




A Systematic Mapping Study of Empirical Research Methods in Software Ecosystems

Larry Abdullai¹, Hatef Shamshiri¹, Hasan Mahmud¹, Muhammad Hamza¹, Essi Aittamaa¹, Jaakko Vuolasto¹, Mikhail O. Adisa¹, Roope Luukkainen¹, Sonja M. Hyrynsalmi¹, Niina Mässeli¹, Nasreen Azad¹, Bahalul Haque¹, Juha-Pekka Joutsenlahti¹, Wondemeneh Legesse¹, Ahmed Abdelsalam¹, Anastasiia Gurzhii¹, Jouni Ikonen¹, Slinger Jansen^{1,2}, and Casper van Schothorst² (✉) 

¹ LUT University, Lappeenranta, Finland

² Utrecht University, Utrecht, The Netherlands
casper@vanschothorst.com

Abstract. The field of software ecosystems is rapidly maturing and significant numbers of articles are published each year to further develop our understanding of this concept and support innovation through it. The growth of the field also brings along challenges, such as findability and reusability of research results, coordination of research initiatives, and significant review pressure on members of the community. In this mapping study of empirical research methods in the field, we show that few studies do a good job of reporting their research methods and results. Using data from the study, we provide guidelines for performing empirical research in software ecosystems.

Keywords: Empirical research · Software ecosystems · Software engineering · Platform management · Research validity

1 Introduction

Technology has a profound impact on business models and the way companies are successful. As stated by Jacobides [8] companies are no longer “independent strategic actors” but they depend for their success on collaboration with other companies in an ecosystem spanning multiple sectors. The virtual character of software have enabled these developments even more profound in the software industry. While in the early days of software engineering a software product was the result of effort of an independent software vendor to create a monolith product, modern software strongly relies on components and infrastructure from third-party vendors or open source suppliers [9]. Software ecosystems, defined as a set of actors that functions as a unit and collectively serve a market for services and software, usually based around a platform or technology [14] have become a key component in the success of companies in the software industry. Even companies that compete with each other in the same market, also collaborate for

example on development projects. This hybrid behavior comprising competition and cooperation has been named *coopetition* [20].

With these developments the importance of understanding how to set-up and manage software ecosystems has increased. It becomes of vital importance that high quality knowledge is developed through solid academic research and reported in transparent ways. There is a growing body of research papers and SLR's covering Software Ecosystems but there is still clear upside in improving the quality the research standards and methods. In a mapping of systematic literature studies Garcia et al. [7] conclude that the mapping shows that the research of software ecosystems is in its infancy. They refer to Manikas [d] whose SLR on software ecosystems is a leading paper, being cited more than 600 times and who is agreeing with this statement. But the SLR of Manikas is being criticized by Wohlin et al. [24] for being the Manikas the sole author while their research shows that having more than one researcher is very valuable.

Researchers in the software ecosystem domain use, among others, empirical research methods to understand how these networks of companies and individuals produce and maintain complex, continuously developing software systems. The objective of our study is to explore how software ecosystem researchers use empirical research methods in their work, with the larger goal of maturing the field. We hypothesize that, similarly to other young domains such as software engineering research [4], researchers can become better at executing and reporting their findings and conclusions. The research question that drove this research is: *how do software ecosystem researchers use empirical research methods?*

This paper reports on early findings of a mapping study of the empirical research methods observed in literature on software ecosystems. We have initially zoomed in on case studies, as that appear to be the prevalent method. The rest of the paper is organized as: Sect. 2 outlines the systematic mapping study research method, Sect. 3 outlines the result analysis of the extracted dataset, Sect. 4 describes the top 4 papers having the highest value in meeting the ACM SIGSOFT empirical standards for software engineering, Sect. 5 represents the recommendation of the authors after the analysis, Sect. 7 concludes the paper and outlines the future research directions.

2 Systematic Mapping Study Research Method

To arrive at a representative and quality sample of studies from the field of software ecosystems, we commenced our systematic mapping study following the guidelines proposed by Kitchenham et al. [11] and Petersen et al. [15]. We started with defining our research question and designing our mapping studies protocols. Since the ecosystem terminology is used divers definitions in different ways, we chose keywords that would limited the search to finding scientific articles relating to software ecosystems, rather than any other forms of business ecosystems or ecological ecosystems.

This enabled us to extract keywords necessary to answering our research question and using them as a basis to form our search string (“software” AND “ecosystem” AND (“platform” OR “open source” OR “governance” OR “package”). We used some of the orthodox guidelines in conducting systematic mapping. However, we applied some exotic flavor in our strategy for selecting the literature as illustrated in Fig. 1 and described in the following paragraphs.

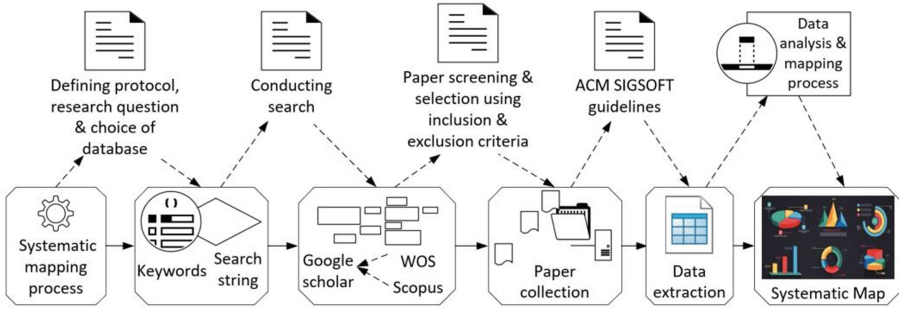


Fig. 1. Systematic mapping study design and execution process

We first chose Google scholar as our primary database search engine. Secondly we divided the authors into groups of two and with the assignment per group to review 20 papers from a selected year from the period between 2011 and 2021. The two authors per group should then discuss and agree between themselves the suitability of the paper selected. We selected this strategy to ensure that relevant papers were evenly selected from each year and to avoid sampling and selection bias.

During the search, it became evident that, due to the large number of papers that were returned as the results of our query, it was not feasible to read every single article. The authors therefore leveraged the “sort by relevance” feature of the Google Scholar search engine to arrange the generated literature. Other databases such as Scopus and Web of Science together with a set of more formal decision criteria were utilized to validate that the most relevant papers were selected through Google Scholar. The following set of exclusion and inclusion criteria were used: (i) Only studies relating to software ecosystems with focus on platform, open source, governance, or package were selected. (ii) Only papers with twenty or more citations were selected except for articles published in 2020 and 2021, to ensure a minimum degree of quality. (iii) Only full papers were selected. We define a full research paper as an articles with not less than seven pages. (iv) Only open access literature or those that could be accessed through legal sources were considered. Finally, (v) systematic review studies were excluded from the list.

After applying these inclusion and exclusion criteria, the final literature selection process relied on the researchers’ thorough reading of the full paper since the data set was arranged in order of relevance, to select ten papers per author each for analysis. Overall, a total of 89 sample papers were included in the mapping study. Although there is the likelihood that we might have missed some potentially relevant papers giving the selection approach we adopted, we believe that our strategy allowed us to select papers year by year evenly and fairly which is representative of the topic under investigation.

For the analysis of the research methods, we have used the ACM SIGSOFT standards for empirical research in software engineering [17]¹. This set of standards includes a quality guide for a selected number of different research methods, e.g., case study, mixed methods, or grounded theory. These empirical standards have been proposed to provide generic standards for the evaluation of empirical research or research that uses data, as

¹ Latest standards <https://github.com/acmsigsoft/EmpiricalStandards/>.

defined by ACM SIGSOFT [f]. The standards provide a checklist per research method that is divided in essential, desirable and extraordinary attributes. For each article we have used the checklists to compile a review that is consistent and seen as reasonable by the SE community.

Additionally, the requirements set by the scientific community make it sensible to follow the ACM SIGSOFT standards. Since we included scholarly peer-reviewed articles (e.g., journal papers) as well as conference proceedings, the predetermined criteria helped us focus on the necessary components that the chosen papers should contain. We used 19 criteria (e.g., justification for the choice of case(s) or object(s) that have been studied, description in rich detail, and presentation of the precise chain from observation to findings, etc.) to evaluate the chosen research papers for further analysis.

3 Results

Using a two-step analysis, we started with the classification of used research methods and secondly utilized the checklists of the ACM SIGSOFT Case Study standards. In classification of used research methods, we initially identified the research method from each selected paper and then counted the frequency of each research method type. In ACM SIGSOFT Case Study standard analysis, we analyzed the selected case study articles, including mixed method studies and repository mining case studies. Analysis results are presented in following Sections.

3.1 A Classification of Used Research Methods

The term research method describes an approach of conducting research, specifically the numerous sorts of activities used to systematically address the research problem that is predicated on assumptions and provide a rationale for the decisions made. A research methodology may use a variety of research methods which are the tools used to gather and analyze data [27].

To determine the methodology adopted by the selected studies, a research method analysis on a sample of the collected papers was conducted. The results of the performed analysis can be seen in Fig. 2. It can be noticed that most researchers have conducted case studies to conduct the research. Furthermore, the top three research methods used are, in order:

Case Study. Case study research is an empirical method aimed at investigating contemporary phenomena in their context [18]. With characteristics of the flexible type, coping with the complex and dynamic characteristics of real-world phenomena, like software engineering, bases its conclusions on a clear chain of evidence, whether qualitative or quantitative, collected from multiple sources in a planned and consistent manner, and it adds to existing knowledge by being based on previously established theory, if such exist, or by building theory.

Mixed Methods. Mixed methods research is a combination of multiple research methods. The method uses both quantitative and qualitative data collection and analysis

strategies. The mixed methods research approach can be utilized if the researcher desires a more robust, multifaceted viewpoint, one example could be using Interviews to supplement statistical results from a survey [25].

Design Science. Design science research is a qualitative research methodology that simultaneously produces knowledge about the process used to create an artifact and the artifact itself, with the object of the study being the design process. Furthermore, design science research seeks to provide information that is prescriptive for professionals in a field and to share empirical leanings from studies of the prescriptions used in context [16]. Such information is known as “design knowledge” since it aids practitioners in creating solutions to their issues [5]. On the other hand, the least used methods are exploratory studies, interviews, and repository mining.

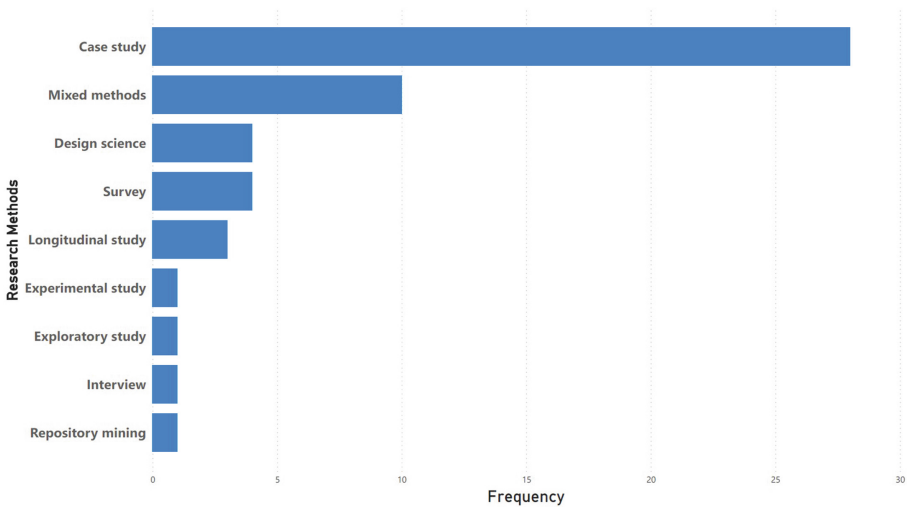


Fig. 2. Classifiable and identifiable research methods within the articles of the study.

3.2 An Analysis of Utilized ACM SIGSOFT Case Study Standards

In Table 1 we present an in-depth analysis of the research methods encountered in the case study articles that were part of the mapping study. We have analyzed 51 articles that were either pure case studies, mixed method studies, or repository mining studies, hence the number is larger than the number of pure case studies mentioned in Fig. 2.

When looking at the overall results, it is pleasing to observe that most studies check the essential aspects. Most studies explain in rich detail the object of study and why the case study was done. However, authors frequently neglect to mention the type of case study (e.g., exploratory, or explanatory). More problematic is the lack of explanation of how the case studies lead from observations to findings. While this in part can probably

be blamed on page limitations, it is nowadays increasingly common to also make the case study report available, in which this chain of evidence could be provided.

When looking at the desirable aspects, things become more dire. While several studies use triangulation and present evidence such as interview quotations (approximately half), many do not. Furthermore, only approximately a fourth of studies use cross-checking techniques for corroborating evidence. Also, fewer than half include an in-depth discussion of the biases that the work potentially suffers from. If the field of software ecosystems is to be matured, it is essential that more studies report on these desirable aspects in the future.

Finally, the extraordinary attributes are rarely observed. While not problematic, these are aspects that researchers in the future may have to include in future studies. The authors of this work, for instance, have little experience with pre-publishing a case protocol beforehand. While this practice may be common in other fields, we are still maturing. One thing the field can pride itself with: we often publish multiple case studies and perform cross case analysis. This may be a sign that the field is maturing, that case studies are ample available, and that we may wish to create better access to case materials, to encourage reuse and sharing of research assets.

4 Highlighting Four Studies from the Data Set

4.1 The Dynamics of Openness and the Role of User Communities A Case Study in the Ecosystem of Open-Source Gaming Handhelds

Using the open-source handheld games ecosystem as a case study, the paper reveals the influence of openness on games manufacturers' interactions with their user communities and customers' changes in preference. By using a longitudinal approach, the paper answered the two research questions: 1) how does the discrepancy between the openness provided by firms and the openness demanded by a user community emerge? And 2) what are the consequences for the ecosystem (especially the firms) when firms fail to address such a discrepancy? For the first question the authors posit that often time, discrepancies occurred when the level of openness provided by the firm fell below the expectation of their targeted users' community (a case of misaligned information releases). Position to influence a firm's openness levels users.

Regarding the second research question, the inability of a firm to effectively manage the users' demands for openness and emerging conflict can constitute a huge risk to the firm's survival, its competitiveness and may lose to an alternative user-initiated project springing up in the ecosystem thereby making openness a dimension of competition. A firm's degree of openness, therefore, relies on its strategy as well as extra-organizational actors (users and developers) [26].

Most firms are committed to achieving a symbiosis relationship with their users-developers community in their openness by benefiting from the valuable contributions offered by the external community contributors and reducing the possibility of incurring negative outcomes that may jeopardize their productivity [20]. The study shows how users' empowerment through the online community put them in a particularly good.

The paper proposed a framework of the dynamics of openness built on the repeated patterns illustrating the open-source game ecosystem. It highlighted the importance and

Table 1. Scoring of 51 studies using the ACM SIGSOFT Standards

Quality factor case studies	Out of 51
<i>Essential</i>	
Justifies the selection of the case(s) or site(s) that was(were) studied	46
Describes the site(s) in rich detail	42
Reports the type of case study	31
Describes data sources (e.g., participants' demographics and work roles)	38
Defines unit(s) of analysis or observation	43
Presents a clear chain of evidence from observations to findings	37
<i>Desirable</i>	
Provides supplemental materials such as interview guide(s), coding schemes, coding examples, decision rules, or extended chain-of-evidence tables	25
Triangulates across data sources, informants, or researchers	23
Cross-checks interviewee statements (e.g., against direct observation or archival records)	16
Uses participant observation (ethnography) or direct observation (non ethnography) and clearly integrates these observations into results	18
Validates results using member checking, dialogical interviewing, feedback from Non-participant practitioners or research audits of coding by advisors or other researchers	19
Describes external events and other factors that may have affected the case or site	25
Uses quotations to illustrate findings	19
EITHER: evaluates an a priori theory (or model, framework, taxonomy, etc.) using deductive coding with an a priori coding scheme based on the prior theory	25
OR: synthesizes results into a new, mature, fully developed and clearly articulated theory (or model, etc.) using some form of inductive coding (Coding scheme generated from data)	6
Researchers reflect on their own biases	18
<i>Extraordinary attributes</i>	
Multiple, deep, fully developed cases with cross-case triangulation	16
Uses a team-based approach, e.g., multiple raters with analysis of interrater reliability (see the IRR/IRA Supplement)	8
Published a case study protocol beforehand and made it publicly accessible (See the Registered Reports Supplement)	2

dynamism of openness as a dimension for competition. It also emphasizes the importance of user-driven innovation over the manufacturer driven product lifecycle. Three examples (Gamepark, GP Holdings, Project Ninja) of firms that failed to manage conflict arising from openness are presented and one example (Open Pandora) that was able to manage

their users' demand for openness and resulting conflict successfully. The Gamepark's failure is due to resentment from dissatisfied users' community members who felt their demands are being ignored while asserting that the Gamepark's success is mostly from the users' contribution. This resulting conflict is a huge blow to the company as the users' community eventually launched an alternative product which eventually dies due to the same reason. Decisions regarding a firm's openness strategy required adequate planning and effective communication with the targeted users' community as such decision is often difficult to reverse [20].

The authors used multiple data sources (forums, expert interviews, and secondary archival data) to validate and support their argument and used triangulation in the analysis. The proposed framework provides an abstract way of identifying and managing discrepancies that emerge between openness offered by firms and openness demanded by the user communities as well as the consequences arising thereafter. In addition, the author cites real-life cases of Handheld gaming firms to support their findings and by stating the research limitations, the authors provide useful direction for the future of the research work. All the data presented also meets the FAIR principles, respectively.

The major weakness of the paper is the ecosystem selection, which limited the choice to the handheld gaming industry ecosystem, thereby making it narrow and too specific. The paper also failed to consider the contribution of ordinary users (non-developer) during the semi structured interview process. Input from other related ecosystems as well as other users would have helped to strengthen the findings.

The longitudinal case study approach focuses on the open-source gaming handheld industry and helps to validate the evolution and the dynamics of firms, user communities, and their interactions. We find that a suboptimal level of openness can pose a threat to a firm's very existence.

4.2 How Do Software Ecosystems Evolve? A Quantitative Assessment of the R Ecosystem

Plakidas et al. [12] focus on the analysis of the emergence of software ecosystems by examining the structure and emergence of the R ecosystem as a case study of the open-source ecosystem. Package repository mining and statistical analysis method was used to extract R package data over a period of 12 months. The paper reveals metrics that help to identify and characterized a successful software ecosystem and compares it to approaches from related ecosystems literature by creating and comparing the prediction models based on package download frequency.

The software ecosystem consists of three main components namely: the software platform, the community of users, and the marketplace(s) respectively. The R ecosystem is classified into Platform Characteristics, Marketplace and Package Characteristics, and Community Characteristics. The paper was structured to provide answers to three research questions RQ1: How does the R software ecosystem evolve? The R ecosystem still enjoys robust growth in sizes and varieties since its inception RQ2: How do the community members collaborate, and how does this impact the software marketplace? The stakeholders play a significant role in boosting the marketplace by providing several packages that extend the functionality of the R core, with the active involvement of "insiders" as well as the single-package contributors. RQ3: What makes a software

ecosystem marketplace product successful? A strongly established community commitment and frequently maintained package contributed by experienced authors constituted a successful marketplace ecosystem [12].

The paper was well structured and clear answers were provided to the research questions. The paper employed a quantitative analysis of the R ecosystem to assess and quantify its emergence, and derive metrics on its core software components, the marketplace as well as its community, in addition to validating existing theories from the literature. The metrics and basic characteristics of the mined data are presented and supported with related analysis. Threats to validity are discussed from multiple perspectives and this help to strengthen the validity of the findings.’

One weakness identified from this research is the sourcing of data from Bioconductor (data downloaded were limited to the previous 12 months), which indicates incomplete versioning differentiation. A better approach would have been obtaining data from multiple repositories like Github or the package homepage.

4.3 Knowledge Boundaries in Enterprise Software Platform Development: Antecedents and Consequences for Platform Governance

The extant research on platforms has studied extensively the roles and actions of a platform leader and complementors, whether it is about governance or technological aspects. However, according to Foerderer et al. [6] “a comprehensive picture of knowledge management in platform ecosystems did not yet exist.” This is important, because complementors need knowledge about the platform functionality – what they can do with it to provide add-on services or products – and about the design of interfaces – how they can do it. A lot of the prior research has studied consumer-focused platforms, so the authors decided to focus on enterprise software, which is considered complex by nature. This complexity makes it non-trivial for the complementors to develop add-ons. The authors discuss knowledge management as one of the strategic activities of a platform owner, using and extending a knowledge boundary framework by Carlile [2]. The paper presents reasons for what hinders knowledge from passing the firm boundaries within a platform ecosystem, and what are its consequences.

The authors lay their foundations of the platform governance and knowledge management and show how they are related. From there they identify the research gap: what are the causes for knowledge gaps when boundaries are crossed and how the platform owners can try to manage the caps. Regarding the results, providing add-on services and products is about technological issues, but at the same they are not enough. The authors link technological properties of a platform – functional extent and interface design – with knowledge boundaries. The paper presents a classification of platform owners’ approaches to managing knowledge boundaries: broadcasting, brokering, and bridging. The research is a multiple case study with four platforms and interviewees from different industries, emphasis on the complementors. Empirical standard of ACM SIGSOFT for case study is matched well. Data triangulation is used: archival data complements the interviews. The analysis phase utilizes grounded theory with both a priori scheme and emerging codes. Inter-coder discussion and agreements was used. These tactics are applied in a rigorous fashion in the context of enterprise software platforms to show how knowledge boundaries come about and how they can be managed.

The multi-case approach and extensive data set provide a solid starting point. For some grounded theory researchers, the use of a priori coding scheme could be an issue, as grounded theory is about the emergence of concepts from the data. While the authors used a priori coding as a starting point based on the theoretical background and then allowed for the emergence of concepts related to knowledge boundary resources, it could be asked if something was missed or emphasized incorrectly due to it. Another question is related to the interviewees. The selection of CEOs and or CPOs was justified, but would for example a product manager point of view enriched the results? However, the interviewees had a chance to discuss and validate the research results, and according to the authors this strengthened the internal and external validity of the study.

4.4 Technology Ecosystem Governance

The case study by Wareham, Fox, and Cano Giner [22] identifies and describes three major tensions present in an enterprise software ecosystem: standard variety, control-autonomy, and collective-individual. These tensions can lead to exclusive either-or choices that actors must make, or they can manifest as complementary options. The authors have two research questions, first: “How are the main tensions in technology ecosystems addressed in technology ecosystem governance?” and second “Tensions can manifest themselves as either contradictory or complementary logics. Are contradictory and complementary logics present in technology ecosystems? If so, how are they governed?.” Based on the analysis of the tensions and their interactions the article then provides advice for the design of an ecosystem governance.

The article presents foundations of ecosystems around platforms: community of complementors, innovation, governance mechanisms, and generativity. It then focuses on the management of complementors (or third-party partners/providers) and the paradoxes between stability and evolvability that are present in the governance of modern ecosystem. Conceptual development in the article starts from tensions and how they can appear in three dimensions: outputs, actors, and identifications. They are studied in the context of ERP software, and the selection of the case is justified well. The selected ERP ecosystem provides a B2B setting with “severe heterogeneity across customers, complementors, and complements” – in other words, a real-life example. With their case study the authors show evidence for the presence of the tensions in the ecosystem, describe causes and effects, and finally present insights how to “harness tensions as enabling forces that serve the overall needs of the platform.”

Strengths of the article include detailed representation of the case study and its various attributes. Selection of the interviewees aims to handle the diversity present in the ecosystem. Grounded theory is used as a method for analyzing the data and a coding scheme is presented as well. Quotes from the interviews provide an exceptionally good understanding. Additional data sources are used to supplement the interview data. Finally, the results are summarized in a comprehensive implications section.

As the data set of the study is extensive and the analysis process was described as iterative, a more detailed description of the analysis would have been interesting to read. Cross validation or external factors that may have affected the data collection were not described in detail. Any bias of the authors was not discussed, although the Methods section mention that there were multiple validation incidents.

5 Recommendations for Empirical Researchers

Following the ongoing discussions, it is apparent that researchers in Software Ecosystems are adopting a variety of research methods, techniques, strategies, practices, and tools to increase the quality of their research. Despite widespread interest in empirical SECO studies [4], some researchers fail in presenting explicitly rigorous empirical evidence of their research goals, research questions, research methods, and validity defense, as we find in this preliminary study. Based on our findings, we make the following recommendations presented in Table 2.

Firstly, we noticed that in many articles the research goal and research questions are not explicitly stated but more generic, vague, and implicit. While some of these studies are well cited, it is unclear what their direct contribution is to the field and theory of software ecosystems. Similarly, scholars in [4] posit that, it is essential when selecting an appropriate research method to first clarify the research question. Therefore, as the first recommendation (R1), we recommend researchers to explicitly state the research objective and research question, possibly even highlighting them for increased readability.

Secondly and thirdly, it is striking to see how few articles are matching the essential and desirable criteria in the ACM SIGSOFT empirical standards. We therefore encourage researchers to ensure that their articles (R2) meet at least the essential the ACM SIGSOFT standards for empirical software engineering research and when research setting allows it, a study should also (R3) utilize the desirable standards. The ACM SIGSOFT standards offer guidelines to improve the quality of empirical studies in SECO.

Fourthly, to provide possibility for external validation and future works it necessary to get access to used data and extra materials, e.g., via appendix or link to external data storage. Therefore, we recommend researchers to provide data and extra materials, i.e., (R4) utilize FAIR data sharing principles [23] to make data findable, accessible, interoperable, and reusable.

Table 2. Recommendations for SECO studies.

ID	Recommendation	Reasoning
R1	Explicitly state the research question and research objective	Clarify and highlight the objective
R2	Utilize at least minimum essential attributes from ACM SIGSOFT standards	Improve and standardize the quality of empirical studies in SECO
R3	Utilize desirable attributes from ACM SIGSOFT standards when research setting allows it	Deepen the quality by verifying in more detail what and why is made
R4	Utilize FAIR data sharing principles	Enable external validation and future work by meeting principles of data findability, accessibility, interoperability, and reusability

Based on our observations in the studied articles, these recommendations can be applied, e.g., for case studies. Therefore, in case study once the research question is stated, attention should be paid to the case description (essential attribute): why it was selected and what makes it suitable for the study. When performing interviews, the triangulation of informants is an important phase (desirable attribute), i.e., selecting interviewees from different work roles. The studies we highlighted in Sect. 4 had a broad set of informants, intended to cover for the diversity. For example, interviewing only CEO level may introduce a bias. We encourage the researchers to utilizing multiple data sources (desirable attribute), as it is a feature that was present in many of the studies that matched ACM SIGSOFT criteria well. Moreover, although validity checking and validity discussion (desirable attributes) is by no means a novelty, it was surprising to find studies that only touched the subject lightly or even omitted it. To meet desirable attributes adding a section for validity discussion is advised.

6 Discussion

One could state that page limitations, for instance for conference publications, are a valid reason for not being able to cover all aspects required for excellent reporting of empirical research. We found it surprising to find few articles that refer to external data, such as case reports, software repositories, and data publications. We hope that researchers in the future start using options such as Mendeley Data, Arxiv, and other research repositories for storing their reusable data. This would be beneficial for the quality of publications, as they can focus on the essentials, but also for the reusability of research products.

An interesting question for discussion is the relative maturity of a research artifact. One of the criteria in the ACM SIGSOFT states that the work “synthesizes results into a new, mature, fully-developed and clearly articulated theory”, implying that work that does not do this is not up to par. Such questions touch all topics of science, such as the role of publication in the scientific process, the incrementality of the scientific process, and the quality and expectations of research outlets. We could even wonder if novelty, which is generally considered one of the main quality criteria for accepting papers, is an essential aspect of scientific progress. Obviously, we cannot answer these questions within this work, but we do recommend that authors attempt to add a significant increment of understanding in any contribution to the field of software ecosystems.

7 Conclusions and Future Work

In this study, we have explored how do software ecosystem researchers use empirical research methods, and which of the criteria of the ACM SIGSOFT empirical standards were met in case studies. For this, we conducted a literature study, with a pragmatic approach, to identify relevant literature in software ecosystem field, and it is likely that some relevant articles are missing.

We analyzed empirical software ecosystem papers from the years 2011–2021 and determined that case study is the most common research methods in empirical software ecosystem research followed by mixed methods, design science and survey. The collected articles were scored based on ACM SIGSOFT Case Study standards and we highlighted

the four highest rated articles as examples of good articles according to aforementioned standards.

The scoring of the articles revealed a vast variety in the research quality of the papers. Therefore, to help researchers to improve the quality of their software ecosystem research we provide four recommendations presented in Table 2.

In the future, we extend this study to include a more detailed description of the most prevalent research methods besides case studies. Furthermore, we plan to highlight more publications as exemplar for the field. As the mapping study is currently incomplete, we only inductively deduce from the gathered data.

It has been insightful to perform the analyses of the articles against the ACM SIG-SOFT standards; they provide a useful tool for assessing research quality and as a guideline for designing future studies. However, collecting data has been tedious. We also recommend that reviewers in the future perform similar assessments with each review and make this data publicly available. In this way, we can ensure that it becomes easier to perform systematic mapping studies in the future.

The field of software ecosystems has always had a strong empirical background, in part because of its origins in software engineering, but also because data is typically abundantly available. However, for all kinds of reasons, and as the data in this work shows, we observe that authors report their research results sloppily, insufficiently report about their methods, and do not share their data in any public manner. For the field to become more mature, this needs to significantly improve and this paper inherently holds a call to action for better research (reporting).

Acknowledgements. We thank Kari Smolander for organizing the session that sprouted this research.

References

1. Bosch, J.: From software product lines to software ecosystems. In: SPLC, vol. 9, pp. 111–119 (2009). <https://doi.org/10.1145/1753235.1753251>
2. Carlile, P.R.: Transferring, translating, and transforming: an integrative framework for managing knowledge across boundaries. *Organ. Sci.* **15**(5), 555–568 (2004). <https://doi.org/10.1287/ORS.1040.0094>
3. Decan, A., Mens, T.: What do package dependencies tell us about semantic versioning? *IEEE Trans. Softw. Eng.* **47**(6), 1226–1240 (2021). <https://doi.org/10.1109/TSE.2019.2918315>
4. Easterbrook, S., Singer, J., Storey, M.A., Damian, D.: Selecting empirical methods for software engineering research, pp. 285–311 (2008). https://doi.org/10.1007/978-1-84800-044-5_11
5. Engstrom, E., Storey, M.A., Runeson, P., Höst, M., Baldassarre, M.T.: How software engineering research aligns with design science: a review **25**, 2630–2660 (2020). <https://doi.org/10.48550/arXiv.1904.12742>
6. Russpatrick, S.: Understanding platform ecosystems for development: enabling innovation in digital global public goods software platforms. In: Bandi, R.K., Ranjini, C.R., Klein, S., Madon, S., Monteiro, E. (eds.) IFIPJWC 2020. IAICT, vol. 601, pp. 148–162. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-64697-4_12
7. Garcia-Holgado, A., Garcia-Penalvo, F.J.: Mapping the systematic literature studies about software ecosystems. In: TEEM 10-2018 (2018). <https://doi.org/10.1145/3284179.3284330>

8. Jacobides, M.G.: In the Ecosystem Economy, What's Your Strategy? September 2019. <https://hbr.org/2019/09/in-the-ecosystem-economy-whats-your-strategy>
9. Jansen, S., Cusumano, M.A.: Defining software ecosystem: a survey of software platforms and business network governance (2013). <https://doi.org/10.4337/9781781955628.00008>
10. Jansen, S., Cusumano, M.A., Brinkkemper, S.: Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry. Edward Elgar (2013). <https://doi.org/10.4337/9781781955628.00008>
11. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering **2**, 66 (2007). <https://doi.org/10.1016/j.infsof.2008.09.009>
12. Kuhn, T.S.: The Structure of Scientific Revolutions, vol. 111. University of Chicago Press, Chicago, January 1970
13. Manikas, K.: Software ecosystems - a systematic literature review. *J. Syst. Softw.* (2013). <https://doi.org/10.1145/3284179.3284330>
14. Manikas, K., Hansen, K.M.: Reviewing the health of software ecosystems—a conceptual framework proposal. In: Proceedings of the 5th International Workshop on Software Ecosystems (IWSECO), pp. 33–44. Citeseer (2013)
15. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: an update. *Inf. Softw. Technol.* **64**, 1–18 (2015). <https://doi.org/10.1016/j.infsof.2015.03.007>
16. Plakidas, K., Stevanetic, S., Schall, D., Ionescu, T.B., Zdun, U.: How do software ecosystems evolve? A quantitative assessment of the ecosystem. In: Proceedings of the 20th International Systems and Software Product Line Conference, pp. 89–98 (2016). <https://doi.org/10.1145/2934466.2934488>
17. Ralph, P., et al.: Empirical standards for software engineering research (2020). <https://arxiv.org/abs/2010.03525>. <https://doi.org/10.48550/arXiv.2010.03525>
18. Runeson, P., Host, M.: Guidelines for conducting and reporting case study research in software engineering. *Empir. Softw. Eng.* **14** (2009). Article Number: 131. <https://doi.org/10.1007/s10664-008-9102-8>
19. Tashakkori, A., Creswell, J.W.: The new era of mixed methods **01**, 3–7 (2007). <https://doi.org/10.1177/2345678906293042>
20. Teixeira, J., Robles, G., González-Barahona, J.M.: Lessons learned from applying social network analysis on an industrial Free/Libre/Open Source Software ecosystem. *J. Internet Serv. Appl.* **6**(1), 1–27 (2015). <https://doi.org/10.1186/s13174-015-0028-2>
21. Van Aken, J.E.: Management research as a design science: articulating the research products of mode 2 knowledge production in management **16**, 19–36 (2005). <https://doi.org/10.1111/j.1467-8551.2005.00437.x>
22. Wareham, J., Fox, P.B., Cano Giner, J.L.: Technology ecosystem governance. *Organiz. Sci.* **25**(4), 1195–1215 (2014). <https://doi.org/10.2139/ssrn.2201688>
23. Wilkinson, M.D., et al.: The fair guiding principles for scientific data management and stewardship. *Sci. Data* **3**(1), 1–9 (2016). <https://doi.org/10.1038/sdata.2016.18>
24. Wohlin, C., Mendes, E., Romero Felizardo, K., Kalinowski, M.: Guidelines for the search strategy to update systematic literature reviews in software engineering. *J. Syst. Softw.* (2020). <https://doi.org/10.1016/j.infsof.2020.106366>
25. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M.C., Regnell, B., Wesley, A.: Experimentation in Software Engineering. Springer, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-29044-2>
26. Zaggi, M.A., Schweisfurth, T.G., Herstatt, C.: The dynamics of openness and the role of user communities: a case study in the ecosystem of open-source gaming handhelds. *IEEE Trans. Eng. Manag.* **67**(3), 712–723 (2019). <https://doi.org/10.1109/TEM.2019.2897900>
27. Zelkowitz, M., Wallace, D.: Experimental models for validating technology **31**, 23–31 (1998). <https://doi.org/10.1109/2.675630>