solutions and success factors in collaboration, and to highlight concepts which may be relevant to user-led open source consortia, we focused on six systematic literature reviews in different domains: collaborations and co-opetition in general, strategic partnerships, and collaborations in Information Technology. By comparing and mapping findings of these six studies, we identified 92 success factors. Eighty-two of these factors were mentioned only in a maximum two of these six studies. We present a complete mapping of the success factors in (Yenişen Yavuz et al., 2022). In Figure 2, we present seven factors and their explanations which were found at least in three studies. These factors are: partner selection, common goals, ground rules, equality, regular progress reviews, top management commitment, and collaboration champions.

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Factor	Explanation	Reference
Partner selection	Choosing partners which have mutual understanding, trust, complementary expertise, and strengths, increases the success potential of a collaboration from the beginning	Mattessich & Monsey, 1992; Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Chin et al., 2008; Rikkiev & Mäkinen, 2009
Common goals	Defining common and concrete objectives agreed upon by all parties increases the success potential of a collaboration and reduces the risk of failure	Mattessich & Monsey, 1992; Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Chin et al., 2008; Rikkiev & Mäkinen, 2009
Ground rules	Setting clearly defined rules, responsibilities, and procedures at the beginning of the collaboration helps to avoid conflicts in later phases	Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001
Equality	Having equality, in terms of resource sharing and decision making in the community, strengthens the commitment to the collaboration	Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Rikkiev & Mäkinen, 2009
Regular progress reviews	Periodic reviews increase the success potential of a collaboration by easing the information sharing, helping to monitor the collaborators' performance, reducing the likelihood of conflicts and enabling timely adjustments of collaboration strategy	Bruce et al., 1995; Hoffmann & Schlosser, 2001; Chin et al., 2008
Top management commitment	Gaining top management support ensures having sufficient resources in a collaboration which increases the success potential	Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Rikkiev & Mäkinen, 2009
Collaboration champions	Having mentors who support the collaboration and help to overcome difficulties by guiding involved people increases the potential for the success of a collaboration	Bruce et al., 1995; Rai et al., 1996; Rikkiev & Mäkinen, 2009

Figure 2. Success Factors of Inter-Company Collaborations

2.3 Problems and Solutions of OSS Projects

In user-led OSS consortia participants follow the basic principles of OSS development, such as openness, transparency, collaborative working, and resource sharing. In order to understand the development

Research Article ISSN: 1529-3181

process and ecosystem in user-led OSS consortia, we provide an overview about the problems and solutions faced in OSS development practices. A summary of the problems and solutions identified in the literature are presented in (Yenişen Yavuz et al., 2022).

The developer community, source code, and coordination mechanisms are key resources of OSS projects which affect success (Chengalur-Smith et al., 2010, Midha & Pavia, 2012; Sagers, 2004). In this section, we explain the problems and solutions associated with these three components of OSS project success.

Developer community. Developer community in OSS consists of core developers, who contribute mainly the source code and regulate administrative responsibilities, and peripheral developers, who make contributions and help to improve the quality of the code by reporting bugs, submitting patches, and developing new features (Lee & Cole, 2003; Crowston & Howison, 2005). The sustainability of OSS projects often depends on an active developer base (Colazo & Fang, 2010). Inactive developers, developers' turnover, having too many developers, and barriers to entry for newcomers are the problems we associate with the developer base. When developers, especially core developers, stop contributing to a project without communicating their intention, the quality and sustainability of the project is affected because their absence is not always immediately observed (Michlmayr, 2004). Developers' departure and arrival to projects has a negative effect on the quality of a team's work and quality of the modules (Foucault et al., 2015). High developer turnover can also cause problems in a project, due to knowledge loss and experience gaps (Rashid et al., 2017). On the other hand, having a greater number of active developers in a project means an increase in activity and creates a need for more effort for project management and coordination (Midha & Palvia, 2012). Finally, barriers to entry can prevent new developers from joining in the first place, reducing the pool of potential core contributors (von Krogh et al., 2003; Steinmacher et al., 2015a).

The problem of inactive developers can be reduced by managing to avoid burnout, educating new maintainers on the importance of communicating when they are unable to fulfill responsibilities, and making personal inquiries into their well-being (Michlmayr, 2004; Bacon, 2012). Developing a high degree of connectedness and increasing social connection between the community participants increases their

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

likelihood of staying in the community and continuing their contributions (Maruping et al., 2019). Both developer turnover and barriers to entry can be reduced by developing modular code and ensuring good documentation (Midha & Palvia, 2007; Steinmacher et al., 2015a; Barcomb et al., 2018; Barcomb et al., 2020). Additionally, turnover can be reduced by balancing writing new code with maintaining old code, and balancing writing code and writing documentation (Lin et al., 2017). Providing public guidelines for the community about "welcoming new contributors" and providing contributor guide to newcomers about "first contributions" may help to lower the entry barriers for newcomers (Lumbard et al., 2020). Providing social support, mentorship, classifying tasks based on their complexity, are additional techniques for aiding newcomers (Qureshi & Fang, 2011; Ducheneaut, 2005; Steinmacher et al., 2015b, Riembauer et al., 2020), while social norms, satisfaction, and community commitment contribute to participants' intention to remain (Barcomb et al., 2019). Providing a short path for information flow, which means having a small number of developers as intermediaries in the information or knowledge transfer, increases the speed of transfer, and minimizes the information decay in communities with a large number of developers (Singh, 2010).

Source Code. Ultimately, OSS requires software—source code—to exist. Problems involving source code include the lack of maintainability, quality, and interoperability. Maintainability is especially important for OSS, because the primary activity in OSS projects is the production of new versions of existing software (Yu et al., 2012). Quality and internationalization are important to increase user satisfaction and popularity of the project (Conley & Sproull, 2009; Radtke et al., 2009; Midha & Palvia, 2012). Finally, due to the large number of stakeholders, OSS offers particular opportunities for interoperability, which in turn is of high importance to industry and government alike (Almeida et al., 2011).

A modular structure helps to increase maintainability because it allows parallel, decentralized, and incremental development (Feller & Fitzgerald, 2000; Narduzzo & Rossi, 2005; Midha & Palvia, 2012). Coordination through open superposition (process of incrementally developing open source software components by layering development tasks independently) can improve the quality and the popularity of a project (Howison & Crowston, 2014; Medappa & Srivastavaa, 2019). Code quality can be improved by

Research Article ISSN: 1529-3181

time-based releases, testing, peer review, version control, reducing complexity and public discussion of issues (Rigby et al., 2008; Conley & Sproull, 2009; Mauerer & Jaeger, 2013; Michlmayr et al., 2015; Geiger et al., 2021). Effective bug fixing activities have a positive influence on project success (Singh, 2010). Lastly, interoperability is assisted by having a well-defined API, which allows developers to reuse the software without fully understanding the source, and by integration testing (Haefliger et al., 2008).

Social interaction and coordination. Effective coordination increases user and developer satisfaction, and improves the success of software projects (Sagers, 2004). Geographically distributed software development, lack of diversity, toxic contributors, and companies' bad behaviors are the problems associated with OSS coordination. OSS is often characterized by geographically distributed development which leads to difficulties in building trust between team members due to a lack of face-to-face meetings, a lack of informal communication, or a lack of established relationships (Herbsleb & Grinter, 1999; Piccoli & Ives, 2003; Damian, 2003; Henkel, 2004; Filippova & Cho, 2015; O'Leary et al., 2020). Diversity in OSS communities leads to knowledge of a greater array of topics, higher creativity, and increase in team productivity, but achieving greater diversity is frequently not a priority, and can even be hindered by the widespread belief among OSS developers in meritocracy (Castilla & Benard, 2010; Nafus, 2012; Daniel et al., 2013; Vasilescu et al., 2015; Bosu & Sultana, 2019). People who create a toxic atmosphere in the project can lead to unproductivity and contribute to attrition (Carillo & Marsan, 2016; Guizani et al., 2021). Although having an organizational sponsor is accepted as a sign of having technical support and sustainability of the software, companies' bad behaviors—such as trying to control influence the development process for their own interest, free-riding, conflict between companies, and code dumping—lead to conflicts or extra works in OSS communities (Dahlander & Magnusson, 2005; Stewart et al., 2005; Ciesielska & Westenholz, 2016; Zhou et al., 2016; Ehls, 2017; Pinto et al., 2018; Kochhar et al., 2019, Weikert et. al., 2019, Geiger et al., 2021).

Transparent, asynchronous, and open communication address the difficulties of distributed development approach by allowing community members to see the past work activity and project history (Tsay et al., 2014; Riehle, 2015; Riembauer et al., 2020). Proposals to address a lack of diversity include providing a

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

social platform in the OSS communities for minoritized groups to share their experiences and motivate each other, and having a code of conduct (Bosu & Sultana, 2019; Singh & Brandon, 2019). Toxic people can be addressed by identifying their concerns, discussing them, and developing an evidence-based summary of poor behavior if the problem persists (Bacon, 2012; Fogel, 2005). Providing guidelines for an inclusive communication space is a further strategy for avoiding toxic environments (Guizani et al., 2021). To avoid too much control by companies in the OSS environment, conflict management strategies are necessary. This can be combined with code modularity, parallel development approach and allowance of the forking of projects to provide freedom for companies and developers (van Wendel de Joode, 2004).

2.4 Problems with Outsourcing Software Development

In user-led OSS consortia, organizations either devote their own employees to the software development project or sponsor development effort by outsourcing to software vendors, which develop the code on their behalf.

When companies outsource software development, as is often the case in user-led open source consortia, there is the threat of opportunism, or “specific acts of self-interest seeking with guile” (Williamson, 1993) on the part of the vendor, in addition to strategic risks. Risks include shirking (deliberate under-performance of a vendor), behavior unobservability (inability to assess the performance of the vendor), poaching (misuse of information acquired during the contract after termination of the contract), opportunistic renegotiation (renewal of the contract in favor of the vendor due to lock-in), difficulty with project or requirements specifiability, project complexity, and requirements volatility (Tiwana & Bush, 2007; Aron et al., 2005). In the context of an open source user-led consortia, the risks of poaching and opportunistic renegotiation are reduced; avoiding lock-in is often a motivation for the initiative (Courant and Griffiths, 2006).

Companies can use a portfolio of control mechanisms, which contain both formal and informal mechanisms, to reduce outsourcing problems. Formal mechanisms include outcome controls and behavior controls, such as progress reports. Informal controls are social mechanisms to align the client

Research Article ISSN: 1529-3181

and vendor goals, which can be achieved through regular meetings and long-term alliances, and self controls, which can be achieved by working with timetables and milestones (Choudhury & Sabherwal, 2003). Shirking can be addressed through close monitoring and outsourcing the same job to two or more vendors and dispensing with the worst performer (Aron et al., 2005). Contracts with detailed roles and responsibilities, including monitoring procedures and penalties, can help mitigate opportunism (Barthélemy & Quélin, 2006). Precisely specifying functional requirements can help avoid specification issues (Choudhury & Sabherwal, 2003).

3 Research Method

We followed an exploratory single-case study research approach. We chose case study research in order to investigate a user-led consortium's problems and solutions in a natural setting and focus on actual events (Benbasat et al., 1987; Yin, 2018). The methodology of the research is adopted from the theory building framework suggested by Eisenhardt (1989). After defining our research questions, we chose our sample case purposefully. We triangulated our data by using multiple sources of evidence: (1) documentation of meeting minutes, administrative documents, news and information on the consortium's website; (2) interviews and (3) analysis of another user-led open-source consortium. We conducted data collection and analysis steps iteratively. During the theory building phase, we searched for evidence of the relationships between the constructs and compared our findings with similar literature.

3.1 Sampling Model and Sample Selection

We performed purposeful sampling (Patton, 1990) by determining the relevant dimensions, describing the population based on these dimensions, and selecting a project which met our criteria.

As the first step of the sample selection, we created a sampling model by listing possible user-led OSS consortia which we found by online search. We filled the model with the specifications of each consortium based on our dimensions.


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Industry was important because the majority of studies on user-led OSS consortia have been drawn from a single industry, the higher education industry, which may limit generalizability. We considered maturity because we expect that success can only be evaluated after a certain maturity. Senyard & Michlmayr (2004) present three stages of OSS development: cathedral, transition and bazaar phases. Cathedral is the initial phase of the development process which is led by one individual or a group of core developers. In the bazaar phase, besides the initial developers, a community of users (both non-technical and technical) are actively contributing to the code, the quality of the software increases due to modular development and parallel review of the code. Transition phase is the stage between the cathedral and bazaar phases, a phase in which decisions are made about the distribution of the code, license choice, management style to attract others for contributions. Considering Senyard & Michlmayr (2004)'s approach, we evaluated the maturity based on its developer and user community and stage of the product development. Finally, we included activity, on the grounds that high activity is a sign of project success (Crowston et al., 2003). Although not a dimension, an additional factor was the availability of public, online data, and the willingness of participants to engage with us.

The active projects we found were from the industries of automotive, agriculture, culture, energy, entertainment, finance, geospatial, higher education, infrastructure, tourism, and transportation industry. We described maturity as early (having an initial code without a product release and core team of organizations), growing (having a released product with a team of developer organizations and users), or mature (having an established product with regular releases and an ecosystem with developers, users, and contributors). We examined code repositories to evaluate the amount of activity and the frequency of releases in order to assess activity.

We selected a project from the automotive industry, with growing maturity and high activity: openMDM. Additionally, there was significant online documentation available, and consortium members were willing to support the research. openMDM is described with reference to the selection criteria in Table 2.

To supplement our findings about openMDM, we included a mature user-led OSS consortium from the higher education industry: Sakai. The Sakai Project was initially established in 2004 by four universities in

Communications of the Association for Information Systems

Research Article ISSN: 1529-3181

the United States with the goal of developing an open source learning management system (LMS) mainly for their own use. As of 2019, approximately 300 universities around the world are using Sakai LMS[1]. We did not conduct a full analysis of Sakai, but rather compared it to our findings of openMDM.

The purpose of this comparison was to help identify the extent to which the openMDM findings also apply to user-led consortia observed in the higher education industry, and to help distinguish which aspects of openMDM are likely to be specific to this collaboration, and which are more likely to be universal. Basic information about Sakai is also given in Table 2.

Table 2. Sampling Dimensions

	openMDM		**Sakai**	
Dimension	**Summary**	**Details**	**Summary**	**Details**
Industry	Automotive	software development project about measured data management	Higher education	software development project about learning management system (LMS)
Maturity	Growing phase	• released product • ecosystem with leading organizations and vendors • not yet industry standard	Mature	• established product with developers and users from multiple universities • ecosystem with user organization, vendors, contributors

[1]https://www.apereo.org/projects/sakai-lms/2018-2019-software-community-health-metrics-sakai-lms#bookmark=id.l3omy8i978gh

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Activity	Active	regular code contributions & periodic releases (since July 2017, yearly at least 6 releases; between July 2017 and May 2020, 19 releases)[2]	Active	regular code contributions & periodic releases (since July 2017, yearly at least 3 releases; between February 2017 and May 2020, 15 releases)[3]

3.2 Data Collection

The data collection process started in November 2018 and lasted until June 2019. We investigated the case of openMDM consortium for its activities between the years 2014 and 2019. Qualitative data from multiple sources of evidence were collected for the purpose of data triangulation. We collected documentation in the form of meeting minutes, website content, Eclipse Wiki, Jira tracking tool, and email archives. The insights of the people leading the openMDM Eclipse Working Group (EWG) were collected in the form of semi-structured interviews. Furthermore, we included data from another user-led open-source consortium, Sakai, in order to triangulate our findings. Detailed information and data sources of Sakai are listed in (Yenişen Yavuz et al., 2022).

The search for archival data was conducted online and iteratively. We present the steps of the data collection process in Figure 3. After the initial analysis of the collected documents, we conducted a prolonged-interview which lasted for three hours with the Managing Director of Eclipse Foundation (EF) Europe GmbH, and a semi-structured interview with the toolkit manager of the openMDM consortium. Our interview partners were the key informants of the consortium. The managing Director of EF Europe has been involved in the project from the beginning of the consortium establishment process, in 2012. The toolkit manager started working on the project in 2018, when major problems arose and solutions were

[2] https://projects.eclipse.org/projects/automotive.mdmbl/downloads
[3] https://www.apereo.org/projects/sakai-lms/news

Communications of the Association for Information Systems

Research Article ISSN: 1529-3181

needed to improve the consortium progress. With the toolkit manager's involvement, the consortium started to solve their problems. The interview protocol is presented in (Yenişen Yavuz et al., 2022).

In total, 86 distinct documents were evaluated. (Yenişen Yavuz et al., 2022) contains a complete list of all public documents. In the rest of this paper, we refer to data sources as follows:

- "A" for annual meeting presentations and minutes
- "I" for interview transcripts
- "G" for guidelines
- "L" for legal documents (e.g., participation agreement)
- "M" for meeting minutes
- "S" for Sakai data sources

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Step 1: Collecting general information

Action	Source	Collected documents (units)
Collecting basic information about the consortium	openMDM official website: https://www.openmdm.org	History, Governance, News, Members pages (4); Participation Agreement (1); Charter (1)

Step 2: Collecting detailed information

Action	Source	Collected documents (units)
Collecting public data for analysis	Eclipse Wiki: https://wiki.eclipse.org/Open-Measured-Data-Management-WG	Annual Meeting Documents (2); Steering Committee Meeting Minutes (8); Architecture Committee Meeting Minutes (45); Quality Committee Meeting Minutes (9); Workshops and toolkit management slides (3); Guidelines (2)

Step 3: Snowballing

Action	Source	Collected documents (units)
Collecting documents mentioned in the meeting minutes	openMDM email list (Eclipse: list: open-measured-data-wg) and Jira Tracking Tool	Annual Meeting Documents (3); SC Meeting Minutes (3); Working Group Protocol (2)

Step 4: Data triangulation

Action	Source	Collected documents (units)
Performing interviews	Interviews	Interview transcripts (2)
Drawing parallels with an additional consortium (Sakai)	Internet search & interviews	Research papers, book, blog posts and interviews about Sakai (38)

Figure 3. Data Collection Process


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

3.3 Data Analysis

We performed qualitative data analysis by using the MaxQDA data analysis tool.[4] We used the coding paradigm of grounded theory, as defined by Strauss & Corbin (1990) and performed open, axial, and selective coding. The data analysis process was conducted in the following manner:

Step 1. We collected publicly available documents from the oldest to the newest, read and analyzed them. We searched for missing documents, iteratively. We performed open coding by labelling important points in documents.

Step 2. After open coding, we created higher categories by relating sub-categories. We sought a deep understanding about the structure and dynamics of the consortium. For this reason, we grouped sub-categories into the following axial categories: consortium structure, Steering Committee (SC) duties, SC decisions, Architecture Committee (AC) duties, toolkit management, Quality Committee (QC), communication, and documentation.

Step 3. In this phase, we created selective codes related to problems and solutions categories. Based on these categories we built the base of our theory and shared our preliminary results with the consortium members.

Step 4. After finding out major problems and solutions in the consortium, we performed interviews, and coded the transcripts into our final code segments. The final code system is presented in (Yenişen Yavuz et al., 2022).

Step 5. We presented our findings at the openMDM General Assembly 2019 to the SC members as a form of member checking (Guba, 1981). As the response to our analysis was positive, we made only minor refinements following the presentation.

Going beyond the openMDM case, we analyzed parallels with another consortium, Sakai, for data triangulation.

---

4 https://www.maxqda.com/

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

3.4 Background of the Case of openMDM Consortium

openMDM is a user-led open source consortium established by the automobile companies Audi, BMW, and Daimler, and the service providers HighQSoft, Gigatronik, Canoo Engineering, Science+Computing, and Peak Solutions in 2014. The goal of the consortium is "promoting the development and distribution of open source tools for measurement data management based on the ASAM (Association for Standardization of Automation and Measuring Systems) ODS (Open Data Services) standards" (W3).

The origin of the openMDM, the MDM (Measured Data Management) project, dates to 1999. The MDM project was initiated by Audi AG as a software development project for their internal use. In 2008, Audi AG open sourced the developed software for other vehicle manufacturers and suppliers (W1). Over time, the users of the software started demanding new functionalities and a demand for an equal partnership, with equality of both decision making and funding, arose (I1). In 2012, Audi and Eclipse Foundation started conversations about the possible structure of the consortium and in 2014 the openMDM consortium became an Eclipse Working Group (I1).

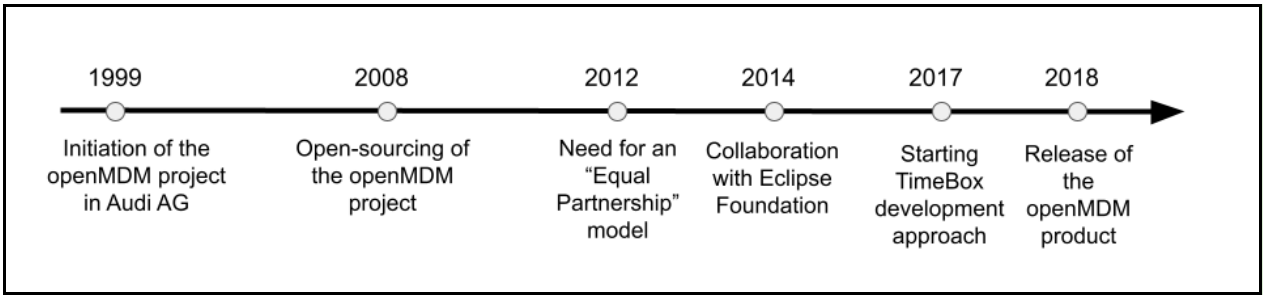

Figure 4. Milestones of the openMDM

The openMDM consortium has five membership types: drivers, service providers, application vendors, users, and guests. At founding, the driver members of the community were Audi, BMW, and Daimler. In 2015 and 2016, respectively, Müller-BBM and Siemens joined the community as driver members. A number of service provider members joined and left the consortium between the years 2014 and 2019. As of 2018, Gigatronik, Peak Solutions, Karakun are the main service providers. HighQSoft is the only

Research Article ISSN: 1529-3181

application vendor member of the consortium. The consortium has one user member, TATA, and one guest member, ASAM.

Driver members and user members are both the potential users of the openMDM software. The difference is that driver members invest more resources in the project and have more influence on the direction of the development. Guest members are mainly research and development partners, academic entities, and potential future full-fledged members. They do not have voting rights. Service provider members offer services for deployment, development, and maintenance of the software, while application vendor members aim to use the components of this software in their products (L1).

4 Research Results

This section presents the results of this case study research. During the coding process, we grouped the problems and solutions in four categories: consortium management, project management, software, and external factors. We present results in subsections based on these categories.

4.1 Problems of a User-Led Open Source Consortium

Our RQ1 is: “What problems occur in a user-led open-source consortium?” We addressed this question by identifying the problems openMDM faced from 2014 to 2019. As the results of our open, and axial coding process, we identified 13 main problems, and at the end of selective coding, we grouped them in four categories: consortium management, process management, user management, and external factors. Some of these problems have already been solved, and some were still in progress when the research was conducted. We present these problems and their occurrences in time, and map them to the literature in Table 3.

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Table 3. Problems in openMDM

Category	Code	Problem	Observed period	Data source	Reference in literature
Consortium Management	PC.1	Slow return on investment (ROI)	2017	I1, A2, L2	
	PC.2	Turnover in service provider members	2015 - 2018	I1, A1, A2, A3, A5	Rashid et al., 2017; Foucault et al., 2015
	PC.3	Non-user-friendly website	2014 - ongoing as of June 2019	I1, W4	Riembauer et al., 2020
	PC.4	• Lack of promotion • Low involvement in conferences • Lack of user stories	2016 - ongoing as of June 2019	I1, M3, M4	Ågerfalk & Fitzgerald, 2008
	PC.5	Low number of users	2018 - ongoing as of June 2019	I1, I2	Radtke et al., 2009
	PC.6	Low number of driver members	2018 - ongoing as of June 2019	A1, A3	Shaikh et al., 2009
	PC.7	Lack of financial resources	2018 - ongoing as of June 2019	I1, A2, A3, A5	Fortuin & Omta, 2008; Kelly et al., 2002
Process Management	PP.1	Split development responsibility without a consortium wide authority	2014 - 2016	I1, A2, A6	Boldyreff et al., 2004; Liu et al., 2010; Tiwana & Bush, 2007; Aron et al., 2005
	PP.2	Integration problems	2016 - 2018	M35, M44, M45, M46, I1	
	PP.3	Delayed release	2017	I1, A3	Michlmayr et al., 2015

	PP.4	Knowledge loss	2017 - 2018	M1, M49	Rashid et al., 2017
User Management	PS.1	Lack of a multilingual GUI	2014 - 2019	I1, M52	Midha & Palvia, 2012
External Factors	PE.1	Industry dynamics	2017	I1	

4.1.1 Consortium Management

The consortium management category comprises problems concerned with the actions and processes which affect the stability of the project, namely *slow return on investment (ROI), turnover in service provider members, non-user-friendly website, lack of promotion, low number of users, low number of members,* and *lack of financial resources.*

PC.1. Slow return on investment (ROI) (2017). Collective development efforts for openMDM started in July 2014 (L2). In the beginning, member organizations split development responsibilities and set a timeline (June 2016) to integrate components (A2). Since the split development approach did not work, the release of the software was delayed and the parties, in particular service provider members, could not create a sustainable business until 2017 (I1). Since service providers were investing money (i.e., paying membership fees) and resources (e.g., developers' effort on code contribution), the *delayed release* led to *slow return on investment* for them and some to leave openMDM.

Interviewee 1 explained: *"[U]ntil 2017, all these service companies had started to leave openMDM. Because they couldn't justify internally why they should pay 20,000 to the EF in membership. Because there was no return on investment for them. So, they started to leave. And that was not good."*

PC.2. Turnover in service provider members (2015 - 2018). The consortium experienced turnover among service provider members mainly due to *slow return on investment* (A1, A2, A3, A5, I1). Although the missing resources were replaced with new members, the change in the service providers had a negative influence on the consortium's image (I1).


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Interviewee 1 explained the effect of turnover in the service provider members: *"It did not affect the actual MDM|BL project but it affected the public view of the openMDM. Losing members or losing participants is always a bad thing for us because it sheds a negative light on openMDM."*

PC.3. Non-user-friendly website (2014 - ongoing as of June 2019). openMDM's website contains a broad range of information from the history of the consortium to the governance structure, software releases, and events. (W4). However, the website does not provide any content about the benefits of using the software or of joining the consortium (I1).

PC.4. Lack of promotion (2016 - ongoing as of June 2019). The consortium does not actively promote the project. In the early phase (2015), consortium members were planning activities such as creating presentations about the community, presenting them in the EclipseCon and EuroForum events, and sharing them via online channels such as SlideShare (M3, M4). Although they followed these plans in 2015, they did not continue. Consortium members do not attend conferences as speakers *(low involvement in conferences)*.

Furthermore, the consortium did not create and share user stories which would serve as a sort of requirements document and attract organizations with similar needs (M3, I1) *(lack of user stories)*.

PC.5. Low number of users (2018 - ongoing as of June 2019). The number of users has two main effects on the openMDM. First, more use of the software means increased bug reports, which have a positive influence on the quality of software (I2). Second, users are seen as the potential driver members (I1). Since the project did not launch on time, there was no software to use until 2017 (*delayed release*). Furthermore, until April 2019 the user interface of the software was only in German (*lack of a multilingual GUI*). Together with the *lack of promotion* problem, user member numbers of openMDM did not increase until April 2019 (I1).

PC.6. Low number of driver members (2018 - ongoing as of June 2019). openMDM had five driver members as of 2019. Driver members influence the development direction, and provide the main financial resources for the continuity of software development (A3, I1). An increase in the driver member numbers is necessary for growth and sustainability (A1, I1).

Research Article ISSN: 1529-3181

PC.7. Lack of financial resources (2018 - ongoing as of June 2019). The collaboration relies on membership fees for sustainable development. The minimal financial requirement for a working model is €500k annually. However, the annual income is €206k, which is insufficient to cover expenses (A5). Lack of funds slows the development process (I1).

"Evaluation concludes that the working group has been running underfunded for the past years. This may be one of the reasons why progress is rather underwhelming." (A5)

4.1.2 Process Management

The process management category outlines the problems of the software development process. *Split development responsibility without a consortium-wide authority, integration problems, delayed release* and *knowledge loss* are the problems listed in this category.

PP.1. Split development responsibility without a consortium wide authority (2014 - 2016). Software development responsibility was allocated to the three driver members of the consortium: Audi, BMW, and Daimler (A2). Each of these members paid for, coordinated and monitored their part of the development with different service providers. The plan was to integrate the separately developed components in June 2016 (I1, A6). However, as there was no collaboration between the service providers and no central control mechanism or monitoring, the vendor behaviors were unobservable, and one vendor shirked responsibility. Consequently, the development process failed (I1, A6).

Interviewee 1 described the lack of accountability: *"Supplier 1 came back with this, Supplier 2 came back with that, Supplier 3 came back with nothing. Because they decided at that time, they didn't want to do anything. It was a wrong supplier, just couldn't do it."*

PP.2. Integration problems (2016 - 2018). Since the software components were developed independently without a consortium-wide authority, each provider used their own tools, repositories and frameworks (M35, M44, M45, M46). In the end, some of the components did not integrate well. As a result, integrating code took more time and effort than expected and it led to *delayed release* (I1).

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Interviewee 1 described the situation: *"In 2016, the organization came to a point where nothing worked, they had spent a ton of money, each of them. They were really frustrated with the results. They created a lot of disjunct pieces that just didn't build a solution."*

In the AC meeting on the date of 20th of January 2017 (M35), integration problems were discussed: *"The current structure of the openMDM web client together with the openMDM API is not well suited for the implementation of the search function [...]. In the API, every entity is loaded separately, leading to unnecessary overhead and poor performance."*

PP.3. Delayed release (2017). Due to the *split development responsibilities without a consortium wide authority* and *integration problems*, the software was not usable for more than three years, whereas the basic operational product was originally expected to launch by the end of 2017 (A3). *Delayed release* and *slow return on investment* weakened the support for the project inside of the member companies (I1).

PP.4. Knowledge loss (2017 - 2018). The base code of the openMDM was developed by the service provider members. Turnover in one of the service provider members caused problems with the pace of the development from July 2017 to August 2018. Due to the loss of experienced staff on the project, the code integration process was affected by knowledge loss, resulting in delays (M1, M49).

In SC meeting minutes of 22th of March 2018 the concerns about losing experienced developers was mentioned: *"[Name of the service provider] faces a lot of contract terminations and with that partially a loss of know-how. Stabilization of the situation is on target but the timeline [is] unknown. [Name of the service provider] stays committed to openMDM. All tasks will be executed as agreed. Mid-term future of [Name of the service provider] and OpenMDM depends on personnel and know-how availability and is still in planning."*

4.1.3 User Management

The user management category contains problems related to users' needs and expectations. The only problem in this category was the lack *of a multilingual GUI*.

Research Article ISSN: 1529-3181

PS.1. Lack of a multilingual GUI (2014 - 2019). Initially, the software only had a user interface in German, which was an obstacle for companies from other countries (I1, M52).

The following note highlights the issue: *"Tata asked for a quick internationalization of the GUI, since today's implementation displays a GUI in German language."* (M52)

4.1.4 External Factors

The external factors category contains the problems related to the outside of the consortium. Only one factor was found in this category, *industry dynamics*.

PE.1. Industry dynamics (2017 - ongoing as of June 2019). In 2017, the scandal involving falsified emissions tests led to a negative public image of the automotive industry, which made openMDM members unwilling to promote the project publicly (I1). Service providers are reluctant to be associated with automotive industry projects (I1).

Interviewee 1 talked about the impact of the emissions falsification scandal on the entire industry: *"You may have heard [that] the automobile industry has little problem in presenting [what they do] within terms of the diesel scandal and all these things. None of them wants to talk in public which is a huge problem for us. Because they would be the ones to provide us the user stories [...]. It is just an issue with industry. If it was another industry, IT or so, everybody would be happily willing and blurting out that we did something great. But here we have this issue, unfortunately."*

4.1.5 Causal Relationships of the Problems

openMDM consortium's problems have a causal relationship with each other. We present the relationship between these problems in Figure 5.

Split development responsibility without a consortium wide authority caused *integration problems*. Integration problems led to *delayed release*. This was a cause for both *slow ROI* and *low number of users.* Slow return on investment led to *turnover in service provider members* and *knowledge loss*. Meanwhile, the low number of users was one of the reasons for the *low number of driver members.* Low number of

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

driver members led to *lack of financial resources*, since the driver members are providing the financial support for the consortium. Knowledge loss and lack of financial resources led to *slow pace of development.*

Other reasons for low number of users and later low number of driver members were *lack of a multilingual GUI*, *non-user-friendly website*, and *lack of promotion,* which was also a result of *industry dynamics*.

The conclusion, *slow pace of development*, is not an ideal situation for the consortium, but due to the solutions (described in Section 4.3), development continues at a steady pace, albeit slower than originally envisioned. The current *slow pace of development* is expected and therefore does not contribute to a repetition of previous problems, *delayed release*.

Communications of the Association for Information Systems
Research Article ISSN: 1529-3181

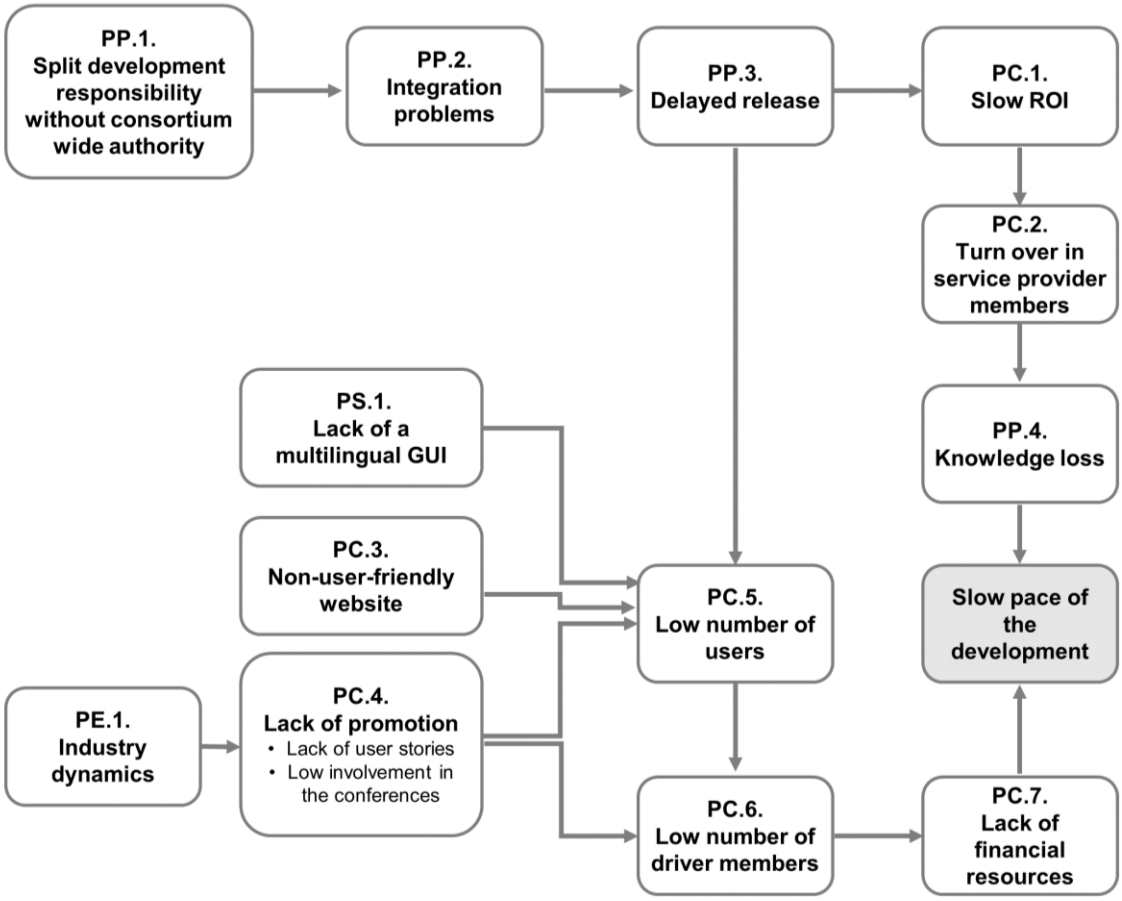

Figure 5. The Causal Relationships between Problems in openMDM

The diagram shows the relationship between distinct problems (white boxes) and the ultimate consequence (gray box). Arrows between boxes indicate a proposed causal relationship between problems and other problems, or between problems and the consequence.

4.2 Solutions and Success Factors of a User-Led Open Source Consortium

RQ2 is "What are solutions to the problems which occur in a user-led open source consortium, and which factors lead to success?" We addressed this question by examining openMDM consortium. We found that openMDM consortium is following a combination of best practices from the inter-company collaboration, OSS governance and OSS development practices. Besides following the best practices, the openMDM consortium developed some solutions which became success factors for the consortium.


Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

We found 27 success factors and grouped them into four categories: consortium management, process (software development) management, user management and external factors. These success factors are listed in Table 4. In some cases, we were able to map these factors to known success factors in the literature and present these references. Furthermore, we found support for many of these factors through triangulation with another consortium, namely Sakai. We did not conduct a complete analysis of Sakai, but a partial analysis in order to validate our findings on openMDM. As Sakai was not the primary focus of this research, additional information about Sakai can be found in the appendix (Yenişen Yavuz et al., 2022). We present the supporting evidence from Sakai in the same table.

Table 4. Solutions and Success Factors of openMDM

Category	Code	Solutions and success factor	openMDM data source	Sakai (Supporting data source)	Reference in literature
Consortium Management	SC.1	Clearly defined rules and boundaries	I1, L1	SB, SI1, SPP1	Ostrom, 1990; Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001
	SC.2	Collective prioritization	I1	SBP5, SBP10, SBP16, SI2, SPP1	Ostrom, 1990; Mattessich & Monsey, 1992; Rikkiev & Mäkinen, 2009; Ågerfalk & Fitzgerald, 2008
	SC.3	Openness and transparency	I1, I2, A1, M2, M4, M6, M14, M20, M21	SB, SBP5, SBP10, SI1, SPP1	Tsay et al., 2014; Riehle, 2015; Riembauer et al., 2020
	SC.4	Shared resources and equality	I1, I2, L1, L2, M2, M6, M9, M11	SB1, SPP1	Ostrom, 1990; Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Rikkiev & Mäkinen, 2009
	SC.5	Commitment of the members	I1, A2, A5	SB1, SBP9, SPP1	Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Rikkiev & Mäkinen, 2009


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

	SC.6	Inheriting established governance and legal structure	I2	SB	
	SC.7	Periodic communication	I2, A4, A5, M2, M12, M21	SBP1, SBP8, SI1, SPP1	Mattessich & Monsey, 1992; Bruce et al., 1995; Choudhury & Sabherwal, 2003
	SC.8	Events	I2	SI1, SPP1	Barcomb et al., 2018
	SC.9	Promoting the project (via events)	I1, M5, M28, M43	SB, SI1, SI2, SPP1	
Process Management	SP.1	Timebox development with milestone releases	I1, I2		Michlmayr et al., 2015; Choudhury & Sabherwal, 2003
	SP.2	A dedicated project manager and a persistent team of developers	I1, I2, A6, A7		Hoffmann & Schlosser, 2001; Chin et al., 2008; Sagers, 2004; Colazo & Fang, 2010; Rashid et al., 2017; Ågerfalk & Fitzgerald, 2008
	SP.3	Sanction mechanism	I1		Ostrom, 1990; Barthélemy & Quélin, 2006
	SP.4	Monitoring and regular assessment	I1, I2	SI1, SI2	Ostrom, 1990; Bruce et al., 1995; Hoffmann & Schlosser, 2001; Chin et al., 2008; Barthélemy & Quélin, 2006
	SP.5	Code review	I2		Rigby et al., 2008
	SP.6	Single repository	I1, I2, M31, M44, M47, M48, M49	SBP9	
	SP.7	High-quality code	I1, I2		Conley & Sproull, 2009; Ågerfalk & Fitzgerald, 2008

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

User Management	SU.1	Being responsive to the users (Quick responses to bug reports)	I2, M2, M63	SBP1, SBP5, SBP25, SI2	Singh, 2010
	SU.2	Multilingual graphical user interface (GUI)	I1, I2	SI2, SPP3	Midha & Palvia, 2012
	SU.3	Proper documentation	I1, I2	SPP1	Steinmacher et al., 2015; Lin et al., 2017; Barcomb et al., 2018
	SU.4	Customization (Product line approach)	I1	SBP17, SI1, SPP1, SWi1	van Wendel de Joode, 2004
External Factors	SE.1	Power of the driver members	I1		
	SE.2	Collaboration opportunities	I1	SB, SPP1	

4.2.1 Consortium Management

The consortium management category includes practices associated with collaboratively using a common resource pool, which is based on the Ostrom's managing the commons principles (Ostrom, 1990) as expressed by Interviewee 1, successful inter-company collaboration, and successful open source governance.

SC.1. Clearly defined goals, rules and boundaries. The goals and rules of the consortium, processes, responsibilities and privileges of the members, and governance structure are clearly defined in the charter (L1). In order to join the collaboration, organizations must accept these preset rules. Setting rules and boundaries at the beginning is necessary to avoid potential conflicts and is essential for the sustainability of the collaboration (I1).

Interviewee 1 explains the importance of having clear goals: *"[I]f you take openMDM, and another thing in the automotive industry and try to mix them, they don't have one mission anymore, they do not have one goal. [S]o, it needs to be clearly defined in this working group, we deal just with management of measured*

Research Article ISSN: 1529-3181

data, in the other working group we deal with evaluation of AI, and in the third working group, we deal with testing or simulation. If you mix it into one, it will fail."

SC.2. Collective prioritization. openMDM members have a common goal, and they are working collaboratively to reach this goal. Although driver members have different priorities, for the health of the collaboration they jointly determine the priorities. In order to work effectively and avoid conflicts, openMDM follows a well-defined decision process with a Steering Committee (SC) and an Architecture Committee (AC) (I1).

Interviewee 1 explained how they prioritize the requirements with the following words: *"It goes by collective choice arrangements. So, the collective choice arrangements helped us to say okay for now, we will do this, and later on we may do that. But for now, our focus is on building exactly this."*

SC.3. Transparency and openness. Transparency is vital for the health of the consortium (I1). SC and AC meetings are open to anyone to attend, and meeting minutes are mostly publicly available. Wiki pages of the community and mailing lists are used to share information with the public and project participants (A1, M6, M14, M20, M21). Issues, assignments, bug reports, and achievements are open and transparent to all members (I1, I2, M2, M4).

Interviewee 2 explained the importance of transparency with the following words: *"We have high-quality releases. They are documented. People know what's going on because we are announcing our releases via mailing lists. I think we have a good quality and even more. It is really about making open source, having this in the open and transparent."*

Since some of the driver members are competitors with each other, their activities should be in accord with the antitrust law in Germany. This increases the importance of transparency. Competitive members are only allowed to work together by sharing their common activities with the public (I1).

Interviewee 1 explained how openness benefits the consortium: *"As of today, if a person from Daimler and a person from Audi need to talk to each other, they cannot do it. Because they might break antitrust law*

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

[...]. When they come to the EF, they can talk. They can plan together because it is open, public, and transparent."

SC.4. Shared resources and equality. Resource sharing has a positive influence on the efficiency of the development process and on the quality of the code (I2). Driver members have equal rights by means of resource sharing and influence on the project decisions (L1, I1). Each of the driver members has a seat on the SC, and each of them has three voting rights (L2). The SC is responsible for governance of the strategic decisions. Acceptance of these decisions are decided by majority, mostly unanimously (e.g., M2, M6, M9, M11).

Interviewee 2 explained the importance of resource sharing with the following words: *"From my point of view, it's very useful to share resources, share know-how and do things together. And if you do it in the open and everybody can be involved, you're faster to market and you have a higher quality because you can share your resources, especially at the moment. Everybody knows it's not very easy to get good developers. So, if you share your resources, you have more outcomes*."

SC.5. Commitment of the members. Daimler positions the openMDM as part of their Industry 4.0 vision (A2). Daimler and BMW are investing in in-house projects based on openMDM (A5, I1). These activities show that the project is important for driver members (I1).

SC.6. Inheriting established governance and legal structure. The consortium is working with an umbrella open source foundation, the EF. This strategy allows the consortium to apply established open source best practices, which helps prevent potential problems. These practices are about governance structure, bylaws, intellectual property (IP) management policy, and development tools (I2).

Interviewee 2 explained this advantage with the following words: *"So, if you share your resources, you have more outcomes. And furthermore, if you do it open source, do it like in a working group like the EF. You have the governance model and the bylaws for example, having antitrust rules that don't put you in danger of having accusations because of trust issues, which we know is very important for our German OEMs."*

Research Article ISSN: 1529-3181

SC.7. Periodic communication. Committee members of openMDM have regular meetings (A5, M2). AC has three weekly conference calls, and since 2017 the toolkit management team has conducted weekly meetings (M2, A4, M12, I2). Members discuss specifications, technology decisions, job assignments, and project status. For transparency and observability, meeting minutes and assignments are shared on the community's wiki and the openMDM consortium mailing list (M2, M21). Periodic communication has an influence on development of *trust and understanding* between team members and developers which has an influence on the *improved code development process.*

SC.8. Events. In addition to the periodic meetings, all community members gather in the annual meetings, which take place once a year. openMDM also hosts hackathons and developer workshops, which create an opportunity for developers to build trust, exchange experiences, present best practices, discuss requirements and solutions of the projects (I2, M5, M28, M43). Events help to *improve the trust and understanding* between team members and developers, which has a positive influence on the *code development process*.

SC.9. Promoting the project. Attending conferences as speakers and explaining the project to the audience in related industries is helpful to gain more attention and users (I1).

4.2.2 Process Management

The process management category covers success factors related to code development strategy and coordination.

SP.1. Timebox development with regular milestone releases. The toolkit management team consists of developers and a toolkit manager, and follows the timebox development approach, which enables self control mechanisms. The team uses six-week timeboxes culminating in milestones. At the end of each timebox, an updated version of the software is being released (e.g., in A7). The toolkit manager presents the results of the timeboxes, status of the development team, and the next steps to the SC periodically (e.g., in A6, A7). This approach enables the SC to monitor and measure the development process (I1, I2).

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

This solution addresses the *split development*, and *delayed release* problem and resulted in *improved code development*.

Interviewee 2 explained the process and importance of following this approach: *"[R]eleasing every six to eight weeks, knowing exactly what features are going on what branch document, documenting this via Git commits and writing, for example, bug numbers into the Git commits, these give us a great overview what we are doing and we can always prove at every moment, what we are doing on what branch and what is already in our code base, what is released and what is still on development. So, we have a very good overview about what we are doing and what's going on."*

SP.2. Having a dedicated project manager (PM) and a persistent team of developers. After the failure of splitting development responsibilities, openMDM changed its development strategy. In 2017, the consortium created a shared pool of resources and assigned a project manager (toolkit manager) as the head of the software development team (I1). Four developers from three companies were involved in this development team in 2017 and 2018 (A6, A7). Developers are either employees of the service providers or of the vendors working for the driver members. Developers' contracts are for the duration of a timebox, which lasts six weeks and is extended after an evaluation of development efforts (I1, I2). The whole development process is controlled by the toolkit manager who reports to the SC. This approach helps to perform *monitoring and regular assessment* and is a solution to the *split development effort without a consortium-wide authority* problem.

SP.3. Sanction mechanism. In openMDM, developers' contracts are for the duration of a timebox. The decision of whether or not to prolong a contract is made by an evaluation of the work performed during the timebox period (I1). This solution is a response to the *split development effort without a consortium-wide authority*.

Interviewee 1 explained how the sanction mechanism can address shirking: *"You have to have a sanction mechanism. If you don't develop what you have promised, you are out, you get fired."*

SP.4. Monitoring and regular assessment. *Monitoring and regular assessment* provide members with the opportunity to plan their processes and prioritize requirements (I1, I2). Having *a dedicated project*

Research Article ISSN: 1529-3181

manager and following the *timebox development* approach improved monitoring of the development process for the openMDM (I1, I2).

Interviewee 1 explained the importance of regular assessment: *[S]o, we said okay, these are the important things that we want to focus on in the first time-box. And then in the second time-box, we could do two more things. By this, you could measure every milestone, whether we have reached what we expected to reach. So, the development became a lot more predictable, and all the different organizations had a very good and exact understanding."*

SP.5. Single repository. Since 2016, openMDM members have been using a single repository (Eclipse Git) (M31). This means developers are working on the same repository, instead of using different repositories and combining code later. This approach improved *collaboration between teams,* enabled code versioning, and provided *better monitoring of the development process* (I1, I2, M31). An exception to this approach is the code developed by external contributors. These contributions are integrated into the Eclipse Git after a quality check (M44, M47, M48, M49).

SP.6. Code review. Code is not committed to the main codebase until the code is reviewed by another team member, which *increases the quality of the code* (I2).

SP.7. High-quality code. An improved code development process together with *code reviews*, using a *single repository*, *proper documentation* and quick response to bug reports (*being responsive to users*) led to an *increase in the quality of code* which resulted in *high-quality code*. With increased quality and a modular structure, the driver members benefit from the code through the software product line approach which allows *customization*. Having *high-quality code* increases the popularity of the software. Some data management systems suppliers are considering replacing their own code with the openMDM interface (I1). The consortium anticipates that this solution will address the *low number of users* and *low number of driver members* problems.

Interviewee 1 explained the expected effect of having high-quality code: *"We are also seeing companies that are building products for data management systems like AVL in Austria and FES in the Netherlands.*

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

We reached a point with the quality of our code where they are thinking about replacing their own code with this MDM interface."

4.2.3 User Management

This category focuses on the factors which help to ease the users' use of the software and increase the demand for the software. These are *quick responses to the bug reports*, *considering users' needs*, *proper documentation*, and *customization*.

SU.1. Being responsive to users. Being responsive to users shows that the project is alive (I2). It demonstrates that reported bugs are seen, and will be handled (I2). In openMDM, the toolkit manager responds quickly to bug reports whether they are reported by members or non-member users (I2).

SU.2. Multilingual graphical user interface (GUI). The openMDM software did not have a GUI in English until 2019. In April 2019, the development team published the English user interface (I1). This solution addresses *the lack of a multilingual GUI* problem and is expected to increase the user numbers as a response to the *low number of users* problem.

SU.3. Proper documentation. The AC and the toolkit manager prepare documents for different audiences. These documents are guidelines, specifications, release notes, and process plans which have the purpose of providing information about technical aspects and showing how to avoid repeating problems (I1, I2). This solution addresses *knowledge loss* and *low number of users*.

Interviewee 2 explained the importance of documentation: *"When I started, there was not much technical documentation. I set up all the technical documentation and I am still maintaining it. In each milestone we do not only do the code updates or publish new code, but we also publish updates in our documentation."*

SU.4. Customization. Since 2018, openMDM has been following a product line development approach. This approach allows members to use openMDM software as a core and build tailor-made components on top of it for their specific requirements. Allowing *customization* increased the satisfaction of the members and is expected to address *low number of users* and *low number of driver members* problems of the

Research Article ISSN: 1529-3181

consortium (I1). Meanwhile, increasing modularity means increasing the complexity of the code and slows down the development pace of the process (I2).

Interviewee 1 explained how the members of the consortium benefit from the product line: *"Let's say this is Daimler and this here is BMW. They are all using the platform and they are building other things in parallel. Other functions that they need."*

4.2.4 External Factors

External factors refer to the aspects which are not in control of the members. In the openMDM, these factors are the *power of the driver members* and *collaboration opportunities* in the industry.

SE.1. Power of the driver members. Driver members of the openMDM consortium (e.g. Audi, Daimler, BMW) have power in the automobile industry to set standards. After the software reaches maturity, the driver members aim to use it as part of their core technology for measuring data, and they will look for compatible products with openMDM standards (I1). It is expected that this power will help to overcome *the low number of users* problem and become a success factor for the consortium.

Interviewee1 explained how key industry players' power affects the project: *"This number will go up for a reason within the organizations like Daimler and BMW; now they will not buy anything that is not based on openMDM5. Any system."*

SE.2. Collaboration opportunities. The automobile industry has a significant role in Germany. A number of associations and institutions support the industry, such as VDA (Verband der Automobilindustrie) and Fraunhofer Institute. There are a number of meetings and organizations that take place every year around the automobile industry. These events make it easy for company members to meet, get acquainted and collaborate (I1).

4.2.5 Relationships between Solutions and Success Factors

Finding solutions to specific problems and using best practices which were already established in inter-company collaboration and OSS governance helped the openMDM consortium to achieve a sustainable

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

development process and success among members. These factors are related to one another. Some results of the success factors have already been established and some are expected to be seen in the longer term. We present the relationship between these factors in Figure 6.

Clearly defined rules and boundaries, *collective prioritization*, *openness and transparency*, *shared resources and equality*, *commitment of the members* and *inheriting already established open source governance and legal structure* lead to build a successful collaboration from the beginning. *Increased trust and understanding* between the members which could be reached through *periodic communication* and *events* is another factor for the sustainability and success of the collaboration.

Following a *timebox development* approach, having *a dedicated project manager* and *a persistent team of developers*, having *an increased trust and understanding between team-members*, having a *sanction mechanism*, and *monitoring and regular assessments* together lead to *an improved code development process*.

Following open source development best practices such as *using a single repository*, conducting *code reviews,* performing *proper documentation* together with an improved code development process and being responsive to users results in having high-quality *code.*

High-quality code leads to a *high-quality product*. Having *high-quality code* allows *customization* and provides opportunity for members to use this code in parallel on their in-house products and create value added services.

On the other hand, *being responsive to users*, having a *multilingual GUI*, using *collaboration opportunities* and *promoting the project* are projected factors to increase the user and member base of the consortium and *increase the use of the product*.

Power of driver members is a factor which is projected to be a reason for *demand for compatible products* in the market.

Communications of the Association for Information Systems
Research Article ISSN: 1529-3181

Continuing the *successful collaboration* together with having *a high-quality product*, *increased use of the product* in the market, *demand for compatible products* are expected to lead to the end result of *sustainability of the product* in the market.

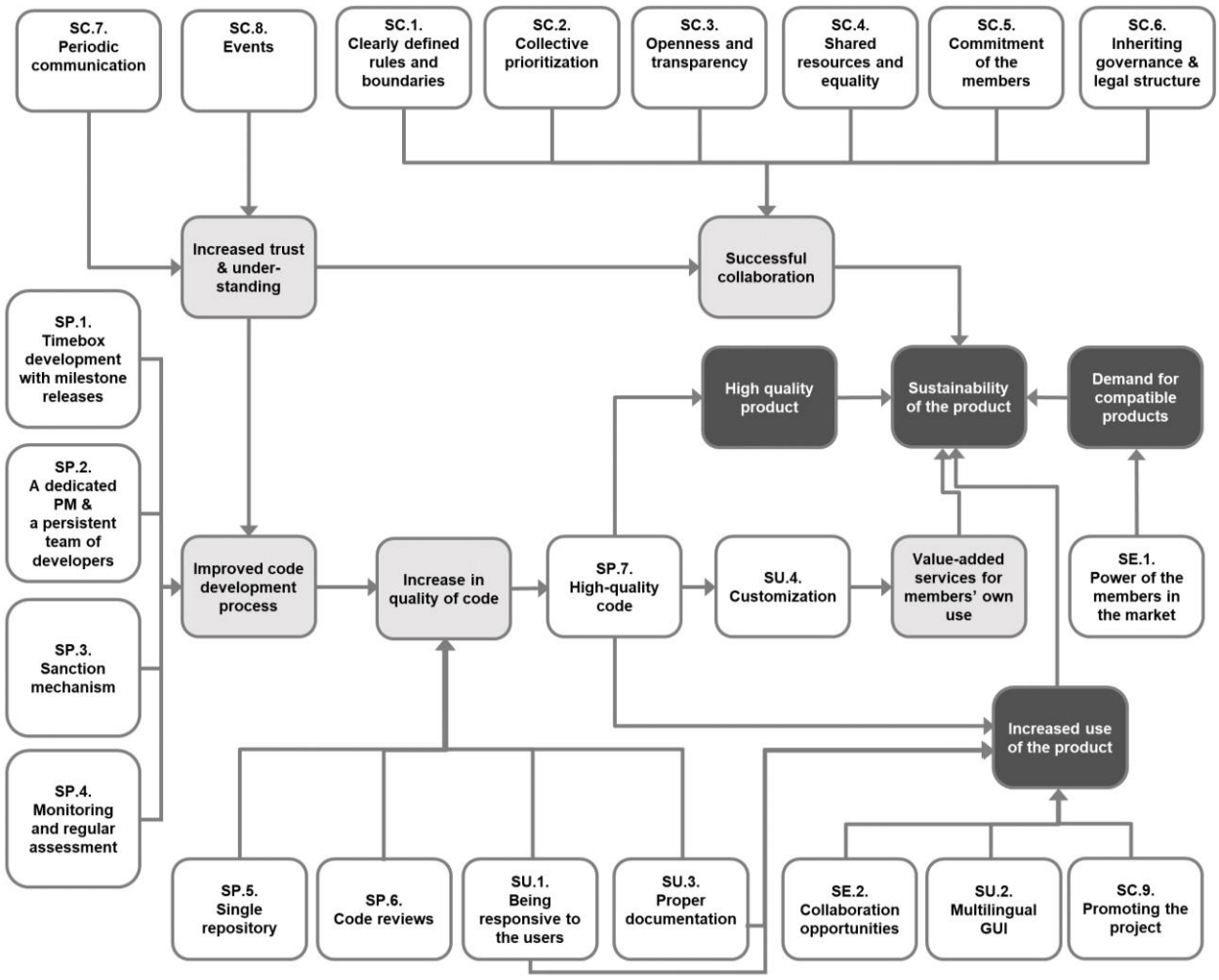

Figure 6. The Relationships between Success Factors in openMDM

Diagram shows the relationship between distinct solutions (white boxes), short-term consequences already observed in the openMDM project (light gray boxes), and expected long-term consequences which have not yet been realized (dark gray boxes). Arrows between boxes indicate a proposed causal relationship between solutions and consequences, and between consequences and other consequences.


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Some solutions, such as *high-quality code*, were enabled by earlier solutions. In such cases, the solution is shown as following from the consequence of the earlier solutions, and can be viewed as both a consequence and a solution.

5 Discussion

We found that many of the problems and solutions presented in this study are discussed in the OSS, inter-company collaboration, and outsourcing literature, suggesting that the user-led consortium phenomenon is best understood by considering all three types of literature.

One of the biggest problems openMDM faced was related to financial resources. Since the financial resources were insufficient for development efforts, the pace of development slowed down. Furthermore, *lack of financial resources* was and remains an obstacle for effective code development. Developing the software took more time than initially projected. This meant that members of the project did not receive an ROI as anticipated. In some cases, members left the collaboration due to the *slow ROI*.

Since user-led OSS consortia depend on financial support from members, when a consortium needs to increase its financial resources, it needs to increase its member base. Increasing the member base provides additional benefits to OSS development: the literature demonstrates that having more users has positive effects on the development in terms of improved software testing and bug finding (Midha et al., 2012). If a project is not explained and promoted enough, its potential members will not be aware of the project and will not consider joining the collaboration. Midha et al. (2012) suggest that having a software interface in different languages increases the attractiveness of the product and potential for market success. Our research shows that having the user interface only in one language, in German in our case, was an obstacle in reaching more users and members. In the Sakai data, we also found evidence that having a multi-language user interface increases the number of users of the software and increases its recognition and acceptance. Further data from Sakai shows that promoting the product by attending conferences, speaking about the product and organizing events around this specific software increases the awareness about the product and consortium. As a result, we suggest that promoting the project and

Research Article ISSN: 1529-3181

using a multi-language GUI are two important factors for increasing user and member numbers of a user-led OSS consortium.

Another problem was the lack of coordination between stakeholders. Our analysis shows that splitting the code development responsibility between different parties without a consortium wide authority, and assigning partial responsibility led to coordination and integration problems. These findings support our expectation that user-led OSS consortia may face problems due to being inexperienced with OSS development methods. In OSS, it is unusual for large components to be developed in private and released after completion, and this is a known risk when it comes to realizing the benefits of OSS (Pinto et al., 2018). Rather, software is developed publicly, enabling discussion and allowing for ongoing adjustments to support integration (Ågerfalk & Fitzgerald, 2008). openMDM applied formal and informal controls from outsourcing (Choudhury & Sabherwal, 2003; Barthélemy & Quélin, 2006) to the problem by following *time-box development with milestone releases*, *periodic communication*, *monitoring* and *regular assessment* of the development process, and having *a sanction mechanism*. *Periodic communication*, *monitoring and regular assessment* are also seen as a success factor in Sakai.

A further risk of the development and implementation process is the *turnover in the developers*. When the core developers leave the project, it leads to experience loss, *know-how loss*, and time loss for the project, which is consistent with what is known about OSS (Rashid et al., 2017). Having *a dedicated project manager*, having a *persistent team of developers*, and *documentation* are solutions for this problem. *Increased understanding and trust* among developers, which can be built on *periodic communication* and *events*, have positive effects on effective code development, as well. In the OSS literature, communication is described primarily as open and asynchronous (Fogel, 2005; Tsay et al., 2014; Riehle, 2015), and these aspects were also observed in the *periodic communication* of openMDM. Effective code development increases the possibility of having regular milestone releases and of having *high-quality code*. Performing *code review*, working on a *single repository*, *proper documentation*, and *being responsive to the users* are additional factors which improve the quality of the code. The latter two

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

also influence the usability of the software. In particular, being responsive to the users was also observed in Sakai.

Our research shows that usability and customizability are important factors in the value of open source user-led consortia software. *Customization* enables members to create in-house systems based on the consortium's core software, which increases its utility to them. Having a standard core enables them to use their market power to set standards relying on the software, increasing the sustainability of the project. In both openMDM and Sakai, this approach is effective.

Our findings agree with previous findings in the inter-company collaboration research that *equality* of the members, *openness and transparency*, *setting boundaries* at the beginning of the process, having *collective responsibility*, and *commitment of the members* leads to successful collaboration (e.g. Mattessich & Monsey, 1992; Bruce et al., 1995; Rai et al., 1996; Hoffmann & Schlosser, 2001; Rikkiev & Mäkinen, 2009). Many of the problems observed in the OSS literature were not relevant to openMDM, but *knowledge loss* (Rashid et al., 2017), the effect of *code quality* on the number of users (Conley & Sproull, 2009), and the necessity of *effective coordination* (Sagers, 2004) were important factors in openMDM.

We performed this research by investigating a case from the automotive industry. In order to develop a fuller understanding of the problems which are specific to user-led OSS consortia, replication studies of other user-led OSS consortia in other industries should be conducted. A longitudinal study would also help establish the effectiveness of the recently applied solutions. Finally, there is significant room for further investigation into the ecosystem of user-led open source consortia, governance models, and process management.

6 Limitations

We followed an exploratory single-case case-study approach. A major limitation of this study is that the results are based on one user-led open source consortium. Since we followed a qualitative research method, we adopted Guba's (1981) trustworthiness criteria, namely credibility, transferability, dependability, and confirmability, to evaluate our research.

Research Article ISSN: 1529-3181

Credibility concerns the truth of the research findings. We used two methods to improve credibility: prolonged engagement and data triangulation. The investigation process lasted eight months. During this period, we evaluated all meeting minutes and published documents from July 2014 to April 2019. After that, we conducted interviews with the Eclipse Foundation representative and openMDM toolkit manager. This allowed us to triangulate data from a total of 86 sources. In July 2019, we presented our findings in the annual meeting of openMDM and received positive feedback.

Transferability is about establishing context-relevant statements. The subject of this research, the openMDM consortium, is an example from the automotive industry. By relating the case to the existing literature of collaboration and OSS success, we were able to identify common success factors and highlight those which appeared in the openMDM case. Furthermore, we used data from another consortium, Sakai, to demonstrate the transferability of our findings. 16 out of 22 solutions show similarities in both cases, but it remains for future work to determine if the openMDM observations apply to other user-led OSS consortia generally, or are unique to this particular case.

Dependability refers to having reliable and traceable research findings. Transparency is important for the openMDM consortium. They publish meeting minutes and decision documents online. With the exception of interviews, the data used in this research are publicly available in (Yenişen Yavuz et al., 2022). This facilitates the traceability of our findings.

Confirmability concerns objectivity. Member checking is one of the most effective ways of establishing that the analysis reflects the reality of the participants. We shared our research findings with the consortium members via email. In addition, we presented preliminary research results in the 2019 openMDM annual meeting to a positive reception.

7 Conclusion

Although the user-led OSS consortium phenomenon is not new, there are not many studies about this phenomenon. Existing studies are mostly in the educational sphere and investigate this phenomenon from the OSS perspective. Since the dynamics of the education industry are different from other industries, and


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

user-led OSS cannot be understood only by considering OSS literature, we identified this as a gap. We conducted an exploratory single-case case study, focusing on a user-led OSS consortium from the automotive industry, openMDM.

RQ1 was "What problems occur in a user-led open source consortium?" We examined this question by focusing on the problems experienced by openMDM over a 5-year period, from 2014 to 2019. We organized our results into four categories: consortium management, process management, user management, and external factors. In total, we found 13 problems. We found that the most important problems were seen during the development of the core code, which is related to process management. *Split development responsibility without a consortium-wide authority* is an obstacle to developing working code which led to *delayed release* and financial instability. These factors resulted in a *slow pace of development*. Furthermore, our research shows that using different frameworks and code repositories causes *integration problems* which affects the development process.

RQ2 was "What are solutions to the problems which occur in a user-led open source consortium, and which factors lead to success?" We applied the same categories used for the problems to find related solutions. According to our research, *working on timeboxes with milestone releases, having a dedicated project manager and a persistent team of developers,* and *monitoring and regular assessment* lead to having *high-quality code and software*. When the core code of the software offers the *customization* opportunity for the members, it increases utility. *The power of the members* increases the potential for *usability of the software* and *setting industry standards*.

Although this study focuses on a single case, it offers practitioners an understanding of problems which can arise in a user-led open source consortium, and how these might be addressed. Furthermore, this research demonstrates the advantage of viewing the user-led open source consortia as an example of both OSS development and inter-company collaboration, as neither category of literature was fully able to explain all the problems and solutions observed. The OSS literature explained eight of the 13 problems and 13 of the 22 solutions, the inter-company collaboration literature explained one of the problems and eight of the solutions, and the outsourcing literature explained one of the problems and four of the


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

solutions. Additionally, lack of familiarity with OSS development led to an initial closed development process (*split development responsibility without a consortium-wide authority*) which resulted in *integration problems* and a *delayed release*. The number of problems arising from OSS development suggests that founders and vendors seeking to establish a successful user-led open source consortium should acquire expertise in OSS development processes.

User-led OSS consortia are a distinct phenomenon which share elements of inter-company collaboration, outsourcing software development, and vendor-led OSS development and cannot be understood by using only a single lens. In common with inter-company collaborations, user-led OSS consortia experience coordination problems with multiple stakeholders. We also observe shirking and lack of behavior observability, which are known from the outsourcing literature. We find that motivations for pursuing the creation of the consortia, such as cost-sharing, interoperability and the creation of a de-facto standard are familiar from vendor-led OSS foundations. Finally, user-led OSS consortia experience problems related to a lack of knowledge of OSS development, which can exist in both the members and the vendors, leading to inefficient processes.

Our research has practical implications for companies wanting to engage in successful user-led OSS consortia, and also contributes to the academic understanding of the phenomenon outside of higher education.

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Acknowledgements

We would like to thank our interviewees for their time and sharing their experience about the openMDM project. We also would like to thank Maximilian Capraro, Nikolay Harutyunyan, Andreas Kaufmann, Julia Krause, and Sebastian Schmid for their valuable feedback on this paper. Finally, we would like to thank the reviewers for their extensive suggestions on how to improve this article.

References

Ågerfalk, P. J., & Fitzgerald, B. (2008). Outsourcing to an unknown workforce: Exploring opensourcing as a global sourcing strategy. *MIS Quarterly*, *32(2),* 385-409. DOI:10.2307/25148845

Almeida, F., Oliveira, J., & Cruz, J. (2011). Open standards and open source: enabling interoperability. *International Journal of Software Engineering & Applications*, *2*(1), 1-11. DOI: 10.5121/ijsea.2011.2101

Aron, R., Clemons, E. K., & Reddi, S. (2005). Just right outsourcing: Understanding and managing risk. *Journal of management information systems*, *22*(2), 37-55.

Bacon, J. (2012). *The art of community: Building the new age of participation*. O'Reilly Media, Inc.

Baldwin, C., & Von Hippel, E. (2011). Modeling a paradigm shift: From producer innovation to user and open collaborative innovation. *Organization Science, 22(6)*, 1399-1417. DOI: 10.2139/ssrn.1502864

Barcomb, A., Kaufmann, A., Riehle, D., Stol, K. J., & Fitzgerald, B. (2018). Uncovering the periphery: A qualitative survey of episodic volunteering in free/libre and open source software communities. *IEEE Transactions on Software Engineering*. DOI:10.1109/tse.2018.2872713

Barcomb, A., Stol, K. J., Riehle, D., & Fitzgerald, B. (2019). Why do episodic volunteers stay in FLOSS communities? In *2019 IEEE/ACM 41st ICSE*. DOI: 10.1109/ICSE.2019.00100


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

Barcomb, A., Stol, K. J., Fitzgerald, B., & Riehle, D. (2020). Managing Episodic Volunteers in Free/Libre/Open Source Software Communities. *IEEE Transactions on Software Engineering*. DOI: 10.1109/TSE.2020.2985093

Barthélemy, J., & Quélin, B. V. (2006). Complexity of outsourcing contracts and ex post transaction costs: an empirical investigation. *Journal of Management Studies*, *43*(8), 1775-1797.

Benbasat, I., Goldstein, D. K., & Mead, M. (1987). The case research strategy in studies of information systems. *MIS Quarterly*, 369-386. DOI:10.2307/248684

Berdou, E. (2006). Insiders and outsiders: paid contributors and the dynamics of cooperation in community led F/OS projects. In *IFIP International Conference on Open Source Systems* (pp. 201-208). Springer.

Boldyreff, C., Nutter, D., & Rank, S. (2004). Communication and Conflict Issues in Collaborative Software Research Projects. *26th International Conference on Software Engineering - W8S Workshop "Collaboration, Conflict and Control: The 4th Workshop on Open Source Software Engineering"* DOI:10.1049/ic:20040258

Bosu, A., & Sultana, K. Z. (2019). Diversity and Inclusion in Open Source Software (OSS) Projects: Where Do We Stand?. In *2019 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*. DOI:10.1109/ESEM.2019.8870179

Bruce, M., Leverick, F., Littler, D., & Wilson, D. (1995). Success factors for collaborative product development: a study of suppliers of information and communication technology. *R&D Management, 25(1)*, pp. 33-44. DOI:10.1111/j.1467-9310.1995.tb00898.x

Carillo, K. D. A., & Marsan, J. (2016). “The Dose Makes the Poison”-Exploring the Toxicity Phenomenon in Online Communities. In *International Conference on Information Systems.*

Castilla, E. J., & Benard, S. (2010). The paradox of meritocracy in organizations. *Administrative Science Quarterly, 55(4)*, 543-676. DOI:10.2189/asqu.2010.55.4.543


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Ciesielska, M., & Westenholz, A. (2016). Dilemmas within commercial involvement in open source software. *Journal of Organizational Change Management*.

Chengalur-Smith, I., Sidorova, A., & Daniel, S. L. (2010). Sustainability of free/libre open source projects: a longitudinal study. *Journal of the Association for Information Systems*, *11*(11), 5. DOI:10.17705/1jais.00244

Chin, K. S., Chan, B. L., & Lam, P. K. (2008). Identifying and prioritizing critical success factors for coopetition strategy. *Industrial Management & Data Systems.* DOI:10.1108/02635570810868326

Choudhury, V., & Sabherwal, R. (2003). Portfolios of control in outsourced software development projects. *Information systems research*, *14*(3), 291-314.

Colazo, J. A., & Fang, Y. (2010). Following the sun: Temporal dispersion and performance in open source software project teams. *Journal of the Association for Information Systems*, *11*(11), 4. DOI:10.17705/1jais.00243

Conley, C. A., & Sproull, L. (2009). Easier said than done: An empirical investigation of software design and quality in open source software development. In *2009 42nd Hawaii International Conference on System Sciences*. IEEE. DOI:10.1109/HICSS.2009.687

Courant, P. N., & Griffiths, R. J. (2006). Software and collaboration in higher education: A study of open source software. *New York: Ithaca. Retrieved January*, *30*, 2009.

Crowston, K., Annabi, H., & Howison, J. (2003). Defining open source software project success. In Proceedings of the International Conference on Information Systems. DOI:10.1287/mnsc.1060.0550

Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday*, *10*(2). DOI:10.5210/fm.v10i2.1207

Dacin, M. T., Hitt, M. A., & Levitas, E. (1997). Selecting partners for successful international alliances: Examination of US and Korean firms. *Journal of World Business, 32(1)*, 3-16. DOI:10.1016/S1090-9516(97)90022-5


Communications of the
Association for Information Systems

Research Article ISSN: 1529-3181

Dahlander, L., & Magnusson, M. G. (2005). Relationships between open source software companies and communities: Observations from Nordic firms. *Research Policy*, *34*(4), 481-493. DOI:10.1016/j.respol.2005.02.003

Dahlander, L. (2007). Penguin in a New Suit: A Tale of How De Novo Entrants Emerged to Harness Free and Open Source Software Communities. *Industrial and Corporate Change*, *16*(5), 913-943.

Damian, D. (2003). Global software development: growing opportunities, ongoing challenges. *Software Process: Improvement and Practice, 8(4),* 179-182. DOI:10.1002/spip.182

Daniel, S., Agarwal, R., & Stewart, K. J. (2013). The effects of diversity in global, distributed collectives: A study of open source project success. *Information Systems Research, 24(2)*, 312- 333. DOI:10.1287/isre.1120.0435

Ducheneaut, N. (2005). Socialization in an open source software community: A socio-technical analysis. *Computer Supported Cooperative Work (CSCW)*, *14*(4), 323-368. DOI:10.1007/s10606-005-9000-1

Eisenhardt, K. M. (1989). Building theories from case study research. *Academy of Management Review, 14(4)*, 532-550. DOI:10.5465/amr.1989.4308385

Ehls, D. (2017). Open source project collapse–sources and patterns of failure. In *2017 Proceedings of the 50th Hawaii International Conference on System Sciences.*

Feller, J., & Fitzgerald, B. (2000). A framework analysis of the open source software development paradigm. In *ICIS 2000 Proceedings of the 21st International Conference on Information Systems*.

Filippova, A., & Cho, H. (2015). Mudslinging and manners: Unpacking conflict in free and open source software. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing* (pp. 1393-1403).

Fogel, K. (2005). *Producing Open Source Software: How to Run a Successful Free Software Project.* O'Reilly Media, Inc.

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Fortuin, F. T., & Omta, S. W. F. (2008). The dark side of open innovation: a survey of failed inter-company cooperation. *In Proceedings of the 8th International Conference on Management in AgriFood Chains and Networks.*

Foucault, M., Palyart, M., Blanc, X., Murphy, G. C., & Falleri, J. R. (2015, August). Impact of developer turnover on quality in open-source software. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering* (pp. 829-841).

Geiger, R. S., Howard, D., & Irani, L. (2021). The Labor of Maintaining and Scaling Free and Open-Source Software Projects. *Proceedings of the ACM on Human-Computer Interaction*, *5*(CSCW1), 1-28.Germonprez, M., Allen, J. P., Warner, B., Hill, J., & McClements, G. (2013). Open source communities of competitors. *Interactions, 20(6)*, 54-59. DOI:10.1145/2527191

Guba, E. G. (1981). Criteria for assessing the trustworthiness of naturalistic inquiries. *Educational Communication and Technology, 29*(2), pp. 75-91. DOI:10.1007/BF02766777

Guizani, M., Chatterjee, A., Trinkenreich, B., May, M. E., Noa-Guevara, G. J., Russell, L. J., Cuevas Zambrano, G.G., Izquierdo-Cortazar, D., Steinmacher, I., Gerosa, M.A. & Sarma, A. (2021). The Long Road Ahead: Ongoing Challenges in Contributing to Large OSS Organizations and What to Do. *Proceedings of the ACM on Human-Computer Interaction*, *5*(CSCW2), 1-30.

Haefliger, S., Von Krogh, G., & Spaeth, S. (2008). Code reuse in open source software. *Management Science*, *54*(1), 180-193. DOI:10.1287/mnsc.1070.0748

Harutyunyan, N., Riehle, D., & Sathya, G. (2020, January). Industry best practices for corporate open sourcing. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*.

Henkel, J., 2004. Open source software from commercial firms–tools, complements, and collective invention. *Zeitschrift für Betriebswirtschaft*, 4, pp.1-23.

Herbsleb, J. D., & Grinter, R. E. (1999). Splitting the organization and integrating the code: Conway's law revisited. In *Proceedings of the 21st International Conference on Software Engineering*. DOI:10.1145/302405.302455

Communications of the Association for Information Systems

Research Article ISSN: 1529-3181

Hoffmann, W. H., & Schlosser, R. (2001). Success factors of strategic alliances in small and medium-sized enterprises—An empirical survey. *Long Range Planning*, *34(3),* 357-381. https://doi.org/10.1016/S0024-6301(01)00041-3

Howison, J., & Crowston, K. (2014). Collaboration through open superposition: A theory of the open source way. *MIS Quarterly*, *38*(1), 29-50.

Kelly, M. J., Schaan, J. L., & Joncas, H. (2002). Managing alliance relationships: Key challenges in the early stages of collaboration. *R&D Management, 32(1)*, 11-22. DOI:10.1111/1467-9310.00235

Kochhar, P. S., Kalliamvakou, E., Nagappan, N., Zimmermann, T., & Bird, C. (2019). Moving from Closed to Open Source: Observations from Six Transitioned Projects to GitHub. *IEEE Transactions on Software Engineering*. DOI:10.1109/tse.2019.2937025

Lee, G. K., & Cole, R. E. (2003). From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development. *Organization Science*, *14*(6), 633-649. DOI:10.1287/orsc.14.6.633.24866

Levy, M., & Germonprez, R. M. (2015). Is it egalitarianism or enterprise strategy? Exploring a new method of innovation in open source. *In 21st AMCIS 2015*.

Lin, B., Robles, G., & Serebrenik, A. (2017). Developer turnover in global, industrial open source projects: Insights from applying survival analysis. In *2017 IEEE 12th International Conference on Global Software Engineering*. DOI:10.1109/ICGSE.2017.11

Liu, M., Hansen, S., & Tu, Q. (2014). The community source approach to software development and the Kuali experience. *Communications of the ACM, 57(5)*, 88-96. DOI:10.1145/2593687

Liu, M., Hansen, S., & Tu, Q. (2020). Keeping the family together: Sustainability and modularity in community source development. *Information and Organization*, *30*(1), 100274.

Liu, M., Wang, H., & Zhao, L. (2007). Achieving flexibility via service-centric community source: The case of Kuali. *AMCIS 2007 Proceedings*.

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Liu, M., Wang, H. J., & Zhao, J. L. (2012). Technology flexibility as enabler of robust application development in community source: The case of Kuali and Sakai. *Journal of Systems and Software, 85(12)*, 2921-2928. DOI:10.1016/j.jss.2012.06.026

Liu, M., Wu, X., Zhao, J. L., & Zhu, L. (2010). Outsourcing of community source: identifying motivations and benefits. *Journal of Global Information Management, 18(4)*, 36-52. DOI:10.4018/jgim.2010100103

Liu, M., Zeng, D. D., & Zhao, J. L. (2008). A Cooporative Analysis Framework for Investment Decisions in Community Source Partnerships. *AMCIS 2008 Proceedings*.

Lumbard, K., Wethor, G., Goggins, S., Buhman, A., Hale, M., & Germonprez, M. (2020). Welcome? Investigating the reception of new contributors to organizational-communal open source software projects. In *26th Americas Conference on Information Systems, AMCIS 2020*. Association for Information Systems.

Maruping, L. M., Daniel, S. L., & Cataldo, M. (2019). Developer centrality and the impact of value congruence and incongruence on commitment and code contribution activity in open source software communities. *MIS Quarterly*, *43*(3), 951-976.

Mattessich, P. W., & Monsey, B. R. (1992). *Collaboration: What Makes It Work. A review of research literature on factors influencing successful collaboration.* Amherst H. Wilder Foundation.

Mauerer, W., & Jaeger, M. C. (2013). Open Source Engineering Processes/Open Source-Entwicklungsprozesse. *IT-Information Technology*, *55*(5), 196-203. DOI:10.1524/itit.2013.1008

Medappa, P. K., & Srivastava, S. C. (2019). Does superposition influence the success of FLOSS projects? An examination of open-source software development by organizations and individuals. *Information Systems Research*, *30*(3), 764-786.

Michlmayr, M. (2004). Managing Volunteer Activity in Free Software Projects. In *USENIX Annual Technical Conference, FREENIX Track*.

Michlmayr, M., Fitzgerald, B., & Stol, K. J. (2015). Why and how should open source projects adopt time-based releases?, *IEEE Software*, *32*(2), 55-63. DOI:10.1109/MS.2015.55

Midha, V., & Palvia, P. (2007). Retention and quality in open source software projects. *AMCIS 2007 Proceedings*.

Midha, V., & Palvia, P. (2012). Factors affecting the success of Open Source Software. *Journal of Systems and Software, 85(4),* 895-905. DOI:10.1016/j.jss.2011.11.010

Nafus, D. (2012). 'Patches don't have gender': What is not open in open source software. *New Media & Society, 14(4)*, 669-683. DOI:10.1177/1461444811422887

Narduzzo, A., & Rossi, A. (2005). The role of modularity in free/open source software development. In *Free/Open Source Software Development* (pp. 84-102). Igi Global. DOI:10.4018/978-1-59140-369-2.ch004

O'Leary, K., Gleasure, R., O'Reilly, P., & Feller, J. (2020). Reviewing the contributing factors and benefits of distributed collaboration. *Communications of the Association for Information Systems*, *47*, 476-520.

Ostrom, E. (1990). *Governing the commons: The evolution of institutions for collective action*. Cambridge University Press. DOI:10.1017/CBO9780511807763

Patton, M. Q. (1990). *Qualitative evaluation and research methods*. SAGE Publications, Inc.

Piccoli, G., & Ives, B. (2003). Trust and the unintended effects of behavior control in virtual teams. *MIS quarterly*, 365-395.

Pinto, G., Steinmacher, I., Dias, L. F., & Gerosa, M. (2018). On the challenges of open-sourcing proprietary software projects. *Empirical Software Engineering*, *23*(6), 3221-3247. DOI:10.1007/s10664-018-9609-6

Qureshi, I., & Fang, Y. (2011). Socialization in open source software projects: A growth mixture modeling approach. *Organizational Research Methods*, *14*(1), 208-238. DOI:10.1177/1094428110375002

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Rai, A., Borah, S., & Ramaprasad, A. (1996). Critical success factors for strategic alliances in the information technology industry: An empirical study. *Decision Sciences, 27(1)*, 141-155. DOI:10.1111/j.1540-5915.1996.tb00848.x

Radtke, N. P., Janssen, M. A., & Collofello, J. S. (2009). What makes Free/Libre Open Source Software (FLOSS) projects successful? An agent-based model of FLOSS projects. *International Journal of Open Source Software and Processes*, *1*(2), 1-13. DOI:10.4018/jossp.2009040101

Rashid, M., Clarke, P. M., & O'Connor, R. V. (2017). Exploring knowledge loss in open source software (OSS) projects. In *International Conference on Software Process Improvement and Capability Determination*. DOI:10.1007/978-3-319-67383-7_35

Riehle, D. (2010). The economic case for open source foundations. *Computer, (1)*, 86-90. DOI:10.1109/MC.2010.24

Riehle, D., & Berschneider, S. (2012). A model of open source developer foundations. In *IFIP International Conference on Open Source Systems*.

Riehle, D. (2015). The five stages of open source volunteering. In *Crowdsourcing* (pp. 25-38). Springer. DOI:10.1007/978-3-662-47011-4_2

Riembauer, S., Hornung, O., & Smolnik, S. (2020). Knowledge Unchained or Strategically Overseen? Knowledge Management in Open Source Software Projects. In *Proceedings of the 53rd Hawaii International Conference on System Sciences*.

Rigby, P., German, D., & Storey, M. A. (2008). Open source software peer review practices: a case study of apache server. In *2008 ACM/IEEE 30th International Conference on Software Engineering*. DOI:10.1145/1368088.1368162

Rikkiev, A., & Mäkinen, S. (2009). Success factors for technology integration convergence collaborations: Empirical assessment. In *2009 Portland International Conference on Management of Engineering & Technology*. DOI:10.1109/PICMET.2009.5262187

Sagers, G. (2004). The influence of network governance factors on success in open source software development projects. *International Conference on Information Systems 2004 Proceedings*.

Schaarschmidt, M., Bertram, M., & von Kortzfleisch, H. F. (2011). Exposing differences of governance approaches in single and multi vendor open source software development. In *IFIP International Working Conference on Governance and Sustainability in Information Systems-Managing the Transfer and Diffusion of IT*.

Schwab, B., Riehle, D., Barcomb, A., & Harutyunyan, N. (2020). The Ecosystem of openKonsequenz, A User-Led Open Source Foundation. In *IFIP International Conference on Open Source Systems*. DOI:10.1007/978-3-030-47240-5_1

Senyard, A., & Michlmayr, M. (2004, November). How to have a successful free software project. In *11th Asia-Pacific Software Engineering Conference* (pp. 84-91). IEEE.

Singh, P. V. (2010). The small-world effect: The influence of macro-level properties of developer collaboration networks on open-source project success. *ACM TOSEM, 20(2)*, 1-27. DOI:10.1145/1824760.1824763

Singh, V., & Brandon, W. (2019). Open source software community inclusion initiatives to support women participation. In *IFIP International Conference on Open Source Systems*. DOI:10.1007/978-3-030-20883-7_7

Steinmacher, I., Conte, T., Gerosa, M. A., & Redmiles, D. (2015) a. Social barriers faced by newcomers placing their first contribution in open source software projects. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*. DOI:10.1145/2675133.2675215

Steinmacher, I., Conte, T. U., & Gerosa, M. A. (2015) b. Understanding and supporting the choice of an appropriate task to start with in open source software communities. In *2015 48th Hawaii International Conference on System Sciences* (pp. 5299-5308).

Communications of the Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

Stewart, K. J., Ammeter, A. P., & Maruping, L. M. (2005). A preliminary analysis of the influences of licensing and organizational sponsorship on success in open source projects. In *Proceedings of the 38th Annual HICSS.* DOI:10.1109/HICSS.2005.38

Strauss, A., & Corbin, J. (1990). *Basics of qualitative research: Grounded theory procedures and techniques.* Sage Publications, Inc.

Tsay, J., Dabbish, L., & Herbsleb, J. (2014). Let's talk about it: evaluating contributions through discussion in GitHub. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*. DOI:10.1145/2635868.2635882

Tiwana, A., & Bush, A. A. (2007). A comparison of transaction cost, agency, and knowledge-based predictors of IT outsourcing decisions: A US-Japan cross-cultural field study. *Journal of Management Information Systems*, *24*(1), 259-300.

Vasilescu, B., Posnett, D., Ray, B., van den Brand, M. G., Serebrenik, A., Devanbu, P., & Filkov, V. (2015). Gender and tenure diversity in GitHub teams. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems* (pp. 3789-3798).

Weikert, F., Riehle, D., & Barcomb, A. (2019). Managing commercial conflicts of interest in open source foundations. In *International Conference on Software Business*.

West, J., & O'Mahony, S. (2005). Contrasting community building in sponsored and community founded open source projects. *In Proceedings of the 38th Annual HICSS.* DOI:10.1109/HICSS.2005.166

West, J., & Gallagher, S. (2006). Challenges of open innovation: the paradox of firm investment in open-source software. *R&D Management*, *36*(3), 319-331. DOI:10.1111/j.1467-9310.2006.00436.x

Wheeler, B. (2004). The open source parade. *Educause Review*, *39*(5), 68-69.

Wheeler, B. (2007). Open source 2010: Reflections on 2007. *Educause Review, 42(1)*, 49-52.

Williamson, O. E. (1993). Opportunism and its critics. *Managerial and decision economics*, 97-107.

Communications of the Association for Information Systems

Research Article ISSN: 1529-3181

Van Wendel de Joode, R. (2004). Managing conflicts in open source communities. *Electronic Markets*, *14*(2), 104-113. DOI:10.1080/10196780410001675059

Von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: a case study. *Research policy*, *32*(7), 1217-1241. DOI:10.2139/ssrn.387500

Yenişen Yavuz, E., Barcomb, A., & Riehle, D. (2022). Problems, Solutions, and Success Factors in the openMDM User-Led Open Source Consortium (Appendix). [online] Available: https://faubox.rrze.uni-erlangen.de/getlink/fiRCBgxanJUqXTBFKPLADQ4N/Appendix%20-%20Problems%20Solutions%20Success%20Factors%20of%20openMDM%20User-Led%20Open%20Source%20Consortium.pdf

Yin, R. K. (2018). *Case study research and applications*. Sage.

Yu, Y., Benlian, A., & Hess, T. (2012). An empirical study of volunteer members' perceived turnover in open source software projects. In *2012 45th HICSS*. DOI:10.1109/HICSS.2012.97

Zhou, M., Mockus, A., Ma, X., Zhang, L., & Mei, H. (2016). Inflow and retention in oss communities with commercial involvement: A case study of three hybrid projects. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, *25*(2), 1-29.

About the Authors

Elçin Yenişen Yavuz. Elçin Yenişen Yavuz, M.Sc. is a researcher and doctoral student at the Professorship for Open Source Software at Friedrich-Alexander University Erlangen-Nürnberg (FAU), Germany. She received her master's degree of International Information Systems from the FAU, Germany. Before joining academia, she led various projects in the automotive, healthcare and pharmaceutical industries. Her areas of interest are open innovation, open source software development, collaborative software development, and digital transformation.

Ann Barcomb. Dr. Barcomb is an assistant professor at the Schulich School of Engineering, University of Calgary. Her previous post was at Friedrich-Alexander University Erlangen-Nuremberg, Germany. Dr. Barcomb received her PhD from the University of Limerick, Ireland, in 2019, with a specialization in software engineering, and a master's in information systems from Maastricht University, The Netherlands. In the course of her industry career, she worked as a software developer for multiple firms and as a community manager for RIPE NCC. From the beginning, she has been active in free/libre/open source software, organizing events, speaking at practitioner conferences, and writing for practitioner outlets. Her research is characterized by a desire to understand and generalize processes and practices within


Communications of the
Association for Information Systems

Research Paper DOI: 10.17705/1CAIS.044XX ISSN: 1529-3181

free/libre/open source software communities, to facilitate the exchange of knowledge between practitioners.

Dirk Riehle. Prof. Dr. Dirk Riehle, M.B.A., is the Professor of Open Source Software at the Friedrich-Alexander University Erlangen-Nürnberg. Before joining academia, Riehle led the Open Source Research Group at SAP Labs, LLC, in Palo Alto, California (Silicon Valley). Riehle founded the Open Symposium (OpenSym, formerly WikiSym). He was the lead architect of the first UML virtual machine. He is interested in open collaboration principles, methods, and tools, most notably open source and inner source software development. Prof. Riehle holds a Ph.D. in computer science from ETH Zürich and an M.B.A. from Stanford Graduate School of Business. He welcomes email at dirk@riehle.org, blogs at https://dirkriehle.com, and tweets as @dirkriehle.