Proceedings of the 53rd Hawaii International Conference on System Sciences | 2020

Knowledge Unchained or Strategically Overseen? Knowledge Management in Open Source Software Projects

Stefan Riembauer
University of Hagen, Germany
stefan.riembauer@gmail.com

Olivia Hornung
University of Hagen, Germany
olivia.hornung@fernuni-hagen.de

Stefan Smolnik
University of Hagen, Germany
stefan.smolnik@fernuni-hagen.de

Abstract

The term "open source software" was formally introduced in the early 2000s to describe source code which are available to the public to be used and modified by anyone. Like any innovative idea attaining a certain maturity level, open source communities have reached a degree of formalization in their structures and practices. This also holds for knowledge management and its related measures in open source communities. Therefore, we investigate the patterns and structures in communication and collaboration of the currently most successful open source software projects through a case study approach. Herewith, we reveal how the different knowledge management aspects are practiced in these internet communities. Due to the projects' success, we identify similarities as good practices and derive practical recommendations for action for other open source communities as well as research opportunities regarding knowledge management in open source software projects.

1. Introduction

Knowledge and knowledge-driven innovations constitute one of the fundamental pillars for the development of our society. Efficient use of knowledge as a resource can be ensured through targeted management [6] and lead to added value as well as a competitive advantage. One of the areas which is generally seen as very knowledge- and communication-intensive is software development [5]. Examples for knowledge-specific activities in software projects are gathering knowledge on new technologies and product domains, exchanging knowledge on local guidelines and practices as well as the identification of knowledge bearers [26]. Creating an environment that is supportive of knowledge exchange between project members can therefore be seen as an important strategic aspect for a software project's success. Since knowledge management is driven by a learning culture, knowledge-sharing intention as well as team activity [31], it is important to regard the communication amongst project members. Thus, openly accessible communication serves as a rich source of data for qualitative explorative studies regarding KM. Open source software development is a software development approach in which several individuals or organizations collaborate on a project with openly accessible source code. Research on the patterns, processes and applied methods of these mostly virtual teams have led to versatile results [2]. With open source software teams all being geographically dispersed, using new technologies and have a need for quick and efficient decision-making [11], they are at the very core of what drives KM and, therefore, are dependent on managing their knowledge well.

In this study, we will explore how the different aspects of knowledge management (KM) are practiced in the currently most successful open source software projects by examining their patterns and structures of communication and collaboration. Due to the projects' success, we see similarities as good KM practices and derive practical recommendations for current and future open source endeavors. We want to apply a KM taxonomy to the projects, so that untapped potential for the open source communities regarding KM can be identified. Therefore, we ask the following research question: *Which good practices regarding KM can be identified in successful open source software projects?*

Furthermore, we do not only want to answer this question, but also derive generalizable recommendations for action for other open source communities and virtual teams in general.

We proceed with providing theoretical foundations of open source, the building blocks of a community, and knowledge management before elaborating on the applied case study method. Then, we present our results and discuss our findings through deriving recommendations for action as well as research gaps leading to further opportunities to research KM in the context of OSS.

2. Theoretical foundations

2.1. Open source

In the early stages of information systems (IS) and programming, sharing readable source code was common practice in research and even commercial organizations [12]. When proprietary software dominated the market in the late 1990s and early 2000s, the open source movement also majorly developed and new licenses were established to clearly distinguish different types of open source software and further advance its commercial use [30].

Since there is no general and conclusive definition of open source, we adhere to the descriptions given by Aksulu and Wade [2] as well as Levine and Prietula [17]: Open source refers to intellectual production relying on loosely coordinated participants which all interact to achieve a common goal (e.g. creating a product or service of value) which is available to contributors and non-contributors for use, inspection and modification.

Such products can be source codes, hardware, documentation, or books. For our study, we focus on open source software (OSS) platforms which focus on the collaborative advancement of the source code in the form of a collaborative project. In this paper, we focus on open source software and, thus, on the product of source code. A software is considered open source if its license adheres to open-source-principles such as open access to the source code [19].

2.2. Building blocks of a community

An open source community consists of all actors involved in developing a software and represents the origin of all contributions to advance it. A major part of any community does not contribute source code, but rather requirements, bug reports and documentation [7]. It is common for communities to be spread worldwide and, thus, work asynchronously in different locations. The motivations why members contribute can be extrinsic (e.g. financial), internalized extrinsic (e.g. reputation, self-interest) or intrinsic (e.g. fun, altruism) [15]. Key success factor for high-quality software and a clear advantage to proprietary software is the size of an open source software community since the large amount of members and testers ensures fast development and extensive quality assurance [25]. Thus, gaining and retaining community members is key for any community's sustainability.

An OSS community's building blocks are a supervisory and/or support body, its governance, its hierarchy and roles, the culture as well as collaboration tools. Because the supervisory body supports the OSS community's aims and might also offer financial support, it can exercise influence on the products the community develops. Thus, OSS communities supported by foundations instead of companies are seen as the more neutral form of supervision and support [12]. In the context of OSS projects, the governance constitutes the instrument to steer, control and coordinate individuals and organizations which collectively contribute to a project [18]. The governance represents core principles and goals as well as encompassing social (e.g. communication, culture and development perspectives for members) and organizational aspects (e.g. decision-making and role assignment) of the entire community [12]. While some of the supervisory bodies enjoin a standardized governance on OSS projects, there are also OSS projects without any explicit governance.

Instead of a regular top-down hierarchy, many OSS communities have an onion-shaped organizational structure [4] with the core consisting of project leaders and core developers who contribute the majority of code contribution. The next layer are the co-developers who do the bug reporting and feature request as well as occasionally contributing to the code and fixing bugs. The active users are on the next layer. They file bug reports and request features, but do not contribute to the code at all. The furthest layer – the outer layer of the onion – are the passive uses which do not contribute at all but use the code or software. The latter are not always welcomed in OSS communities as these communities often embrace a culture of meritocracy in which the individual performance measured by contributions plays the most important role for the reputation and position of project members [28]. Decision-making in OSS projects can either be very absolutist or completely democratic. In both scenarios, consensus still plays an important role as forks should generally be avoided.

Collectively working on software requires many different tools to collaboratively work on the source code as well as for discussion and general communication. We will further elaborate on mechanism and tools used for these purposes in our results section.

2.3. Knowledge management

The KM domain was defined and viewed from different perspectives in the last decades. While Davenport and Prusak [9] focus on codification and measures for knowledge explication, Nonaka and

Takeuchi [21] chose an approach that sees knowledge as experience-based and not to be simply stored in repositories. We choose an approach incorporating both views on knowledge as we assume that KM cannot manage knowledge itself but rather offer support to connect, expand and encourage use of any existing knowledge base. This view is based on the assumption of a differentiation between data, information and knowledge in which data are syntactically arranged characters that can, together with their contextual interpretation, be information for the recipient [24]. In order to find meaning in information and for it to become knowledge, any person has to understand and use their experience with the context and environment when generating or using knowledge [14].

KM is an organization's systematic and conscious effort to improve, preserve and apply knowledge in a manner that adds value and supports task fulfillment as well as improvement of the organization's overall position [13]. KM also includes the systematic use of organizational instruments and communication technologies in different phases or processes.

There are many different approaches to name, split and describe KM processes. Some give a rather general overview of KM enablers, process, intermediate outcome and organizational performance [16], whereas other frameworks already attribute specific names to certain process steps in KM, such as knowledge creation, retrieval, transfer and application [3].

We want to use a process view that clearly distinguishes the different processes from each other to attain a holistic overview, thus, chose to apply the six core and two strategic processes described by Probst et al. [24], also called the eight building blocks of KM (as seen in Figure 1). This framework gives us a very distinct view that makes it possible to investigate which specific measures occur with regard to certain processes. This is especially important since we want to start out with granular steps with the option to find patterns or process groups united by certain measures. With less distinct process frameworks, certain measures might only be attributable to parts of a process (group), thus leading to ambiguous results.

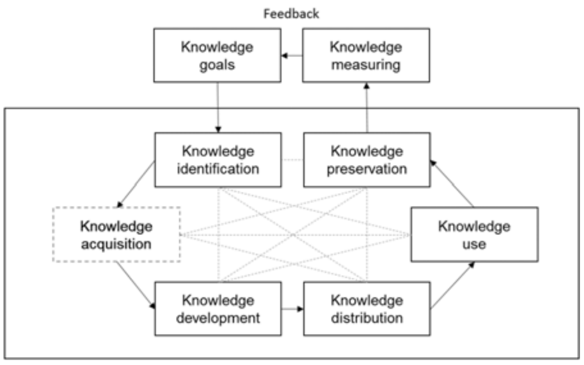

Figure 1. Building blocks of knowledge management (after [24])

The process of knowledge identifications serves the purpose of making knowledge transparent by supporting search efforts as well as providing a neat depiction of data, information and internally and externally available competences. Knowledge acquisition is useful for integrating external knowledge into the organization by, for example, hiring experts or acquiring innovative companies. Knowledge development is complementary to knowledge acquisition. Through knowledge development, competences that do not exist within the organization are newly developed, for example, by creatively using new ideas or creating new products. This can happen individually or in a team. Sharing or distribution knowledge makes individual knowledge available to the organization. This process can only happen effectively if sufficient infrastructure and systems which allow to share (push) or retrieve (pull) knowledge. Knowledge use can be seen as the implementation phase of KM during which knowledge is applied productively and turned into tangible results. The specific preservation of knowledge in documents or other objects through appropriate selection, storage and periodic updating in so-called knowledge repositories happens during knowledge preservation.

The two strategic processes knowledge goals and knowledge measuring are focused on embedding KM into organizational goals. Normative and strategic goals are specified during the process of formulating knowledge goals. Knowledge measuring then reveals how well and to what degree these goals are met.

3. Method

Due do the publicly available data on OSS projects, it is possible to choose qualitative and quantitative approaches to evaluate KM in OSS communities. We decided to apply the research method case study because of its suitability for our purposes to investigate the synergies of organizational and communicative aspects [20]. To strengthen our arguments and be able to integrate our results in a broader context, we conducted a comparative case study by analyzing each case before comparing the different cases. We take on a positivist epistemic stance since we want to identify and show the existence of KM measures throughout our case study research.

The identification of meaningful cases is a critical factor for case study research – while the term *case study* is generally broadly defined [10]. In this study, we see an OSS community and their measures taken to collaborate, exchange information and store information as the primary content of a case.

The first step is to gather measures for collaboration and communication in OSS communities through a brief literature review as seen in Table 2. Then, we identify suitable cases of OSS communities. For transparency reasons, we adhere to the following criteria: the projects have to be independent and deemed successful as well as still being in the stage of developing source code and showing a high degree of activity. As a starting point, we used the list of 30 highest velocity OSS projects provided by the Cloud Native Computing Foundation (CNCF) [29]. Then, we eliminated those OSS projects from our list of potential cases which were controlled by a single company and only one project kept per foundation to eliminate extremely similar projects. Furthermore, we removed projects not producing source code as their primary aim. Finally, to ensure up-to-dateness, we also excluded projects with a lifespan of over 10 years.

For our analysis, we use any documentation available to us, ranging from purely descriptive documents to observation of open collaboration. Instead of deriving the codes inductively from the sample, we maintain the building blocks of KM, and thus, used the approach of deductive coding. Therefore, we derive codes from the definitions of the identified KM core processes [24] and create coding guidelines as seen in Table 1.

During data analysis, we sequentially examine each unit of analysis per case for the occurrence of any code. Additionally, we compile a case summary for each case and then compare the cases to each other to review the OSS communities for similarities and differences. Lastly, we derive recommendations for action and research gaps as seen in Figure 2.

Table 1. Coding guideline

KM building block	**Coding guideline "Measures to…"**	**Example**
Goals	…depict and define goals regarding knowledge handling	Cultural guidelines
Preser-vation	… specifically preserve knowledge through selection and storage	Repository, database
Acqui-sition	… support acquisition of external new knowledge	Recruiting, inclusion
Identifi-cation	… support localization of knowledge and knowledge bearers	Search mechanism, dashboard
Use	… productively apply knowledge to add value	Governance
Develop-ment	… develop new skills within the organization by using people's ideas and creativity	Collaboration tools, virtual conference rooms
Distribu-tion	… support knowledge sharing within the organization	Newsletter
Measur-ing	… examine determined goals	Controlling system

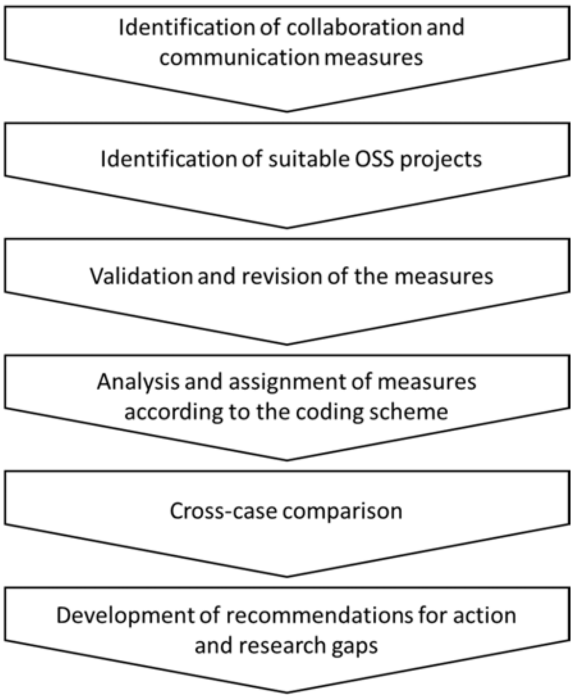
Identification of collaboration and communication measures
Identification of suitable OSS projects
Validation and revision of the measures
Analysis and assignment of measures according to the coding scheme
Cross-case comparison
Development of recommendations for action and research gaps

Figure 2. Research process

4. Results

4.1. Literature analysis

To support the KM building blocks presented in the previous chapter, relevant systems and measures that go beyond the examples given in Table 1 need to be identified. Therefore, we conducted a literature search for "knowledge management" and "open source" in the AIS electronic library as well as the EBSCO business source complete. We decided to limit our keyword search to titles and abstracts to find the papers covering the topic of KM in open source projects most prominently. Then, we manually searched the papers for studies that specifically mention KM measures. Hence, we found seven publications that reviewed organization, collaboration and governance in OSS communities. The results of this analysis can be found in Table 2.

Table 2. Literature analysis

System / measure	**Reference(s)**
Version control systems	[7, 12, 27, 28]
Issue tracker	[12, 23, 27, 28]
Forum	[7, 12, 23, 27, 28]
Mailing list	[1, 7, 12, 23, 27]
Wiki	[12, 27, 28]
Instant messaging	[12, 28]
Project website	[27, 28]
Governance	[12, 18, 27]
Social networks	[28]
Video conference systems	[1, 12]
FAQ	[27, 28]
HowTo	[28]
Meetings (face-to-face)	[7, 12]
Mentoring	[12]
Weblog	[28]

During the subsequent validation of results, we came to the realization that not all measures are used in the case studies or are not relevant for our investigation:

- HowTos and FAQ are not seen as relevant independent systems but rather illustration facilities for online documentation
- Only one of our cases uses a wiki and it is seen of lower priority for documentation
- The use of social networks could be validated, but is seen as insignificant for collaboration within the community
- Only one of our cases uses an explicit discussion forum

During validation, we saw that online documentation plays an important role for explicating knowledge and, thus, added this measure to our study.

4.2. Case descriptions

In order to fully grasp the context of each case, the number of investigated cases should optimally be between two and four cases [22]. Therefore, we decided to include four cases after applying our criteria mentioned in Section 3: Kubernetes, Node.js, Apache Mesos, and OpenStack (see Table 3). These projects are among the largest and most active OSS projects and have foundations as their supervisory body with democratic hierarchies for decision-making.

Table 3. Cases included in our study

Project	**# of authors**	**Estab-lished**	**Supervisory body**
Kubernetes	1728	2014	CNCF
Node.js	507	2009	Node.js Foundation
Apache Mesos	500	2011	Apache Software Foundation
OpenStack	>588	2010	OpenStack Foundation

Kubernetes is a platform for orchestration of software containers. This is a new form of isolating applications especially popular with microservice architectures and cloud solution providers to optimally use their scaling potential. Most of developing the source case happens in special interest groups and other working groups which all are responsible for single elements of the platform.

Node.js is a JavaScript runtime environment to operate scalable network applications from the server-side. Single development aspects are tackled in working groups which use own repositories in the version control system.

Apache Mesos is a software enabling dynamic and efficient administration of a computer cluster's resources. The product is used for research and commercial purposes to administrate workloads in datacenters. For larger elements, work is done in special interest or working groups.

OpenStack's aim is to ultimately provide the dominating platform for public and private cloud operation. The project consists of several individual sub-projects with own repositories and teams that develop the platform's different segments. An overarching aim is to coordinate the sub-projects in such a manner that the users feel like they are using an integrated platform.

4.3. Case study results

The results of coding our case study material can be seen in Table 4. Since we can only properly compare the cases by determining similarities, we omitted measures which could only be identified for one of the cases. We grouped the results by measure and frequency of occurrence. We also assessed each measure regarding its appropriability for each of the KM building blocks.

An explicit **governance** in which project structures, hierarchies, and general conditions (like a code of conduct) for collaboration are defined can be found in all four OSS projects. Kubernetes even has a special commission to inform about breaches of the code of conduct as well as mentoring as part of the chores set in the governance. Apache Mesos even states in its governance that knowledge sharing is a core principle. Governance supports all building blocks except measuring knowledge. The guidelines defined in the governance have an impact on the entire project communication and all used collaboration systems. They help to establish common trust as well as a tolerant, open and fair culture which positively influences KM.

Project websites are starting points for information on the OSS projects. The most important information are presented in a condensed manner with links to other sources and tools. This way, project websites serve as a central portal for users, developers and current or potential community members. Kubernetes' and OpenStack's project websites offer an active menu navigation which offers relevant further information after several questions are answered. Knowledge is shared and a point of contact with potential new members is provided. Through the project website, knowledge identification, acquisition and distribution are supported.

Online documentation serves as an information source for all target groups. It contains much information, for example regarding the architecture, code, governance, collaboration processes, and necessary cues on communication and collaboration tool usage. Some OSS projects integrate the online documentation into their project website, others only reference is there but have it hosted by online services for software development such as github. Knowledge distribution and knowledge use are both supported by online documentation since it enables anyone interested to interpret and absorb the codified knowledge stored in information objects.

All projects also have a **version control system**, a typical feature of OSS projects, only serving for common editing and administrating the software source code for a vast amount of time. Such systems also can be used to collaboratively process further types of legible documents. The authors and the entire history always stay available in these systems. Depending on the role, community members are entitled to submit contributions, validate them and integrate them into repositories. To change the source code, a defined process has to be passed through during which the respective repository is usually cloned. This clone can then be changed and has to undergo extensive testing. Only after successful testing, the changes can be integrated into the main repository by entitled community members. In all four cases, Git is used as version control system and through the use of the commercial platform Github, additional services such as issue-tracking-systems, discussion forums and review boards are supplied. Here, the KM building blocks backed are knowledge preservation, knowledge identification, and knowledge distribution. The version control system is complementary to the online documentation. Thus, knowledge bearers can spread the selected and codified knowledge.

Instant messaging is a measure also used by all four projects, used for direct exchange between community members as well as between community members and users. Although asynchronous communication is possible with instant messengers, they are usually used for synchronous communications replacing face-to-face conversations. Channels enable a discussion on working groups as well as certain topics. The used applications enable both a one-to-one as well as a one-to-many communication. Kubernetes and Apache Mesos also use the provider Slack which also offers a comfortable search function. Regardless of the used application, all four cases prioritize logging, archiving and open access to public discussions – encouraging knowledge preservation, identification, development, and distribution. New knowledge can be developed especially well in discussions on new functionalities combining existing knowledge of different knowledge bearers, in turn leading to new knowledge.

Issue-tracking-systems have replaced the former bug-tracking systems in all four cases. Technically it is the same system, but the use has been extended from bugs to requirements and other general discussions. Typically, anyone can create an issue. Often, issue-tracking-systems are split according to the project's elements and assigned to singe sub-projects or working groups. Issue-tracking-systems play a role for knowledge acquisition, development and distribution. They enable users to document and share their views, ideas, problems or bugs. Especially users with complimentary or contrary views can develop new knowledge by exchanging knowledge.

Despite their focus on virtual collaboration, all cases also have face-to-face **meetings**. These can be regionally held in small groups as well as with a larger group at a community conference. All projects use the platform meetup to organize their meetings. Kubernetes has so far had over 70 meetings with 20.000 members participating in Germany – the number of participants worldwide is much higher. Node.js has a bi-annual large conference balled Collab Summit. Here, active and potentially future community members meet to work on the project or discuss it. Meetings enforce the KM building blocks knowledge development and knowledge distribution.

Another personal measure in addition to meetings is the concept of **mentoring**, applied by all four cases. Mentoring aims at building an individual's competencies through the support of an experienced mentor. Mentoring programs are an important measure to integrate new members as they enable a compact and effective onboarding through short-term activities such as working on the first tasks as well as long-term activities. Kubernetes, for example, offers group mentoring. OpenStack and Kubernetes also participate in externally sponsored programs such as Google Summer of Code and Outreachery, in which students are given the opportunity to work on projects through scholarships to gather valuable experience in software development. Therefore, especially knowledge acquisition and knowledge distribution are supported.

The last measure all four cases display are **weblogs**. These allow for registered authors to publish articles in an uncomplicated manner. Official project weblogs are linked on the project websites and are used to inform members and users about important news and events. OpenStack additionally uses a weblog-aggregator to bundle different weblog posts in a single newsfeed for community members. Here, knowledge distribution and knowledge preservation are the focus.

Kubernetes, Apache Mesos and OpenStack also have **mailing lists** to communicate with their members in a mostly asynchronous manner. Mailing lists are listed on the project websites and usually categorized into target groups and topics. Additionally, web applications to display the courses of discussion well-structured in browsers are used. These also allow for archiving. Besides the focus on asynchronous aspect of communication, the advantages of mailing lists hardly differ from those of instant messaging. Hence, mailing lists are of importance for knowledge preservation, knowledge identification, knowledge development and knowledge distribution.

Video conferences are used by Kubernetes, Node.js and OpenStack for synchronous and direct communication and discussion. Since gestures and mimics can be displayed through video, these video conferences enables a more multifaceted communication than instant messaging and mailing lists. All three projects using video conferences conduct these through the commercial provider Zoom and upload their recordings of these sessions on Youtube afterwards. Video conferences are a very efficient way of exchanging knowledge. Archiving the recordings also supports retaining knowledge. Consequently, knowledge distribution, development and preservation are strengthened by video conferences.

Lastly, Kubernetes and Node.js send out **newsletters** which are similar to weblogs regarding their content. They generally serve the purpose of spreading community news. After registration, anyone interested will periodically be informed about the project's news. The content is usually not written by one author, but compiled from contributions of several authors and different parts of the community. After being sent out, the newsletters are archived so that they remain publicly available. Newsletters are similar to weblogs and, thus, support the KM building blocks of knowledge distribution and knowledge preservation.

5. Discussion

To derive value from our results, we want to give recommendations for action for virtual teams in general as well as specifically for OSS projects. Furthermore, we want to show some opportunities for further research on KM in OSS.

5.1. Recommendations for action

For virtual and distributed teams, we would like to give the following recommendations for action: establishing guidelines, support through IT systems, strengthening knowledge identification, and consolidating measures.

As discussed, the release of *governance guidelines* is still not common practice in many OSS communities. But not just OSS communities, but any community in general should work towards establishing guidelines in governance documents as well as supporting an open, transparent and trustworthy culture with a code of conduct. A good example of such a guideline is Apache Mesos that requires any important decision only to be made based on written communication in mailing lists as well as making archived discussions publicly available.

Beside management support through guidelines, it is also important for the implemented *IT systems* to be sufficiently *supportive* of knowledge explication. In the results, we identified many IT measures, which can be used to identify weaknesses or unused potential. Once weak points are clear, a slow and stable introduction of new technologies should be ensured, together with the implementation of according processes and roles. It is also important to keep the interdependency of the different KM building blocks in mind.

Entering OSS projects can be facilitated through different instruments, as indicated in our results section. Herewith, *knowledge identification* can massively be *strengthened*. Recommendations on how to approach and use the project's communication mechanisms, interactive menu navigation and well-structured project websites which clearly show which information for which purpose can be found provides potential community members with a quick and easy entry into the project. Since recruiting new members is one of the key criteria for a sustainable OSS community, we recommend a focus on knowledge identification.

Although many measures for communication and collaboration were identified in our analysis, there are hardly any redundancies in their purpose. For example, discussion forums and wikis are hardly still used in the examined cases, yet their purpose is still fulfilled by other tools already in use. Therefore, we recommend to cater one purpose through one tool only, while one tool can serve more than one purpose. To *consolidate tools*, it is important to gain an overview over the functionality of each tool. To determine which tool serves which purpose, the previous recommendation for action can also be useful.

The case studies show that knowledge is managed through an array of measures in these OSS communities. Since these OSS projects are organizations similar to large companies, professionalizing KM by applying best practices from commercial organizations in desirable. A possibility here could be to learn from current research.

5.2. Research gaps

The two main opportunities for further research to advance both theory and practice of KM in OSS are studying the strategy layer and success of KM activities.

Although OSS communities have served as research objects in past research, they have hardly been investigated from an explicit KM view. In our study, we examined how KM processes are supported by communication and collaboration mechanisms. This means that, from a business engineering point of view, we remained in the process and systems layer. But what about the strategy layer? Research should focus on answering the question if and how conscious KM efforts on the strategic layer of OSS communities are made. To answer this question, qualitative research methods could be used – such as interviews with OSS project leaders.

To identify good practices of handling knowledge in OSS communities, we compared several successful and independent communities. If and how the application of KM activities has any influence on project success remains unanswered. Is there an influence of consciously conducting KM and according measures on project success? Answering this question positively would strengthen the case for KM in OSS communities. Here, another case study would be useful, comparing OSS communities applying the identified best practices with OSS communities that do not apply them. Dimensions to compare could be the amount of contributions to the source code or release frequency. Furthermore, single KM processes or building blocks could be examined. This could be done by analyzing recruiting efforts or engagement of new community members.

6. Conclusion

In this study, we confirm the assumptions of previous experience as well as the literature analysis and ascertain a lack of KM research on the subject of OSS. Nonetheless, is becomes clear that the investigated OSS communities implicitly manage knowledge in a structured manner. Coding and categorizing according to measures shows that the KM building blocks [24] are overall supported. Because of similarities regarding the use of communication and collaboration tools, the assignment to building blocks / processes is relatively homogeneous. Our findings also show that knowledge distribution is the most supported process, while there are no measures supporting knowledge measuring and only one measure for knowledge goals. This leads us to one of the research gaps, since the not supported processes are of strategic nature. Thus, the strategic observation of KM in OSS is still lacking.

By investigating the area of knowledge exchange in OSS projects, we contribute to filling one of the research gaps mentioned in previous studies [2]. Furthermore, we want to inspire researchers to conduct further studies in the field of KM in OSS by identifying two research opportunities.

Limitations of this study are mainly due to the chosen qualitative research method. Our findings could be supported by further studies on this topic using quantitative methods, such as a survey conducted amongst community members to find out which measures they use for the respective building blocks / processes. We also acknowledge that a generalization of our findings is only possible in theory as we only analyzed four cases – even though we used formal criteria and focused on diversification when selecting the cases.

7. References

[1] Aalst, W., P. Loucopoulos, K. Lyytinen, J. Mylopoulos, B. Robinson, N.M. Sadeh, M.J. Shaw, and C. Szyperski, eds., Design Requirements Engineering: A Ten-Year Perspective: Design Requirements Workshop, Cleveland, OH, USA, June 3-6, 2007, Revised and Invited Papers, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[2] Aksulu, A. and M. Wade, “A Comprehensive Review and Synthesis of Open Source Research”, Journal of the Association for Information Systems, 11(11), 2010, p. 576.

[3] Alavi, M. and D.E. Leidner, "Review: Knowledge Management and Knowledge Management Systems: Conceptual Foundations and Research Issues", MIS Quarterly, 25(1), 2001, p. 107.

[4] Al-Marzouq, M., L. Zheng, G. Rong, and V. Grover, “Open Source: Concepts, Benefits, and Challenges”, Communications of the Association for Information Systems, 16, 2005.

[5] Birk, A., D. Surmann, and K.-D. Althoff, eds., Applications of Knowledge Acquisition in Experimental Software Engineering: Knowledge Acquisition, Modeling and Management, Springer Berlin Heidelberg, 1999.

[6] Bultrini, L., J. Sempéré, S. McCallum, and W. Newman, eds., Knowledge management in libraries and organizations, De Gruyter Saur, Berlin, 2016.

[7] Crowston, K. and J. Howison, “The social structure of free and open source software development”, First Monday, 10(2), 2005.

[8] Crowston, K., Q. Li, K.i. We, U.Y. Eseryel, and J. Howison, “Self-organization of teams for free/libre open source software development”, Information and Software Technology, 49(6), 2007, pp. 564–575.

[9] Davenport, T.H. and L. Prusak, Working knowledge: How organizations manage what they know, Harvard Business School Press, Boston, Mass., 2000.

[10] Denzin, N.K. and Y.S. Lincoln, eds., The Sage handbook of qualitative research, 4th ed., Sage, Los Angeles, Calif., 2011.

[11] Du Plessis, M., "Drivers of knowledge management in the corporate environment", International Journal of Information Management, 25(3), 2005, pp. 193–202.

[12] Haff, G., How Open Source Ate Software, Apress, Berkeley, CA, 2018.

[13] Holsapple, C.W. and K.D. Joshi, “A formal knowledge management ontology: Conduct, activities, resources, and influences”, Journal of the American Society for Information Science and Technology, 55(7), 2004, pp. 593–612.

[14] Jennex, M.E., ed., Knowledge management in modern organizations, IGI Global (701 E. Chocolate Avenue Hershey Pennsylvania 17033 USA), Hershey, Pa, 2007.

[15] Krogh, G. von, S. Haefliger, S. Spaeth, and M.W. Wallin, “Carrots and rainbows: Motivation and social practice in open source software development”, MIS Quarterly, 36(2), 2012, pp. 649–676.

[16] Lee, H. and B. Choi, "Knowledge Management Enablers, Processes, and Organizational Performance: An Integrative View and Empirical Examination", Journal of Management Information Systems, 20(1), 2014, pp. 179–228.

[17] Levine, S.S. and M.J. Prietula, "Open Collaboration for Innovation: Principles and Performance", Organization Science, 25(5), 2014, pp. 1414–1433.

[18] Markus, M.L., "The governance of free/open source software projects: Monolithic, multidimensional, or configurational?", Journal of Management & Governance, 11(2), 2007, pp. 151–163.

[19] McGowan, D., "Legal Implications of Open-Source Software", University of Illinois Law Review, 2001, 2001, p. 241.

[20] Myers, M.D., "Qualitative Research in Information Systems", MIS Quarterly, 21(2), 1997, p. 241.

[21] Nonaka, I. and H. Takeuchi, The knowledge creating company: How Japanese companies create the dynamics of innovation, Oxford Univ. Press, New York, 1995.

[22] Palmberger, M. and A. Gingrich, "Qualitative Comparative Practices: Dimensions, Cases and Strategies", in The SAGE handbook of qualitative data analysis, U. Flick, Editor. 2014. Sage: London.

[23] Peng, G., "Critical Success Factors for Open-Source Innovation: The Case of Open Source Software Development", Issues in Information Systems, 10(2), 2009, pp. 157-164.

[24] Probst, G., S. Raub, and K. Romhardt, Managing knowledge: Building blocks for success, Wiley, Chichester, 2002.

[25] Raymond, E.S., The cathedral and the bazaar: Musings on Linux and open source by an accidental revolutionary, O'Reilly, Beijing, 2001.

[26] Rus, I. and M. Lindvall, "Knowledge management in software engineering", IEEE Software, 19(3), 2002, pp. 26–38.

[27] Scacchi, W., "Free/open source software development", in The 6th Joint Meeting on European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering companion papers, A. Bertolino and I. Crnkovic, Editors, The 6th Joint Meeting, Dubrovnik, Croatia, 3/9/2007 - 7/9/2007. 2007. ACM: New York, NY.

[28] Scacchi, W., "Collaboration Practices and Affordances in Free/Open Source Software Development", Collaborative Software Engineering, 2010, pp. 307–327.

[29] https://www.cncf.io/blog/2017/06/05/30-highest-velocity-open-source-projects/, accessed June 15th, 2019.

[30] van Reijswoud, V. and A. de Jager, Free and open source software for development: Exploring expectations, achievements and the future, Polimetrica, Monza (Milano), 2008.

[31] Yu, S.-H., Y.-G. Kim, and M.-Y. Kim, "Linking organizational knowledge management drivers to knowledge management performance: An exploratory study", in Proceedings of the 37th Annual Hawaii International Conference on System Sciences: 5-8 January, 2004, Big Island, Hawaii, R.H. Sprague, Editor, 37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the, Big Island, HI, USA, 8/1/2004 - 8/1/2004. 2004. IEEE Computer Society Press: Los Alamitos, Calif.