Challenges in Software Ecosystems Research

Alexander Serebrenik
Eindhoven University of Technology
The Netherlands
a.serebrenik@tue.nl

Tom Mens
University of Mons
Belgium
tom.mens@umons.be

ABSTRACT

The paper is a meta-analysis of the research field of software ecosystems, by method of surveying 26 authors in the field. It presents a relevant list of literature and six themes in which challenges for software ecosystems can be grouped: Architecture and Design, Governance, Dynamics and Evolution, Data Analytics, Domain-Specific Ecosystems Solutions, and Ecosystems Analysis. As such, it provides a roadmap for future research in the field.

Categories and Subject Descriptors

K.6.3 [**Management of computing and information systems**]: Software Management—*software maintenance*; D.2.9 [**Software Engineering**]: Management

1. INTRODUCTION

Research on software ecosystems in a software engineering setting has been around for more than a decade, since the work by Messerschmitt and Szyperski in 2003 [46]. In this paper, we overview the state-of-the-art in software ecosystems research, and we shed some light on open challenges and future research.

The insights reported are based on an online survey that was conducted during two months in autumn 2014 with 26 respondents that have been active in software ecosystem research. As such, this survey complements the results of a systematic literature review, carried out by Manikas and Hansen [42], where 59 papers on the topic of software ecosystems were analysed.

The contributions of this paper are twofold. First, we identify open challenges in software ecosystem research. These challenges can be taken up by other researchers, in order to further advance the state-of-the-art and state-of-the-practice in this important field of research. Second, the overview of challenges in software ecosystems research serves as a roadmap for researchers new to the domain.

2. SURVEY

ECSAW '15, September 07 - 11, 2015, Dubrovnik/Cavtat, Croatia
© 2015 ACM. ISBN 978-1-4503-3393-1/15/09…$15.00
DOI: http://dx.doi.org/10.1145/2797433.2797475

Before starting the online survey, we identified 164 potential respondents based on whether they had co-authored a research article or book chapter related to software ecosystem, or an article that was presented in one of the annual workshops related to software ecosystems: the IWSECO series (International Workshop on Software Ecosystems), the WEA series (Workshop on Ecosystem Architectures) and BigSystem 2014 (ACM international workshop on Software-defined ecosystems). The survey was completely anonymous, but respondents were optionally given the possibility to leave an email address on which we could contact them for providing them with the results of our survey.

2.1 Questionnaire

The survey was composed of seven open-ended questions related to the definition of a software ecosystem, challenges and trends in the current research on software ecosystems, as well as pertaining to tools that support software ecosystem researchers and stakeholders:

- What constitutes a software ecosystem according to you and why?
- Provide a prototypical example of a software ecosystem according to the previous definition.
- What specific challenges need to by addressed by software ecosystem researchers and why?
- What are the most important recent trends in software ecosystem research?
- What tools, if any, do you use when studying software ecosystem and for which specific purpose?
- What tools for software ecosystem research are still missing and should be developed to advance the research field?
- What tools, if any, are needed by stakeholders involved in or relying on software ecosystems?

The survey started by asking the respondents what constitutes a software ecosystem and why, as well as to illustrate this definition by one or more examples. The results revealed that research on software ecosystems has been conducted from a broad spectrum of viewpoints, which is in accordance with the research literature; e.g., Lungu considers a software ecosystem as “a collection of software projects which are developed and evolve together in the same environment” [39], while Jansen et al. consider a software ecosystem to be “a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them” [34]. Therefore, in order to correctly interpret the respondents’ answers, we first needed to identify their stance on what constitutes an ecosystem.

Next, we asked the respondents to report recent trends in software ecosystem research and identify challenges that need to be addressed by software ecosystem researchers. Since all respondents were software ecosystem researchers themselves, they are best positioned to identify these challenges and trends.

Finally, since the importance of tools for software ecosystem governance and research has been widely recognised [16, 47] and numerous tools have been proposed [22, 40, 59], we asked the respondents about the tools they currently use as well as about those they believed to be missing still.

2.2 Survey results

Of the 164 potential candidates we originally identified, 23 did not have any valid email address. Of the remaining 141, 26 researchers responded to our survey by filling in the questionnaire, between 25 October 2014 and 17 December 2014. This corresponds to a response ratio of 18,4%.

For each of the seven questions in the survey, respondents were allowed to leave the response field blank if they felt they were not able to respond to a particular question. 9 out of 26 respondents answered all questions, 7 respondents answered all questions but one, 5 respondents provided answers to five questions, 4 to four questions, and one respondent answered only three questions.

20 out of 26 respondents indicated their interest in the survey results and provided their email addresses: 14 respondents were from Western and Northern Europe, five respondents from Latin America and one respondent from North America. Two respondents are currently employed by a company, while the remaining 18 respondents are researchers in academia.

We also observed differences in the number of responses obtained for different questions. All 26 respondents provided a definition and an example of an ecosystem, as well as identifying the challenges that need to by addressed by software ecosystem researchers. Only 21 respondents described the current trends and tools, 18 respondents suggested research tools that should be developed, and 11 respondents suggested future tools for stakeholders. These differences might be explained either by the respondents focusing on the first couple of questions and neglecting the remaining ones due to laziness or fatigue, or by the more challenging nature of the latter questions: describing the past might be an easier task then analysing the present, which is again easier than forecasting the future.

3. DEFINITION OF AN ECOSYSTEM

The metaphor of ecosystem has been used in a software setting occasionally in popular scientific literature since the late 1990s. Huberman and Hogg considered a distributed computing system of concurrent agents as a computational ecosystem, analogous to biological ecosystems [31]. They studied the dynamics and chaotic behavior of such computational systems and showed how reward mechanisms may stabilize the system, thereby optimizing its performance.

Scientific research on software ecosystems in the domain of software engineering only started to gain momentum since the 2003 work by Messerschmitt and Szyperski [46]. They defined a software ecosystem as “a collection of software products that have some given degree of symbiotic relationships.”

Since then, different definitions have been proposed, focusing on relations between projects comprising an ecosystem [39], presence of a shared architectural platform [12], or the business [34] and social context [48] where the ecosystems operate.

In their systematic literature survey, Manikas and Hansen attempted to combine the technical, architectural and business viewpoints above in a single encompassing definition [42]: “Software ecosystems are sets of software solutions functioning as a unit, enabling actors to automate activities and transactions.” More precisely, they conclude that a software ecosystem constitutes “the interactions of a set of actors on top of a common technological platform that results in a number of software solutions or services. Each actor is motivated by a set of interests or business models and connected to the rest of the actors and the ecosystem as a whole with symbiotic relationships, while, the technological platform is structured in a way that allows the involvement and contribution of the different actors.”

Regardless of which definition is adopted, we observed from the survey responses that the following aspects appear to be essential when identifying a software ecosystem:

The *social* aspect, requiring a community-centric way of collaborating and coordinating between the different actors. Depending on the ecosystem under consideration, these actors may include internal and external developers, but also domain experts and users. The actors may also be companies that produce, distribute, sell or buy software.

The *technical* aspect. Whatever the kind of software ecosystem that is considered, its constituent components should influence each other as the software ecosystem evolves. This is often achieved through some kind of technical *architecture* required to manage and control the ecosystem, and to create, share and evolve the software and services offered by the ecosystem.

The *economical* aspect, focusing on a software ecosystem from a business point of view in terms of supply and demand of software and services.

4. EXAMPLES OF SOFTWARE ECOSYSTEMS

Many examples of software ecosystems were provided by the survey respondents.

Different *companies* were cited explicitly as examples of being keystones of a business-oriented software ecosystem: *SalesForce* [30], *SAP* [63] and *FaceBook* [7].

Apple, *Google* and *Microsoft*, the three main companies involved in providing commercial operating systems for desktop PCs or smartphones were also cited by multiple respondents. The mobile ecosystems and app stores managed by these companies were frequently cited, and are popular ecosystems in the research literature [29, 32].

Programming language communities were frequently mentioned as being the core of dedicated software ecosystems. Explicitly cited languages were *Python* (including all its modules) [53], the *R* language and its *CRAN* archive network [27, 18, 69], the *Pharo* development platform for the *Smalltalk* programming language [56], and Microsoft's *.NET* platform (supporting a variety of programming languages such as *C#*). The *Eclipse* IDE, supporting mainly but not exclusively *Java* programming, and allowing third parties to contribute with different plugins was mentioned 10 times by survey respondents. This software ecosystem, supported by IBM, has been extensively studied [23, 43, 13, 14].

Software ecosystems focusing around operating systems are also a popular source of research. *iOS*, *Android*, and (distributions of) the open source *Linux* operating system were cited several times by the survey respondents [33, 55, 15, 70]. Another source of software ecosystems are those surrounding a particular open source software foundation. This was for example the case for *Apache* (cited 2 times) and *Mozilla* (mentioned once). Both Apache [49, 4] and Mozilla [36] are popular subjects of ecosystems research.

5. CHALLENGES

5.1 Quality and Design

The lion's share of the software ecosystem research so far has focussed on analysing quality and historical development. In agreement with this line of thought, several respondents noted the importance of understanding and analysing the quality of a software ecosystem. Quality can, however, be interpreted in many different ways. One of the respondents wonders *"How to create a lively community?"*, suggesting a social point of view, that requires a high-quality ecosystem to have an active community of developers and users. This point of view has recently been explored in several studies [44, 73, 69].

Another respondent indicated as challenge the ability *"to characterize the wealth of the community w.r.t. the wealth of the software components"* suggesting a socio-technical perspective aiming at understanding relations between developers and the software entities they create (cf. [6, 70]).

Yet another respondent focused on *"key architectural challenges such as: platform interface stability, security, reliability"*. This suggests a technical perspective, bringing software ecosystems closer to traditional software systems so that existing software quality measurement techniques may be applied, after adapting them to the specificities of software ecosystems.

A further group of challenges is related to transforming the insights obtained from studying the past in order to design new software ecosystems. In particular, respondents recognise the importance of software architecture in software ecosystem design and wonder *"what traits make a good architecture for building software ecosystems?"*. This challenge has been recognised in the past [21, 11, 3] but does not seem to have been addressed adequately. The software ecosystem research community is not unique in experiencing such difficulties in transitioning from analysis to design: similar difficulties have been reported in other areas of software engineering [35, 38].

5.2 Governance

Once the foundations of the software ecosystem have been laid, procedures and processes can be expected to emerge determining future evolution of the ecosystem. Those procedures and processes are related to the *management* and *governance* of software ecosystems [2], a challenge emphasised by multiple survey respondents: *"how to connect business goals and technical issues of ecosystems"* or *"significant challenges also remain with regard to software ecosystem governance, as the actions of keystones (e.g. platform owners, community managers) may well have far-reaching consequences for the ecosystem and its inhabitants."*

5.3 Dynamics and Evolution

An important challenge, raised by many survey respondents, relates to the dynamics and evolution management of software ecosystems. Indeed, the software evolution community has seen a gradual shift from studies of individual software systems to macro-level studies of software ecosystems [28, 45, 15].

Similar to the discussion of quality and design (Section 5.1), some respondents aim at understanding the past: *"The main [challenge], from my point of view, is to understand the dynamics and evolution of the ecosystem."*

Others focus at implementing the future steps: *"How to evolve the ecosystem. Evolution and compatibility aspects in ecosystem are key aspects for the success. If a ecosystem is not able to evolve quickly it is going to die."*

Some respondents elaborate on specific evolutionary challenges and relate software evolution to dependencies between software components and the need to understand how changes in one of the components affect other components, i.e., the well-known problem of change impact analysis [9].

One respondent stressed the importance of domain knowledge for performing change impact analysis (cf. [1]): *"Provide clear upgrade paths. What happens when Android's API changes? How should the other entities in the ecosystem change, what is the time-frame."*

5.4 Data analytics

Scale. Massive amounts of data are produced when software ecosystems are being created, maintained and used. For example, GitHub counts over 20 million repositories, and Stack Overflow has more than 20 million posts. The challenge of analysing data at this scale has been recognised both in the literature [51] and by the survey respondents: *"Software ecosystems may consist of many systems. Analysing all these systems as a whole may raise some technical problems, due to the quantity of data to take into account."*

Scale is not the only challenge associated with the data derived from software ecosystems. In fact, many of the challenges associated with the big data [20] have been recognized by the survey respondents.

Variety, i.e., different forms of data, also known as heterogeneity, can be witnessed by combination of structured, semi-structured and unstructured data. Moreover, recent studies explore non-textual data, e.g., derived from biometrics [26] or video recordings [62].

Veracity relates to the uncertainty and inconsistency of data. Data may also be incomplete, e.g., due to its unavailability or inaccessibility: *"non-open source, data not captured in repositories (e.g., business data) etc."* Obtaining data necessary to study closed source software ecosystems remains challenging.

Furthermore, data might appear incomplete due to peculiarities of the software engineering process followed. For instance, history as recorded in the Git version control system can be revised, i.e., some of the information might be erased later [8]. The problem of inconsistent data has been extensively discussed in the database community [41]. Kouters et al. [37] have discussed different kinds of inconsistencies found in contributors' aliases in GNOME.

Velocity, i.e., speed of streaming data becomes a challenge when e.g., analyzing commits on GitHub with modifications being committed every second[1].

Privacy. Software engineering data often contains information about individuals involved in software development and maintenance. It is therefore imperative to preserve privacy of those individuals. It has been shown that repository data can be used to infer gender [66, 67] and personality traits [5] of software developers. Privacy issues raised for digital trace data might be even more severe if survey data is concerned as surveys allow one to discover beliefs and perceptions: popularity of surveys in software ecosystem research [58, 68] make maintaining privacy to a major concern. Privacy, however, is conflicting with the growing call for sharing the software ecosystem data following the recognition of importance of openness of the data [52] and of reproducible research in software engineering [61, 57].

5.5 Domain-Specific Ecosystem Solutions

While the concerns discussed in the preceding subsections are applicable to software ecosystem of different kinds, one of the respondents stressed the importance of distinguishing between software ecosystems from different domains: *"Descriptive research, followed by explanatory research, should be the basis for identifying the challenges in the different domains. Once this is done, specific solutions can be developed. An important topic, which is to my opinion insufficiently covered, is what the consequences are for the development process and the software architectures."*

Indeed, popularity of domain-specific solutions such as domain-specific languages [65] and domain-specific architectures [19] suggests importance of domain-based solutions for software ecosystem as well. Bosch has recognised domain-specificity of application-centric software ecosystem such as Microsoft's Office suite [10]. However, one might wonder whether ecosystems developed in the same domain, such as GNOME and KDE, or Python and Ruby, exhibit more similarity in the way they are organised, governed or evolve than ecosystems from different domains. First steps towards understanding specificities of software ecosystems within a given domain have been made for embedded software [25] and the telecom industry [71].

5.6 Analysis of Ecosystems

Statistics. Analysis of software ecosystems data requires advanced statistical techniques, specifically addressing challenges related to aggregation of data [50] and evolution of software ecosystems [17, 60].

Visualization. Visualization techniques form an alternative to statistical analysis. Indeed, one respondent expressed the need to have *"standard ways of visualising ecosystems"* and numerous visualization approaches have been proposed by software ecosystem researchers [54, 72].

Comparisons. Finally, more attention should be dedicated to comparative studies: *"doing comparisons across software ecosystems to understand differences is wide open"* (cf. [24, 23, 64]). Comparative studies can be seen as a prerequisite for designing successful domain-specific ecosystem solutions (cf. Section 5.5).

[1]https://github.com/blog/620-committing-like-crazy

6. CONCLUSIONS

We identified the current challenges and future trends in software ecosystem research, based on an online survey carried out with 26 researchers, that was complemented with our own vision on the research field. We hope that many of the identified challenges in our roadmap will be taken up by other researchers, in order to further advance the state-of-the-art and state-of-the-practice in this important field of research.

7. ACKNOWLEDGMENTS

We are grateful to the survey respondents for sharing their knowledge and insights on different aspects of software ecosystems. This research was carried out in the context of ARC research project AUWB-12/17-UMONS- 3 and NWO project 600.065.120.10N235.

8. REFERENCES

[1] A. Aryani, F. Perin, M. Lungu, A. N. Mahmood, and O. Nierstrasz. Predicting dependences using domain-based coupling. *J. Software Evolution and Process*, 26(1):50–76, 2014.

[2] A. Baars and S. Jansen. A framework for software ecosystem governance. In *Software Business*, volume 114 of *Lecture Notes in Business Information Processing*, pages 168–180. Springer, 2012.

[3] O. Barbosa and C. Alves. A systematic mapping study on software ecosystems. In *Int'l Workshop on Software Ecosystems (IWSECO)*, pages 15–26. CEUR Workshop Proceedings, 2011.

[4] G. Bavota, G. Canfora, M. Di Penta, R. Oliveto, and S. Panichella. The evolution of project inter-dependencies in a software ecosystem: the case of Apache. In *Int'l Conf. Software Maintenance*, 2013.

[5] B. Bazelli, A. Hindle, and E. Stroulia. On the personality traits of StackOverflow users. In *Int'l Conf. Software Maintenance*. IEEE, 2013.

[6] A. Begel, K. Y. Phang, and T. Zimmermann. Codebook: Discovering and exploiting relationships in software repositories. In *Int'l Conf. Software Engineering*, pages 125–134, 2010.

[7] A. Bessi, A. Scala, L. Rossi, Q. Zhang, and W. Quattrociocchi. The economy of attention in the age of (mis)information. *Journal of Trust Management*, 1(1):12, 2014.

[8] C. Bird, P. C. Rigby, E. T. Barr, D. J. Hamilton, D. M. Germán, and P. T. Devanbu. The promises and perils of mining Git. In *Working Conf. Mining Software Repositories (MSR)*, pages 1–10, 2009.

[9] S. A. Bohner and R. S. Arnold. *Software Change Impact Analysis*. IEEE Computer Society, 1996.

[10] J. Bosch. From software product lines to software ecosystems. In *Int. Software Product Line Conf.*, pages 111–119, 2009.

[11] J. Bosch. Architecture challenges for software ecosystems. In *European Conf. Software Architecture*, pages 93–95. ACM, 2010.

[12] J. Bosch and P. Bosch-Sijtsema. From integration to composition: on the impact of software product lines, global development and ecosystems. In *Int'l Conf. Software Product Lines*. Springer, 2009.

[13] J. Businge, A. Serebrenik, and M. G. J. van den Brand. Survival of Eclipse third-party plug-ins. In *Int'l Conf. Software Maintenance*, pages 368–377, 2012.

[14] J. Businge, A. Serebrenik, and M. G. J. van den Brand. Analyzing the Eclipse API usage: Putting the developer in the loop. In *European Conf. Software Maintenance and Reengineering*, pages 37–46. IEEE Computer Society, 2013.

[15] M. Caneill and S. Zacchiroli. Debsources: Live and historical views on macro-level software evolution. In *Int'l Symp. Empirical Software Engineering and Measurement*, pages 1–10. ACM, 2014.

[16] M. Cataldo and J. D. Herbsleb. Architecting in software ecosystems: Interface translucence as an enabler for scalable collaboration. In *European Conf. Software Architecture*, pages 65–72. ACM , 2010.

[17] M. Claes, T. Mens, R. Di Cosmo, and J. Vouillon. A historical analysis of Debian package incompatibilities. In *Working Conf. Mining Software Repositories (MSR)*, 2015.

[18] M. Claes, T. Mens, and P. Grosjean. On the maintainability of CRAN packages. In *Int'l Conf. on Software Maintenance, Reengineering, and Reverse Engineering (CSMR-WCRE)*, pages 308–312, 2014.

[19] E. de Almeida, A. Alvaro, V. Garcia, L. Nascimento, S. Meira, and D. Lucredio. Designing domain-specific software architecture (DSSA): Towards a new approach. In *Working Conf. Software Architecture*, page 30, Jan 2007.

[20] Y. Demchenko, P. Grosso, C. De Laat, and P. Membrey. Addressing big data issues in scientific data infrastructure. In *Collaboration Technologies and Systems*, pages 48–55, May 2013.

[21] D. Dreyfus and B. Iyer. Managing architectural emergence: A conceptual model and simulation. *Decision Support Systems*, 46(1):115–127, 2008.

[22] S. Ducasse, M. Lanza, and S. Tichelaar. Moose: an extensible language-independent environment for reengineering object-oriented systems. In *Int'l Symp. Constructing Software Engineering Tools (CoSET)*, June 2000.

[23] J. Duenas, H. Parada G., F. Cuadrado, M. Santillan, and J. Ruiz. Apache and Eclipse: Comparing open source project incubators. *IEEE Software*, 24(6):90–98, 2007.

[24] N. Economides and E. Katsamakas. Linux vs. Windows: A comparison of application and platform innovation incentives for open source and proprietary software platforms. Working Papers 05-07, NET Institute, 2005.

[25] U. Eklund and J. Bosch. Architecture for embedded open software ecosystems. *J. Systems and Software*, 92(0):128–142, 2014.

[26] T. Fritz, A. Begel, S. C. Müller, S. Yigit-Elliott, and M. Züger. Using psycho-physiological measures to assess task difficulty in software development. In *Int'l Conf. Software Engineering*, pages 402–413, 2014.

[27] D. M. Germán, B. Adams, and A. E. Hassan. The evolution of the R software ecosystem. In *European Conf. Software Maintenance and Reengineering*, pages 243–252, 2013.

[28] J. M. Gonzalez-Barahona, G. Robles, M. Michlmayr, J. J. Amor, and D. M. German. Macro-level software evolution: a case study of a large software compilation. *Empirical Software Engineering*, 14(3):262–285, Mar. 2009.

[29] M. Harman, Y. Jia, and Y. Zhang. App store mining and analysis: MSR for app stores. In *Working Conf. Mining Software Repositories (MSR)*, pages 108–111, June 2012.

[30] D. Hilkert, C. M. Wolf, A. Benlian, and T. Hess. The as-a-service paradigm and its implications for the software industry: Insights from a comparative case study in CRM software ecosystems. In *Software Business*, volume 51 of *Lecture Notes in Business Information Processing*, pages 125–137. Springer, 2010.

[31] B. A. Huberman and T. Hogg. The emergence of computational ecologies. In *Lectures in Complex Systems*, pages 185–205. Addison-Wesley, 1993.

[32] S. Hyrynsalmi, T. Mäkilä, A. Järvi, A. Suominen, M. Seppänen, and T. Knuutila. App Store, Marketplace, Play! An analysis of multi-homing in mobile software ecosystems. In *Int'l Workshop on Software Ecosystems (IWSECO)*, volume 879 of *CEUR Workshop Proceedings*, pages 59–72, 2012.

[33] A. Israeli and D. G. Feitelson. The Linux kernel as a case study in software evolution. *J. Systems and Software*, 83(3):485–501, 2010.

[34] S. Jansen, A. Finkelstein, and S. Brinkkemper. A sense of community: A research agenda for software ecosystems. In *Int'l Conf. Software Engineering*, pages 187–190, May 2009.

[35] H. Kaindl. Difficulties in the transition from OO analysis to design. *IEEE Softw.*, 16(5):94–102, Sept. 1999.

[36] F. Khomh, T. Dhaliwal, Y. Zou, and B. Adams. Do faster releases improve software quality? An empirical case study of Mozilla Firefox. In *Working Conf. Mining Software Repositories (MSR)*, pages 179–188. IEEE, 2012.

[37] E. Kouters, B. Vasilescu, A. Serebrenik, and M. G. J. van den Brand. Who's who in Gnome: using LSA to merge software repository identities. In *Int'l Conf. Software Maintenance*, pages 592–595. IEEE, 2012.

[38] B. H. Krogh. From analysis to design. In *Formal Modeling and Analysis of Timed Systems*, volume 4763 of *Lecture Notes in Computer Science*, pages 4–4. Springer, 2007.

[39] M. Lungu. Towards reverse engineering software ecosystems. In *Int'l Conf. Software Maintenance*, pages 428–431, 2008.

[40] M. Lungu and M. Lanza. The small project observatory: a tool for reverse engineering software ecosystems. In *Int'l Conf. Software Engineering*, pages 289–292, 2010.

[41] J. I. Maletic and A. Marcus. Data cleansing: Beyond integrity analysis. In B. D. Klein and D. F. Rossin, editors, *Fifth Conference on Information Quality (IQ 2000)*, pages 200–209. MIT, 2000.

[42] K. Manikas and K. M. Hansen. Software

ecosystems—A systematic literature review. *J. Systems and Software*, 86(5):1294–1306, 2013.

[43] T. Mens, J. Fernández-Ramil, and S. Degrandsart. The evolution of Eclipse. In *Int'l Conf. Software Maintenance*, pages 386–395, 2008.

[44] T. Mens and M. Goeminne. Analysing the evolution of social aspects of open source software ecosystems. In *Int'l Workshop on Software Ecosystems (IWSECO)*, pages 1–14. CEUR Workshop Proccedings, June 2011.

[45] T. Mens, A. Serebrenik, and A. Cleve. *Evolving Software Systems*. Springer, 2014.

[46] D. Messerschmitt and C. Szyperski. *Software ecosystem: Understanding and indispensable technology and industry*. MIT Press, 2003.

[47] I. Mistrîk, J. Grundy, A. Hoek, and J. Whitehead. Collaborative software engineering: Challenges and prospects. In *Collaborative Software Engineering*, pages 389–403. Springer, 2010.

[48] E. Mitleton-Kelly. *Ten Principles of Complexity and Enabling Infrastructures*, pages 23–50. Pergamon, 2003.

[49] A. Mockus, R. Fielding, and J. Herbsleb. Two case studies of open source software development: Apache and Mozilla. *ACM Trans. Software Engineering and Methodology*, 11(3):309–346, 2002.

[50] K. Mordal, N. Anquetil, J. Laval, A. Serebrenik, B. Vasilescu, and S. Ducasse. Software quality metrics aggregation in industry. *J. Software Evolution and Process*, 25(10):1117–1135, 2013.

[51] O. Nierstrasz and M. Lungu. Agile software assessment (invited paper). In *Int'l Conf. Program Comprehension*, pages 3–10, 2012.

[52] F. S. Parreiras, G. Gröner, D. Schwabe, and F. de Freitas Silva. Towards a marketplace of open source software data. In *Hawaii Int'l Conf. System Sciences (HICSS)*, pages 3651–3660. IEEE, 2015.

[53] F. Pérez, B. E. Granger, and J. D. Hunter. Python: An ecosystem for scientific computing. *Computing in Science & Engineering*, 13(2):13–21, 2011.

[54] J. Pérez, R. Deshayes, M. Goeminne, and T. Mens. SECONDA: software ecosystem analysis dashboard. In *CSMR*, pages 527–530, 2012.

[55] E. S. Raymond. *The cathedral and the bazaar: musings on Linux and open source by an accidental revolutionary*. O'Reilly & Associates, Inc., 2001.

[56] R. Robbes, M. Lungu, and D. Röthlisberger. How do developers react to API deprecation? the case of a Smalltalk ecosystem. In *Int'l Symp. Foundations of Software Engineering*, pages 56:1–56:11. ACM , 2012.

[57] G. Robles. Replicating MSR: A study of the potential replicability of papers published in the mining software repositories proceedings. In *Working Conf. Mining Software Repositories (MSR)*, pages 171–180. IEEE, 2010.

[58] G. Robles, L. Arjona Reina, A. Serebrenik, B. Vasilescu, and J. M. González-Barahona. FLOSS 2013: a survey dataset about free software contributors: challenges for curating, sharing, and combining. In *Working Conf. Mining Software Repositories (MSR)*, pages 396–399. ACM, 2014.

[59] G. Robles, J. Gonzalez-Barahona, D. Izquierdo-Cortazar, and I. Herraiz. Tools for the study of the usual data sources found in libre software projects. *Int'l Journal of Open Source Software and Processes*, 1(1):24–45, 2009.

[60] I. Samoladas, L. Angelis, and I. Stamelos. Survival analysis on the duration of open source projects. *Information & Software Technology*, 52(9):902–922, 2010.

[61] F. Shull, J. C. Carver, S. Vegas, and N. J. Juzgado. The role of replications in empirical software engineering. *Int'l Conf. Empirical Software Engineering*, 13(2):211–218, 2008.

[62] D. Socha and J. D. Tenenberg. Sketching software in the wild. In *Int'l Conf. Software Engineering*, pages 1237–1240, 2013.

[63] G. Timbrell and G. Gable. The SAP ecosystem: A knowledge perspective. In F. F.-H. Nah, editor, *Enterprise Resource Planning: Solutions and Management*, pages 209–220. IGI Global, 2002.

[64] J. van Angeren, V. Blijleven, and S. Jansen. Relationship intimacy in software ecosystems: a survey of the dutch software industry. In *Int'l Conf. Management of Emergent Digital EcoSystems (MEDES)*, pages 68–75, 2011.

[65] A. van Deursen, P. Klint, and J. Visser. Domain-specific languages: An annotated bibliography. *SIGPLAN Not.*, 35(6):26–36, June 2000.

[66] B. Vasilescu, A. Capiluppi, and A. Serebrenik. Gender, representation and online participation: A quantitative study of StackOverflow. In *SocialInformatics*, pages 332–338, 2012.

[67] B. Vasilescu, A. Capiluppi, and A. Serebrenik. Gender, representation and online participation: A quantitative study. *Interacting with Computers*, 26(5):488–511, 2014.

[68] B. Vasilescu, D. Posnett, B. Ray, M. G. J. van den Brand, A. Serebrenik, P. T. Devanbu, and V. Filkov. Gender and tenure diversity in GitHub teams. In *Conference on Human Factors in Computing Systems (CHI)*, pages 3789–3798. ACM, 2015.

[69] B. Vasilescu, A. Serebrenik, P. T. Devanbu, and V. Filkov. How social Q&A sites are changing knowledge sharing in open source software communities. In *Computer Supported Cooperative Work (CSCW)*, pages 342–354, 2014.

[70] B. Vasilescu, A. Serebrenik, M. Goeminne, and T. Mens. On the variation and specialisation of workload: A case study of the GNOME ecosystem community. *J. Empirical Software Engineering*, 19(4):955–1008, 2014.

[71] M. Viljainen and M. Kauppinen. Software ecosystems: A set of management practices for platform integrators in the telecom industry. In *Software Business*, volume 80 of *Lecture Notes in Business Information Processing*, pages 32–43. Springer, 2011.

[72] R. Wettel and M. Lanza. CodeCity: 3D visualization of large-scale software. In *Int'l Conf. Software Engineering*, pages 921–922, 2008.

[73] Q. Xuan, M. Gharehyazie, P. T. Devanbu, and V. Filkov. Measuring the effect of social communications on individual working rhythms: A case study of open source software. In *SocialInformatics*, pages 78–85, 2012.