

Report from Dagstuhl Seminar 18182

Software Business, Platforms, and Ecosystems: Fundamentals of Software Production Research

Edited by

**Pekka Abrahamsson¹, Jan Bosch², Sjaak Brinkkemper³, and
Alexander Mädche⁴**

1 University of Jyväskylä, FI, pekka.abrahamsson@jyu.fi

2 Chalmers University of Technology – Göteborg, SE, jan@janbosch.com

3 Utrecht University, NL, s.brinkkemper@uu.nl

4 KIT – Karlsruher Institut für Technologie, DE, alexander.maedche@kit.edu

Abstract

This report documents the program and the outcomes of Dagstuhl Seminar 18182 “Software Business, Platforms, and Ecosystems: Fundamentals of Software Production Research”.

Seminar April 29 – May 2, 2018 – <http://www.dagstuhl.de/18182>

2012 ACM Subject Classification Software and its engineering → Software organization and properties; Software and its engineering → Software creation and management; Software and its engineering → Software creation and management

Keywords and phrases Software Engineering, Software Ecosystems, Software-intensive Business, Software Production, Software Startups


Digital Object Identifier 10.4230/DagRep.8.4.164

Edited in cooperation with Slinger Jansen and Karl Werder

1 Executive Summary

Slinger Jansen (Utrecht University, the Netherlands, slinger.jansen@uu.nl)

Karl Werder¹ (Universität Duisburg – Essen, Germany)

License  Creative Commons BY 3.0 Unported license
© Slinger Jansen and Karl Werder

Software producing organizations (SPOs) face challenges every day. Whether they are open source consortia or commercial software product companies, they all face the challenges of changing demands, rapidly evolving technology, and a dynamic ecosystem in which their products and services need to operate. SPOs need to rethink their operating models and benefit from current and future trends. E.g. agile software development and DevOps allow them to respond swiftly to changes in their environment, embracing uncertainty. Particularly in conjunction with machine learning and artificial intelligence, SPOs can generate strategic competitive advantages. Particularly companies with a long history in a given domain, such as SAP and Volkswagen, seem to be too comfortable with their status quo. Meanwhile, smaller companies drive innovation on many fronts. Examples are Provenance that benefits from blockchain technology to revolutionize trust in goods, or Tesla and Local Motors that push autonomous cars into consumer markets.

¹ Now at University of Cologne, Germany, werder@wiso.uni-koeln.de



Except where otherwise noted, content of this report is licensed under a Creative Commons BY 3.0 Unported license

Software Business, Platforms, and Ecosystems: Fundamentals of Software Production Research, *Dagstuhl Reports*, Vol. 8, Issue 04, pp. 164–198

Editors: Pekka Abrahamsson, Jan Bosch, Sjaak Brinkkemper, and Alexander Mädche



DAGSTUHL
REPORTS Dagstuhl Reports

Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

The challenge to make these organizations successful is multi-disciplinary. First, there exist technology challenges, such as eliciting and prioritizing requirements, dealing with platforms and technology standards, and operating in complex technology landscapes that constrain and enable their technology. Secondly, there exist adoption challenges: organizations need to find ways to convince their target users to adopt their technologies and to coordinate evolving technologies to provide the most valuable end-user experience. Thirdly, there exist business model challenges, where these organizations must find ways to maximize profit from their innovations and technologies. Because of the pervasiveness of software, the challenges are observed everywhere in the economy, whether it is logistics, online marketing, or e-health. Furthermore, they are applicable to organizations in every stage of development, whether it is a software startup or a software giant that has influenced the market consistently for decades.

Hence, this Dagstuhl Seminar invited thought leaders from academia and industry to share their knowledge and experiences. Participants were asked to share a short position statement of max 300 words and participate in the development of a groundbreaking research agenda. These efforts aimed to increase visibility and impact of software production research and to set a course for the next decades. In addition, the seminar helped bringing together scholars and industry practitioners from different communities, such as product management, technology management, information systems, software engineering, and human-computer interaction in order to sharpen and define the joint community of Software-intensive Business (see Section 4.3.1).

A central outcome of the seminar was the agreement to use the term Software-intensive Business in order to describe the joint community with members of great diversity. Furthermore, the seminar focused on

- defining core concepts and identifying a roadmap
- Software-intensive Business and technology artifacts
- research needs in continuous experimentation & innovation
- lifecycle and research of software ecosystems
- research data for Software-intensive Businesses

As a major result from the seminar, the following achievements have been identified:

1. research a clear agenda for the field of Software-intensive Business research
2. carving out trends and research challenges in further depth
3. forming groups for continuous collaborations on different elements of the research agenda
4. organize bi-weekly meetings on-line for community building and research sharing.

2 Contents

Executive Summary

<i>Slinger Jansen and Karl Werder</i>	164
---	-----

Position Statements

Understanding Software Ecosystems through Visual Analytics and Machine Learning <i>Rahul C. Basole</i>	168
Transforming To A Software Business <i>Jan Bosch</i>	169
Innovation + Velocity + Pivoting in Software Production: The New Normal <i>Christoph Bussler</i>	170
Platform versus Non-Platform Company Performance: Some Exploratory Data Analysis, 1995-2015 <i>Michael A. Cusumano</i>	171
Platform Elasticity for Fast Time-to-Critical Mass and Take-Off <i>Samuel Fricker</i>	171
Research statement <i>Jens Foerderer</i>	172
Feature-Oriented Development in Industrial Automation Ecosystems <i>Paul Grünbacher</i>	173
Analyzing the Mutual Quality Impact of Business Processes and Information Systems <i>Robert Heinrich</i>	174
Research Statement <i>Armin Heinzl</i>	174
Software Engineering evolution <i>Mika Helenius</i>	175
Position Statement <i>Georg Herzwurm</i>	176
From Managing Your Ecosystems to Repositioning Your Business <i>Helena Holmström Olsson</i>	176
Some Research Directions for Software Ecosystems & Platforms <i>Sami Hyrynsalmi</i>	177
Engineering FLOSS Ecosystems for Impact and Sustainability <i>Zhi Jin</i>	178
Academic Structures and Terminology <i>Hans-Bernd Kittlaus</i>	179
Position Statement <i>Thomas Kude</i>	180
Position Statement <i>Alexander Mädche</i>	180

Challenges in Software Ecosystems and Product Development <i>Efi Papatheocharous</i> ,	181
Elements of Platform-based Ambidexterity: An Empirical Investigation <i>Balasubramaniam Ramesh</i>	182
Minimum Viable Products – The Road Ahead <i>Guenther Ruhe</i> ,	183
Only few can be platform owners <i>Kari Smolander</i>	183
Data for Performing Research on Software Business, Platforms, and Ecosystems <i>Diomidis Spinellis</i>	184
Software Ecosystems Research Agenda Update <i>Slinger Jansen</i>	185
AI and Software Business <i>Pasi Tyrväinen</i>	186
High-speed and Sustainable Development of Software Startups <i>Xiaofeng Wang</i>	187
Understanding new development trends, exploring large data, and pushing the boundaries of innovation <i>Karl Werder</i>	188
Improving handling business model changes for software-intensive organizations <i>Krzysztof Wnuk</i>	189
Working Groups	
Working Group on the Software Ecosystem Research Agenda <i>Paul Grünbacher, Jens Förderer, Zhi Jin, Samuel Fricker, Rahul Basole, Slinger Jansen</i>	190
Working Group on Software Business and Technology Lifecycle Artifacts <i>Sjaak Brinkkemper, Armin Heinzl, Robert Heinrich, Alexander Maedche, Kari Smolander</i>	192
Working Group on Software-Intensive Business Research: Definition and Roadmap <i>Karl Werder, Sjaak Brinkkemper, Jan Bosch, Michael A. Cusumano, Georg Herzwurm, Helena Holmström Olsson, Sami Hyrynsalmi, Hans-Bernd Kittlaus, Thomas Kude, Tiziana Margaria, Efi Papatheocharous, Diomidis Spinellis, Pasi Tyrväinen, Xiaofeng Wang</i>	193
Working Group on Health Measurement of Open Source Projects and Ecosystems <i>Slinger Jansen, Paul Grunbacher, Efi Papatheocharous, Diomidis Spinellis</i>	196
Working Group on Research Data for Software Intensive Business <i>Slinger Jansen, Diomidis Spinellis, Krzysztof Wnuk</i>	197
Participants	198

3 Position Statements

3.1 Understanding Software Ecosystems through Visual Analytics and Machine Learning

Rahul C. Basole (Georgia Institute of Technology – USA, basole@gatech.edu)

License  Creative Commons BY 3.0 Unported license
© Rahul C. Basole

The software industry is fiercely competitive and highly dynamic with companies of all sizes and geographic location battling for market share and new entrants emerging constantly. To survive in this hypercompetitive environment, companies must continuously innovate, not just in terms of software functionalities but also in ways they are designed, developed, offered and licensed. Ecosystemic thinking is thus critical.

My research examines the complexity of different types of software ecosystems from micro- to macro-level perspectives using emerging computational and visual analytic approaches. My core argument is that the scale, scope, and evolving dynamics of software ecosystems demand novel data-driven research methods and that we can support our understanding and augment decision making through interactive visual analytic approaches.

Some of my recent and ongoing studies include the examination of API and SDK ecosystems, digital platforms, digital infrastructures, dynamics of developer ecosystems, software alternatives, microservices, and global software startup ecosystems. Our investigations are enabled and driven by large-scale, heterogeneous (structured and unstructured) publicly available and proprietary data. Since the goal of my research is to create actionable insights, and not just archival knowledge, my lab develops interactive, visual, human-centered tools (e.g., *ecoxight*, *graphiti*, *epheno*, *pulse*, etc.) that enable exploration, discovery, and sense-making of the structure and dynamics of such software ecosystems. A set of sample (static) visualizations at different software ecosystem levels is shown below.

There are many exciting open research opportunities in the study of software businesses, platforms, and ecosystems using visual analytics and machine learning that would be worthy of further discussion.

- What are evolutionary patterns of software startups, platforms, and ecosystem and how do they relate to success and failure? Are there segmental or geographic differences?
- How do developer ecosystems react and organize to software launches and changes?
- How do APIs and SDKs complement, enhance, or constrain interfirm relationships?
- How can software platforms orchestrate complex evolving ecosystems and shield against disruption? What role do developers play?
- How do firms adopt, experiment, and discard digital infrastructure technologies?
- How can you anticipate, prepare, and adapt to changes in software ecosystems?

3.2 Transforming To A Software Business

Jan Bosch (Chalmers University of Technology – Sweden, jan@janbosch.com)

License  Creative Commons BY 3.0 Unported license
© Jan Bosch

Digitalization is concerned with creating new revenue and value producing opportunities through the use of digital technologies. In practice, this typically refers to increased product value by adding software, licensing software as a standalone product or services using the data generated by users and products.

Although the literature is filled with examples of companies that were born digital, such as Google, Uber, Booking and AirBnB, the fact of the matter is that there are thousands of companies that need to transform their business model and product portfolio in response to the emergence of digital technologies and the digitalization trend. The data shows that these companies are not very successful at this. For instance, the duration of companies on the Fortune 500 has now shortened to 10 years and digitalization is the predominant cause of disruption for the companies that have disappeared from the list.

Our experience in Software Center (www.software-center.se) shows that there are several challenges that companies need to contend with:

- Top leadership lacks knowledge and has a quarterly results focus: most top leaders have established their careers in non-digital technologies and have, for decades, been trained to focus on delivering on the quarterly results.
- The ecosystem is holding back companies: Even if the company sees the need and wants to change, its customers and partners often are unwilling to change with the enlightened company. And the company can't implement the change without alienating its customers and partners. This leads to a catch 22 situation for many companies.
- Disruptive innovation is unpredictable: Established companies are used to predictable return of investment on investments in sustaining innovations. Disruptive innovation typically follows a power function meaning that most innovations fail, but the few that succeed generate outsized returns. Having to go through dozens of attempts until striking gold is difficult to stomach for most leaders.
- Subsidizing one side of the market is hard: In many cases, a company aspiring to become a platform company needs to subsidize at least one side of a multi-sided market and potentially all sides for a period of time in order to reach the “ignition point”. Companies that have become successful in a traditional value chain have difficulty in subsidizing its customers as the focus tends to be on margins.
- Data ownership: Especially in the B2B space, it is often unclear who owns the data and the customer relationship and if it is clear, the company needs to provide something of value in return for gaining access to relevant data. Especially for traditional companies, sacrificing certain revenue from existing customers for potential revenue from a nascent business around data is difficult to stomach.

Although the above does not necessarily constitute a research agenda, it is a representative overview of some of the challenges that traditional product companies looking to transition towards a software, data and/or platform business experience. As a research community, rather than only focusing on new companies that are born digital, we should also study the challenge of transformation and provide solutions to existing companies looking to continue to be successful in a digitalized world.

3.3 Innovation + Velocity + Pivoting in Software Production: The New Normal

*Christoph Bussler*² (Oracle Corporation – USA, christoph.bussler@oracle.com)

The new normal in software production is that significant innovation has to be delivered at rapid development velocity with the ability to pivot at any time reflecting customer preferences, feedback and uptake.

The predominant software engineering methodology supporting the new normal in software production is a combination of agile methodology, A/B testing (if possible at all) and daily releases into production.

Why is software production and the resulting software products and services still rough sailing despite the advances in software engineering methodology?

- Observation 1: Everybody immediately recognize and appreciates software or services that are fast, easy and simple to use as well as exhibit consistency in terminology, behavior and actions.
- Observation 2: How often do you come across “good” software products or services? And how often do you have a negative reaction? Software production teams are in general multicultural, multilingual, distributed (across time and geography) as well as differ greatly in level of ambition, education and experience.

The agile methodology gets in the way of producing “good” software: it does not provide a general development direction, does not enforce consistent use of terminology, does not foster a consistent software architecture, and documentation as well as planning take a back seat – if present at all.

Time pressure gets in the way as well: shortcuts are taken and functionality is implemented incompletely focusing mostly on the main execution path (the Happy Path).

An interesting research topic to support the new normal would be “improvement infusion” by providing feedback through continuous automated observation of software production engineering activities. For example, a terminology analysis environment can observe the use of terminology in team communication and point out (possibly) inconsistent use.


Areas of analysis can be team communication, code and code changes, engineering and end user documentation, as well as test case stability and performance.

Types of feedback can be highlighting of inconsistent use of terminology, incomplete functionality implementation, requirements vagueness and instability, use case incompleteness and instability, test execution success rate degradation and missing test coverage. The result of “improvement infusion” would be the increase of software production quality based on observations, not regulation and constraints. “Improvement Infusion” supports the velocity and pivoting without attempting to change the predominant engineering culture.

² The views expressed here are my own and do not necessarily reflect the views of Oracle.

3.4 Platform versus Non-Platform Company Performance: Some Exploratory Data Analysis, 1995-2015

Michael A. Cusumano (MIT Sloan School of Management – USA, cusumano@mit.edu)

License  Creative Commons BY 3.0 Unported license
© Michael A. Cusumano

Numerous publications have described platform companies and their strategies and operations. However, there has not been a large-sample statistical study answering questions such as: Are platform companies more profitable than non-platforms? Are they more valuable? Are there differences among types of platform?

As an exploratory analysis, we divided all platforms into two basic types for innovation and transactions. Innovation platforms provide common building blocks that ecosystem partners can use to create “complementary” products and services. Microsoft Windows, Google Android, Apple iOS, and Amazon Web Services are commonly used operating systems and cloud computing services that serve as innovation platforms for computer and smartphone ecosystems. Transaction platforms make it possible for people to access or buy and sell a variety of goods and services, or to share information. Google Search, Amazon Marketplace, the Facebook Social Network, Twitter, and Tencent’s WeChat are examples of commonly used transaction platforms.

We defined a platform company as a firm that had at least 20 percent of revenues from businesses driven by network effects. We analyzed the Forbes Global 2000 list for 2015 and counted 46 platform companies, with 19 innovation and 27 transaction platforms. We then created a data set of over 30,000 yearly firm observations from 2005 through 2015.

We conducted simple regressions and T-tests. The means were significantly different between platform and non-platform companies on several dimensions: Operating profits divided by sales; market value multiples (ratio of sales to market value and price-to-earnings ratios); and absolute sale levels. Compared to transaction platforms, the innovation platforms had higher market values, sales, operating income, employee numbers, and R&D as well as sales and marketing expenditures. Transaction platforms had higher market values.

We confirmed that publicly listed platform companies were more profitable and more valuable than non-platform companies. However, we also identified several problems with this type of study that warrant further discussion.

3.5 Platform Elasticity for Fast Time-to-Critical Mass and Take-Off

Samuel Fricker (FHNW University of Applied Sciences and Arts Northwestern Switzerland – Switzerland, samuel.fricker@fhnw.ch)

License  Creative Commons BY 3.0 Unported license
© Samuel Fricker

We propose that keystones should design platforms for elasticity if they want to achieve a short time-to-critical mass for network effects to take off. We came to that position in our work for building a marketplace for open development of systems of artificial intelligence (AI) ³. There, the extent of third-party data, talent, and AI algorithm offerings as well as

³ H2020-ICT-01-2016 project www.bonseyes.com

the effort and convenience of trying and abandoning such offerings seems to affect platform adoption.

Elasticity is a key characteristic of Cloud computing and stands for rapid on-demand automated provisioning of capabilities (scaling out), possibly in a self-service mode, and rapid releasing of these capabilities (scaling in) ⁴ for efficient resource management ⁵. The Cloud is considered an essential platform for digital businesses and could generate a revenue of EUR 44.8 billion in Europe in by 2020 ⁶.

Elasticity could be brought to any platform with mechanisms to provision, use, and release assets on-demand while enforcing business models, preventing misuse, and enabling trusted choice. We posit that the convenience of value access offered by elasticity could affect the threshold of individuals for getting engaged. The smaller the threshold is, the faster the critical mass of adopters is reached and the network effects take off, letting the platform and a healthy ecosystem self-sustain ^{7 8}.

3.6 Research statement

Jens Foerderer (University of Mannheim – Germany, foerderer@uni-mannheim.de)

License  Creative Commons BY 3.0 Unported license
© Jens Foerderer

Researchers are interested in platform strategies because they involve a fundamentally different set of decisions than conventional “pipeline” strategies. Pipeline strategies create value via a linear series of activities in the sense of the classic value-chain model. Inputs at one end of the chain (e.g., resources) are transformed in various steps into a valuable output: the finished product. In contrast to pipeline strategies, platforms create value by leveraging the innovation capabilities of an independent “crowd” outside of the focal firm’s boundary. When firms follow a platform strategy, value-creating activities are less concerned with the coordination of production and supply but rather with the orchestration of complementary products and services. Thus, platform strategies require the focal firm to focus less on designing, developing, and distributing products but rather to focus on implementing an effective governance of third parties.

The governance of third parties requires a more fundamental and empirically assessed understanding with regards to **cooperation and competition** mechanisms. The phenomenon of platform owners actively competing with complementors has attracted attention substantial attention by researchers and policy-makers. Yet, our understanding today is particularly limited with regards to three questions:

- **Under which conditions does competition with complementors hurt or promote complementary innovation?** Extant research so far has yielded contradictory findings, suggesting that platform owners’ competition with complementors can crowd out

⁴ P. Mell, T. Grance (2010): “The NIST Definition of Cloud Computing,” *Communications of the ACM* 53, 6: 50.

⁵ G. Galante, L. de Bona (2012): “A Survey on Cloud Computing Elasticity,” *IEEE/ACM 5th Intl Conf on Utility and Cloud Computing*, Chicago, Illinois, USA.

⁶ P. Wauters et al (2014): “Measuring the Economic Impact of Cloud Computing in Europe,” *Final Report for the European Commission*. Deloitte.

⁷ E. Rogers (1995): *Diffusion of Innovations*, Free Press.

⁸ S. Jansen (2014): “Measuring the health of Open Source Software Ecosystems: Beyond the Scope of Project Health,” *Information and Software Technology* 56, 11: 1508-1519

innovation, but also increase the overall innovation within the ecosystem by attracting new consumers to the market and setting stronger incentives for differentiation [1]. It appears timely to study the conditions under which competition is innovation-promoting or innovation-hurting.

■ **How can intra-platform competition be effectively regulated by policy-makers?**

Almost overnight, antitrust and platform regulation has become a widely debated topic with policy-makers and regulators around the world considering platform regulation (EU, US, Japan) and some even implementing regulations (Russia). The regulations considered are mostly derived from standard antitrust models in the early 20th century, making it questionable whether they apply to two-sided platform markets. It appears timely to assess whether conventional antitrust approaches are effective in limiting intra-platform competition.

■ **How can platform owners effectively set agendas for the overall ecosystem?**

Platform ecosystems are characterized by independent complementors cooperating more or less at arm's length with a central platform owner. Yet, the platform's success (and competitive advantage compared to rival platforms) deliberately depends on coordinating not only individual complementors, but also coordinating the overall ecosystem. This is, however, a theoretically complex undertaking, as ecosystems often encompass thousands of firms that are impossible to coordinate via the mechanisms we know from conventional interfirm coordination literature. It appears therefore timely to understand the mechanisms (technological, informational, organizational) by which platform owners implement the overall agenda for the ecosystem.

References

- 1 George Valença, Carina Alves, Virgínia Heimann, Slinger Jansen, and Sjaak Brinkkemper. Competition and collaboration in requirements engineering: a case study of an emerging software ecosystem. In *Requirements Engineering Conference (RE), 2014 IEEE 22nd International*, pages 384–393. IEEE, 2014.

3.7 Feature-Oriented Development in Industrial Automation Ecosystems


Paul Grünbacher (Johannes Kepler Universität Linz – Austria, paul.gruenbacher@jku.at)

License © Creative Commons BY 3.0 Unported license
© Paul Grünbacher

Feature-oriented development has been proposed as an approach for engineering large-scale, variant-rich software systems. For instance, features models are widely used to capture the knowledge about product lines and configurable software systems. Features exist at different levels of granularity and define the perspectives of product management, technical solution architecture, and product configuration. We report ongoing work towards a feature-oriented software development approach we are currently developing with an industry partner in the domain of industrial automation ecosystems. Our work is based on empirical studies we conducted on the characteristics and use of features in industry. We present the FORCE modeling approach, which supports modularizing feature models for different purpose and different levels. Our tool environment exploits feature-to-code mappings and configuration-aware analysis. We further present our plans for supporting developer awareness on evolving features in ecosystems.

3.8 Analyzing the Mutual Quality Impact of Business Processes and Information Systems

Robert Heinrich (Karlsruher Institut für Technologie – Germany, robert.heinrich@kit.edu)

License  Creative Commons BY 3.0 Unported license
© Robert Heinrich

In complex technology landscapes new opportunities for the evolution of business goals and processes come up due to novel capabilities of software. Business processes (BPs) and information systems (ISs) mutually affect each other in non-trivial ways. The complex interrelations between BPs and ISs, however, are not adequately researched so far. Especially interrelations between quality properties (e.g., performance or maintainability) concerned with business people and those concerned with IS developers are not well understood. Frequently, the representation of quality aspect differs in the BP and IS domain.

Engineering methods for aligning one domain to the quality objectives of another are missing. One major reason for insufficient quality engineering is that current approaches lack an integrated consideration of quality aspects among several domains. Frequently, BPs and ISs are not well aligned, meaning that BPs are designed without taking IS impact into account and vice versa. Neglecting the mutual impact between BPs and ISs in development leads to serious practical issues. On the one hand, it is not known whether a particular requirement can be satisfied by a proposed IS design, because it is not known how the system is used in the BP, and how this usage affects the IS quality. On the other hand, it is unknown whether a particular requirement can be satisfied by a proposed BP design, because it is unknown whether involved ISs adequately support the adherence of the requirement. Decisions in IS development are not reliably made since important BP-related information may not be considered. This may decelerate IS development due to the rework needed in subsequent development phases. The same applies to neglected IS properties in BP development.

In our research we target the alignment of BP and IS designs by developing simulation and analysis techniques based on design models to predict the quality impact of mutual interrelations between BP and IS.

3.9 Research Statement

Armin Heinzl (University of Mannheim – Germany, heinzl@uni-mannheim.de)

License  Creative Commons BY 3.0 Unported license
© Armin Heinzl

Software Development Research is an intriguing topic which has a superior role in the digital age where digital products and services are uprising. Recent developments like the agile manifesto, Scrum, and Scrum in Scrum are prominent contemporary examples.

Cognitive and mental processes in agile software development teams and the question how agile teams scale, have been among my personal research topics during the past years. They embody the question how knowledge work unfolds in a knowledge centric society.

Today's software solutions are part of sophisticated (enterprise) software ecosystems. Exchanging and aligning knowledge between platform owners and complementors is another fascinating field of investigation. These practices are part of an overall innovation strategy of the participating firms which should be explored concurrently.

3.10 Software Engineering evolution

Mika Helenius (Finnish Computer Science Research Foundation & Finnish Information Processing Association TIVIA – Finland, mika.helenius@iki.fi)

License © Creative Commons BY 3.0 Unported license
© Mika Helenius

Software is the world's most powerful and pervasive general-purpose technology of modern times defined by competency, coordination and capability. Software creates new jobs, markets and industries that did not exist before at faster phase than ever. Software makes possible to create new intangible needs, service, spheres and forms of use anywhere to be exploited in the environment before they are understood. This new industrial development is called software platform based business and economy. Due its significant economic impact on industries and societies it is highly relevant and important for business and academia to understand software business, platforms, and ecosystems evolve. How these complex modular systems-of-systems platforms are engineered – conceived, designed, implemented and operated? What kind of competencies, capabilities, tools, methodologies and technologies are need in the rapidly evolving engineering process in rigor global competition? How we should define platforms in context of economic competition, hybrid warfare and humanity?


Platform are complex software system markets and industries. They are expensive, risky and time consuming to create. It is critical for owners, executives and investors to know how platform businesses are created to understand contextual productivity and innovation aspects. Current research has mainly focused on analyzing the existing business model, ecosystem and technology layers separately. Little attention has been paid on how complex platform systems are created with less risks using platform architecture as strategy management theory to gain business and societal results. Architectural thinking has not become main stream in the management discourse even it has been highlighted as key source of value. Platform architecture as strategy comprises of three layers industry and market layer, business and ecosystem layer, and technology system layer. Multisided platform architecture as strategy support past, present and future properties in the dynamic in vivo or simulated environment.

Software engineering research needs to be expanded to cover realistic and very large-scale engineering and production challenges by establishing large scale problem based learning environment for students, practitioners and researchers. This expansion of empirical software engineering to large-scale complex systems would allow future engineering graduates and researcher study digital grand challenges in realistic settings to understand what is needed in the ecosystems to deliver value in and for the environment.

Societies need knowledge and skills to create sustaining and balanced ecological platforms, which comply with current norms, values and ethics to redefine how value is spread and cultures are saved. Platform architecture as strategy is key in solving grand challenges.

3.11 Position Statement

Georg Herzwurm (University of Stuttgart – Germany, herzwurm@wiu.s.bwi.uni-stuttgart.de)

License  Creative Commons BY 3.0 Unported license
© Georg Herzwurm

Digital goods such as software are specific in economies of scale and network effects and thus require specific strategies and concepts for design, development and marketing. Due to the progress of digitalization in almost any market, the observed phenomena and measures gain relevance and importance in business and academia. Hence the emerging convergences in industries, suppliers, businesses and products cause a game change in markets for digital or digitized goods.

The door to the digital world is opening for more tangible goods employing the Internet of Things (IoT), actuating convergence of digital and analogue markets and value creation systems, enabling the atomization of products / services (e.g. bundled microservices instead of apps) and increasing the number of products and services and their providers. Platforms enable cooperation (i.e. development and sales) of the value creation partners. Since these partners may be cooperation partners and competitors coincidentally, there is cooperation.

Digitization is leading to a sustainable change in the software business towards a platform economy offering a huge potential for innovative business models, creating and satisfying customer needs for business success. However, successful digitized business models require on one side mastery of technology on the other side commercial expertise and HR skills for a mindset change of employees and customers.

Current research is mainly driven by technology and fosters digital innovations which will lead to business success only if technology and innovations meet customer needs, whereas needs and benefits may not be revealed yet. Its disclosure may manifest the innovation and provide the key to business success. Hence successful solutions require at first thorough understanding of customer needs and then design and development of matching processes and products.

We thrive for methods geared to design, development and market successful digitized products and business models within the paradigm of customer-centricity and embedded into a quality management framework.

We aim at systematic approaches and concepts creating sustainable customer benefits and value, designing and developing competitive, solid and profitable solutions (business value) employing customer-centered requirements, quality and product management.

3.12 From Managing Your Ecosystems to Repositioning Your Business

Helena Holmström Olsson (Malmö University – Sweden, helena.holmstrom.olsson@mau.se)

License  Creative Commons BY 3.0 Unported license
© Helena Holmström Olsson

To engage with external actors and to exchange value as part of a larger business ecosystem is one of the most prevailing trends in today's business environment and companies across domains are increasingly realizing the many benefits with engaging with external partners. To proactively engage with suppliers, vendors, distributors, retailers and customers brings with it a number of opportunities that do not present themselves when serving customers in

a one-to-one relationship which, up until now, has been the most common strategy for most companies.

In previous work and as part of the Software Center collaboration, we identified three different types of ecosystems that companies operate in. These ecosystems are related to *innovation, differentiation and commodity* and they are inherently different in nature. Typically, companies seek to involve with others and use collaborative strategies when it comes to innovation, they exclude partners and use competitive strategies when it comes to differentiation and they utilize external resources and, again, use collaborative strategies when managing their commodity ecosystem. In our work, we developed strategies for helping companies position themselves in ways that help them gain competitive power, maximize value and utilize their partner network.

However, for a company to manage – and to position itself within its existing ecosystems – is only one part of the challenge. The other – and even more important part – is to be able to re-position oneself in order to shift the power balance between oneself and the ecosystem partners when needed. Key reasons for re-positioning include e.g. to extract even more value out of the ecosystem, to avoid commoditization of one's role in the ecosystem and to avoid potential disruption.

Recently, and based on our work with the Software Center companies, we have identified a number of challenges that companies face when trying to reposition themselves, and with previous studies focusing primarily on how to manage ecosystems rather than how to reposition in ecosystems we see this as an area to explore further in future research.

3.13 Some Research Directions for Software Ecosystems & Platforms

Sami Hyrynsalmi (Tampere University of Technology – Finland, sami.hyrynsalmi@tut.fi)

License © Creative Commons BY 3.0 Unported license
© Sami Hyrynsalmi

Software ecosystems, platforms and application stores, consisting of various different organisations, have changed how software products and services are produced, distributed and maintained. The emergence of this new platform economy has been painful for some of the old kings of the castle while some newcomers have been able to built successful and sustainable business in these new environments. Nevertheless, to support software producing organisations (SPOs) in the evolving platform economy, further research work with empirical evidence is needed. In the following, I will present areas that I believe would be fruitful for further inquiries.

First, evaluating the sustainability and well-being of an ecosystem. The current work on business and software ecosystem health has produced various measures, yet there is only little empirical evidence available to support the current researches.


Second, studying the influence of multi-homing on software ecosystems. One of the key characteristic differentiating software ecosystems from business ecosystems is the relative easiness of a SPO to offer their products and services in several competing ecosystems at the same time. However, this area has been studied only a little.

Third, understanding software start-ups as a part of niche creation and renewal of an ecosystem as well as the whole software industry. The majority of existing software ecosystem and platform literature has focused on the keystones and ecosystem orchestrators whereas there are studies addressing independent SPOs, yet their number remains small.

In our research, we have thus far focused on software ecosystems and often from the perspective of independent SPOs. In addition, we have studied the financial aspects of software start-ups and we are interested to focus more on new entrepreneurs entering an ecosystem. However, in starting software companies, business development practices are often tightly intertwined with software development activities and thus a holistic view is needed to understand the start-ups.

3.14 Engineering FLOSS Ecosystems for Impact and Sustainability

Zhi Jin (Peking University – China, zhijin@pku.edu.cn)

License  Creative Commons BY 3.0 Unported license
© Zhi Jin

Millions of participants, from independent volunteers to paid representatives of companies or government organizations, are creating and maintaining a huge number of open source software (OSS) eco-systems, such as the Linux Kernel, Android, and OpenStack, which have had a significant impact, not only on the software industry, but also on software-intensive organizations in both the public and private sectors. Despite the substantial amount of research on FLOSS in disciplines such as software development, organizational science, management, and social sciences, it remains unclear how and why OSS ecosystems form, how they achieve their impact, or how they sustain themselves. The data recorded in vast open source and commercial software repositories provide rich opportunities to investigate how people develop software and how they interact with each other and with their environment to accomplish their tasks, and how large-scale projects grow and sustain adapting to the ever-evolving environment. The following shows our studies on this area ranging from the learning trajectory of developers to the participation of companies, and to the health and sustainability of communities and ecosystems.

- Q1: How to retain people?
 - In GNOME and Mozilla, over 10 years, more than 70% of contributors are One-Time-Contributors. Only 3.6% in Gnome and 0.9% in Mozilla joiners become Long Term Contributors.
 - People behave differently when joining. The intension for joining the community depends on the willingness, the macro-climate, and micro-climate.
 - Willingness and climates impact the chance of becoming Long Term Contributors. Practice of the first month affects chance of becoming long term contributors.
 - This can be used to predict who will stay with the project for long term based on the initial behavior of the newcomer. And the FLOSS community could devote their valuable attention to people who are more likely to be useful to the sustainability of the project.
- Q2: How companies participate in FLOSS?
 - For all the types of projects, the full-solution-oriented companies lead the development. Their proportion of commits is 80.99% in median. Together with the specific-business-oriented companies they contribute more than 88% of the commits and approximately 89% of the developers for almost all project types in median.
 - Community-oriented organizations help the team building: focus on developing infrastructure, deployment etc.

- Usage-oriented companies improve user experiences: have a preference on the development of infrastructure, deployment, management tools, document, etc.
- Q3: How do Ecosystems Evolve and Scale?
 - How fast does the Linux grow? The amount of work continues to grow. Tasks for drivers contains most of the changes of the system. The number of joiners has been decreasing.
 - How do the maintainers and their workload evolve? The number of maintainers keeps increasing. The average workload of maintainers seems to decrease instead of growing. 80% work is done by 20% people in drivers, modules have a much more even distribution of work.
 - How well does the Linux ecosystem scale? Adding more maintainers to a file yields only a power of 1/2 increase in productivity, e.g., four parallel maintainers are needed to double the overall output. This suggests limits to scalability that can be achieved by adding multiple maintainers to the same subsystem.

3.15 Academic Structures and Terminology

Hans-Bernd Kittlaus (InnoTivum Consulting & ISPMA (International Software Product Management Association) – Germany, hbk@innotivum.de)

License © Creative Commons BY 3.0 Unported license
© Hans-Bernd Kittlaus

Research in this area faces some structural and fundamental problems that we need to address before it makes sense to discuss individual items of a research agenda on a more detailed level.

The existing academic structures do not sufficiently support research in the area of business aspects and product management of software-intensive products

One of ISPMA’s objectives from the start almost 10 years ago was the establishment of software product management as a discipline of its own on both the academic and industry side. While we have made a lot of progress on the industry side, we have failed on the academic side. The vast majority of computer science faculties and economic faculties do not want to deal with this research area (exceptions granted). This is a problem all over the world despite the extreme importance of the area for more and more industries. We may come up with the most wonderful research agenda, but that will not help if there are not enough researchers to do the work. One of the results of this seminar needs to be a manifesto to change this situation in academia, maybe with political and/or industry help.


Our terminology is not sufficient and seriously lacking in relation to what is happening in this area on the industry side

It is a core responsibility for any academic discipline to create and standardize a domain-specific language. Our terminology is seriously lacking. The description of this Dagstuhl seminar is a case in point. Why do we use the term “software production”? While the implied analogy with the manufacturing industry may be politically helpful, we all know that it is semantically misleading. In manufacturing “production” does not include the development of the product as a “type”, but only the creation of the physical “instances”. That is fundamentally different from what we (probably) mean by software production. When we talk about “software producing organizations”, do we really only mean open source consortia and commercial software product companies (like the text says), or do we

follow the broader semantics of the term which includes companies that produce software for software-intensive products (the text does mention Tesla and Volkswagen), as well as corporate IT organizations and professional service companies who develop custom software? We need to start an initiative to develop a state-of-the-art terminology for our field.

3.16 Position Statement

Thomas Kude (ESSEC Business School – France, kude@essec.edu)

License  Creative Commons BY 3.0 Unported license
© Thomas Kude

As a result of digitization, questions related to technology have become ubiquitous. While previously mostly dealt with in IT departments and tech companies, almost all organizations are now seeking guidance as to what capabilities are needed in a digital world and to what extent and how these organizations should transform into digital businesses. These developments provide ample opportunities for research in technology-related fields, such as information systems and software engineering, to make important contributions.

To do so, a sociotechnical view on digitization is needed. Currently, there seems to be a tendency to give primacy to technology alone, e.g., in the context of machine learning. Notwithstanding the undisputable benefits resulting from technological advances, there are indications that digitization reinforces the need for carefully establishing systems comprising technology, individuals or groups of individuals, as well as related activities.

In my current research, I take such a sociotechnical perspective to study different aspects of digital business at different levels of analysis. For example, I examine the governance of platform ecosystems in different empirical contexts. Some of my recent work was set in the context of enterprise software platforms and studied the motivation of complementors to join an ecosystem as well as governance practices and knowledge boundaries between platform owners and complementors. Some other recent work examined mobile platforms and the governance moves of platform owners along with consequent behaviors of complementors.

As another example, I study agile software development teams. Relying on survey studies and participant observations, I examine the implications of agile development practices, such as pair programming and code reviews, on the performance of software development teams. My focus is on the teamwork factors through which these effects are carried. Recent work includes studies related to the shared and distributed cognition and the backup behaviors among developers.

3.17 Position Statement

Alexander Mädche (Karlsruher Institut für Technologie – Germany, alexander.maedche@kit.edu)

License  Creative Commons BY 3.0 Unported license
© Alexander Mädche

The digital transformation of businesses and society makes most of today's organizations to some kind of software producing organizations (SPO). Driven by accelerating internal and external digitalization, organizations develop and deploy software in order to increase productivity, extend and enrich existing products and services as well as create entire new business models.

In my research I specifically focus on **Information Systems Engineering (ISE) following a socio-technical paradigm**. My research topics are allocated in the domain of scaling software producing organizations. Specifically, I look into three major fields: 1) data-driven ISE, 2) user-centered ISE, and 3) individuals and teams in ISE.

First, the basic idea of following a more **data-driven approach** to ISE goes to back to my PhD project on “Ontology Learning for the Semantic Web”. There, I investigated methods and techniques in order to semi-automatically construct ontologies from existing unstructured and structured data sources. In the last years, I also looked into questions of requirements elicitation following a semi-automated mining approach (requirements mining) as well as the semi-automatic construction of business models (business model mining) from structured data of organizational information systems. Second, I look into **user-centered ISE** because I strongly believe that usability and UX should be much more valued and emphasized by software producing organizations. Therefore, I investigate the role of design techniques as well as user involvement and participation in (agile) software development processes. Third, I’m interested on the social side in the form of **individuals and teams in ISE**. In my research group, we did carry out a number of empirical studies in this context, e.g. on age stereotypes in agile teams, emergence of team agility, coordination in large-scale agile software development. Recently, we leverage NeuroIS concepts (e.g. physiological signals, eye-tracking) in order to capture affective- cognitive states of developers and on this basis adapt the work environment. E.g. we are currently running a field study in cooperation with SAP SE in order to measure the flow state of developers in a SCRUM team and on this basis to intelligently adapt IT-mediated interruptions at the work place (e.g. emails).

3.18 Challenges in Software Ecosystems and Product Development

Efi Papatheocharous, (RISE SICS – Sweden, efi.papatheocharous@ri.se)

License © Creative Commons BY 3.0 Unported license
© Efi Papatheocharous,

The German computer science pioneer Karl Steinbuch in 1966 remarked: “In a few decades time, computers will be inter-woven into almost every industrial product.” The increasing prevalence of software ecosystems and platforms today calls for the ability to **augment solutions and support an emerging portfolio of leading technology solutions and trends**. It is unquestionable to design or use any software technology without taking into account digitalisation trends the emerging technological innovations (e.g., Big Data, Internet of Things, Systems of Systems) and without considering standing on the shoulders of a multitude of layers of platforms and ecosystems.

In our research we investigate efficient ways to organise and carry out product development in software ecosystems with the target to satisfy mutual and conflicting requirements from the involved parties.

This led to the formulation of the overall research questions (RQs):

- RQ1. What are the implications on the business models of the different actors, when moving from a traditional supply chain to a dynamic SECO?
- RQ2. What are the options for improved design of product architectures to handle the contradictory requirements of openness, flexibility and dependability, and to allow efficient product line management?


We identified challenges with respect to 3 categories: a) organizational, b) technical, and c) business and use a schema to conceptualise an ecosystem for Federated Embedded

Systems encompassing of four layers: actors, business processes, services and components. We described in an explorative case study (based on interviews with 15 senior staff members at 9 companies related to Embedded Systems) our findings mapped according to the Business Model Canvas (BMC) to highlight the interrelated parts and characteristics of the domain. Openness in SECO was evaluated in 7 companies including 8 practitioners taking into account their practices and methods.

Moreover, we target efficient and informed architecture formulation through the selection of existing components and services, and fast architectural adaptations which is crucial for companies' success, with a systematic approach in the decision-making process with respect to components, services and platforms.

3.19 Elements of Platform-based Ambidexterity: An Empirical Investigation

Balasubramaniam Ramesh (Georgia State University – USA, bramesh@gsu.edu)

License  Creative Commons BY 3.0 Unported license
© Balasubramaniam Ramesh

The notion of platforms has gained significant attention in both research and practice in information systems. Platform based approaches range from developing a family of products that address common and variable needs of a market to platform-based ecosystems that include open platforms and multi-sided markets. Much of the prior work on platforms focuses on challenges involved in platform-based development and elements of managing platform ecosystems. Platforms have been considered to play a role in facilitating organizations to handle the needs of various market segments efficiently, thereby supporting exploitation. Platforms have also been considered as a driver of innovation, a way to support organizations in their explorative endeavors. We argue that the notion of platforms can be leveraged in developing an approach that will help organizations simultaneously achieve both exploitation and exploration.

Organizations have long recognized the need to address tradeoffs when faced with constraints in meeting conflicting demands. Extensive research on organizational ambidexterity highlights antecedents and practices that play an important role in enabling organizations to balance exploration and exploitation. We posit that platform-based approaches can help organizations achieve ambidexterity.

This leads us to our key research question: “How can organizations leverage the notion of platforms to achieve ambidexterity?” Through a multi-site case comparative case study, we answer this question by identifying specific aspects of a platform-based approach to achieve ambidexterity. Our framework brings together the three important elements of a platform-based approach – development of product platforms, development of process platforms, and development of value-based platforms. We outline various elements of organizational context that drive the different aspects of a platform-based approach. Our findings detail how various factors such as the structure of the organizational units, flexibility of processes, environmental constraints faced, commonality and uniqueness of market needs, risk propensity of the platform developer and the other stakeholders, and culture/value that shape the development of platform based approach shape the ability of the organization to achieve ambidexterity.

3.20 Minimum Viable Products – The Road Ahead

Guenter Ruhe, (University of Calgary – Canada, ruhe@ucalgary.ca)

License  Creative Commons BY 3.0 Unported license
© Guenter Ruhe,

The dynamics of software products has enforced changes in the way products are developed. “Minimum Viable Products” stands for the development of products being viable to the users and customers, but are minimum in terms of the effort and functionality invested. The idea is to accelerate customer feedback not only early, but with minimum effort. Independently, “Technical Debt” is happening in all forms of conscious and non-conscious compromises done in the process of developing a product (version), deviating from what is understood the process “should be” in comparison to how the process actually.

There are numerous research questions around MVP. They relate to questions like:

- How to define experiments being the backbone for a MVP variant?
- Do we run MVP experiments concurrently or just incrementally?
- How often do we change MVP’s?
- Where the inspiration for a specific MVP comes from?
- How sensitive the definition of MVP’s is to different groups of stakeholders?
- How far are features of MVP’s implemented and evaluated to make a decision for their inclusion in future products?
- How many new features are accommodated in a MVP?

The seminar discussed some of these questions and their practical relevance from an industry perspective.

3.21 Only few can be platform owners

Kari Smolander (Lappeenranta University of Technology – Finland, kari.smolander@lut.fi)

License  Creative Commons BY 3.0 Unported license
© Kari Smolander

There is a growing interest on software-based platforms and platform economy and much knowledge has been collected on platform ecosystems and their governance. However, there are fewer attempts to investigate the enterprises that are not dominant players in the platforms, but need to integrate to various platforms to sustain or extend their business capabilities.

In the digital economy, integration has become more important than ever. The emerging technologies, such as the Internet of Things and Big Data, strongly emphasize integration. Improving the efficiency and responsiveness by integrating information systems within and outside the company is unavoidable in the modern collaborative business environment. Enterprise applications and systems can no longer exist as stand-alone entities, but instead they must interact with other information systems inside and outside the company walls. Integration has become a necessity to satisfy customers.

Our activities and transactions are increasingly happening in software-based platforms and ecosystems, such as Facebook, Google, WeChat, AliPay and various industry-specific platforms, that are not directly controllable by single enterprises. We lack understanding of how enterprises should approach and manage their integration to software-based platforms. This integration is often a necessity, because these external platforms increasingly guide the

actions of customers and business partners of enterprises. There is a wealth of studies on platforms themselves and their evolution, platform governance and leadership of keystone players like Google, Amazon, and Apple and on platform creation strategies, but we lack in-depth knowledge of integration to platforms. The integration problems of enterprises that are not dominant players in the platforms have received only little attention. Still, they are the vast majority of enterprises and they need to integrate to various platforms to sustain or extend their business capabilities. This integration can bring business and security risks and a platform lock-in is often a consequence. There is an urgent need to study this, since only extremely few can be platform owners.

3.22 Data for Performing Research on Software Business, Platforms, and Ecosystems

Diomidis Spinellis (Athens University of Economics and Business – Greece, dds@aueb.gr)

License  Creative Commons BY 3.0 Unported license
© Diomidis Spinellis

An identified research challenge in the area of software business, platforms, and ecosystems research is the lack of easily accessible research data [1]. Scientists require these data first, to gain insights regarding theory of software production processes, and second, to empirically validate proposed theories. The lack of data can be addressed by collecting, processing, curating, and making available suitable data sets.

Businesses are understandably reticent about sharing data concerning their processes, operations, outcomes, and strategy. Yet, many types of data are either already available or can be obtained from openly accessible sources, such as corporate web sites, software forges, and app stores.

- Data and metadata of a company’s web presence as well as web log data can reveal details regarding its software development processes and the adoption of technology platforms [2].
- App stores can be mined to gather information regarding products, reviews, and advertising networks [3].
- Open source software forges, such as GitHub [4], can provide data regarding corporate open source software strategy, contributions, and development processes [5].
- Continuous integration [3], Q&A forums [3], and code review servers [6] can be further mined to extract more detailed product and process data.

The outlined data can provide insights on how diverse companies deal with technology challenges, such as software complexity, security, and reliability, the extent and dynamics of platform and technology adoption, and the performance of specific business models. As this approach leans toward empiricism rather than rationalism, it requires disciplined data analysis protocols, such as registered reports [7] and close alignment with theory building.

To move forward as a community in the proposed directions we must collect, curate, and publish existing data sets;⁹ determine areas where new data are required; encourage the development, release, and publication of new data by recognising their scientific value; extract data from untapped data sources; and build upon the data showcasing their utility through advances in theory and practice.¹⁰

References

- 1 Pekka Abrahamsson, Jan Bosch, Sjaak Brinkkemper, and Alexander Mädche. Dagstuhl seminar: Software business, platforms, and ecosystems: Lifecycle challenges of software producing organizations. Available online <https://materials.dagstuhl.de/files/18/18182/18182.SWM.Other.pdf>, 2018.
- 2 Diomidis Spinellis and Vaggelis Giannikas. Organizational adoption of open source software. *Journal of Systems and Software*, 85(3):666–682, March 2012.
- 3 Muhammad Asaduzzaman, Ahmed Shah Mashiyat, Chanchal K Roy, and Kevin A Schneider. Answering questions about unanswered questions of Stack Overflow. In *MSR 2013: 10th Working Conference on Mining Software Repositories*, pages 97–100. IEEE Press, 2013.
- 4 Georgios Gousios and Diomidis Spinellis. GHTorrent: Github’s data from a firehose. In Michele Lanza, Massimiliano Di Penta, and Tao Xie, editors, *9th IEEE Working Conference on Mining Software Repositories (MSR)*, pages 12–21. IEEE, June 2012.
- 5 Diomidis Spinellis. Tools and techniques for analyzing product and process data. In Tim Menzies, Christian Bird, and Thomas Zimmermann, editors, *The Art and Science of Analyzing Software Data*, pages 161–212. Morgan-Kaufmann, 2015.
- 6 M. Mukadam, C. Bird, and P. C. Rigby. Gerrit software code review data from Android. In *MSR 2013: 10th Working Conference on Mining Software Repositories*, pages 45–48, May 2013.
- 7 Christopher D Chambers. Registered reports: A new publishing initiative at Cortex. *Cortex*, 49(3):609–610, 2013.

3.23 Software Ecosystems Research Agenda Update

Slinger Jansen (*Utrecht University – the Netherlands, slinger.jansen@uu.nl*)

License  Creative Commons BY 3.0 Unported license
© Slinger Jansen

Increasingly, software producing organizations collaborate in networks that have become known as software ecosystems [1]. The intricate structures of platforms upon platforms [2] enable rapid innovation like never before. In our research laboratory, we explore how these platforms collect knowledge about the platform itself, the applications running on it, and the end-users that make use of these applications. Through examples and case studies is shown that software operation knowledge in software ecosystems is essential for creating better software, happier users, and more productive developers [3].

In research, the term ‘ecosystem’ is popular. Terms such as the sales force automation ecosystem, enterprise resource planning ecosystem, and artificial intelligence ecosystem are thrown around freely. This is a double edged sword: the term becomes increasingly popular

⁹ See e.g. <https://github.com/awesomedata/awesome-public-datasets> and <https://github.com/dspinellis/awesome-msr>.

¹⁰ The project associated with this work has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 732223.

but it also diffuses the meaning of the term in its context. Simultaneously, the term “software ecosystem” is sometimes attacked for being too specific, when terms such as business or digital ecosystem suffice. However, with the word ‘software’ we emphasize the phenomena caused by the critical component of software underlying these ecosystems, thereby scoping our work around themes such as software business, software engineering, software as a service, and open source.

One of the big challenges of the field is its multi-disciplinarity. It includes work about open source from software engineers, works about automotive platforms from management and information scientists, and works about visualizations from computer scientists. We reiterate that this domain is relevant from different perspectives, and should thus be considered multi-disciplinary.

References

- 1 Slinger Jansen, Eko Handoyo, and Carina Alves. Scientists’ needs in modelling software ecosystems. In *Proceedings of the 2015 European Conference on Software Architecture Workshops, ECSAW ’15*, pages 44:1–44:6, New York, NY, USA, 2015. ACM.
- 2 S. Brinkkemper, I. Soest, and S. Jansen. Modeling of product software businesses: Investigation into industry product & channel typologies. *Information Systems Development*, pages 307–325, 2009.
- 3 Joost te Molder, Ben van Lier, and Slinger Jansen. Clopenness of systems: The interwoven nature of ecosystems.

3.24 AI and Software Business

Pasi Tyrväinen (University of Jyväskylä – Finland, Pasi.Tyrvaainen@jyu.fi)

License  Creative Commons BY 3.0 Unported license
© Pasi Tyrväinen


Changes in software business are driven mainly by demand for new applications, development of hardware technology, development of new reusable software artefacts (platforms etc.) as well as the need to combine the previous three into solutions with business value. The short history of software business has shown that the variation in each of these drivers requires effort of skilled software personnel to capture the functionality into form of a software artefact. The promise of reuse, end-user programming, model driven software development, components, and code generation have not been able to remove the need of human intelligence in the process and the volume of software development effort has continuously had a growing trend. So far, the majority of business activity related to software has been in bespoke software tailored for enterprise use regardless the grooving emphasis on platforms and standardized products delivered as a service over the Internet.

Recently, there has been public discussion on the possibility to use artificial intelligence (AI) as a means to reduce or even replace the human effort in software engineering. However, as majority of the AI effort (55%) is focusing on analytics rather than symbolic methods needed for increasing the automated part of software creation process, this statement seems not to be valid. Further, when recalling the AI solutions to operate dominantly under closed world assumption, the impact of such systems is likely to remain in the phases starting from requirements specification while most of the drivers for new software dominantly demand major software engineering effort in defining the need and elaborating it to a requirements

specification sufficient for automated code creation. Thus AI is unlikely to respond to the three change drivers mentioned above.

3.25 High-speed and Sustainable Development of Software Startups

Xiaofeng Wang (Free University of Bozen-Bolzano – Italy, xiaofeng.wang@unibz.it)

License  Creative Commons BY 3.0 Unported license
© Xiaofeng Wang

“Software is eating the world”, and software startups are disrupting the world. Uprising startups, such as Airbnb, Uber and Spotify, are extremely active and volatile elements in the ecosystem of the global economy. However, building a successful software startup is extremely difficult and the failure rate is strikingly high. Based on the research of our software startup research network, we have identified three grand challenges that software startups and ecosystems are facing:

- How to build a software startup in a high-speed and sustainable manner?

Specific challenges come from various directions, including software development, entrepreneurial team building, business model definition and fund raising. Product and market related issues demand that a software startup acts and reacts with high-speed in extreme uncertainty, whereas other issues, e.g., building entrepreneurial teams, or maintaining acquired customer base, mandate that a startup works in a sustainable manner as it pursues its grand visions. These two aspects are not always compatible and need to be considered simultaneously and balanced properly. In addition, early stage startups are different compared to those that are scaling up. Different knowledge and practices are needed to succeed through different stages. How to maintain vision, passion and innovation momentum, typically the driving forces to initiate a startup, in the later stages of the startup lifecycle, is a challenge faced by software startups. Continuous innovation is therefore essential.

- How to provide better support to software startups?

As startups never live in isolation, their survivability depends on the startup ecosystems they are located in. Organizations such as business incubators, accelerators, business angels and venture capitalists positively influence their chances to survive and grow. However, these ecosystem players need to allocate their resources effectively by figuring out what supporting measure will have the greatest impact for each particular startup, considering its stage of development and individual strengths and weaknesses. Investors equally need to be able to reliably assess the investment risk and possible return. Intermediaries have to deal with a wave of new startup support and development tools such as crowdfunding platforms which are becoming an increasingly important source of funding. Initial Coin Offering (ICO), a phenomenon closely linked to blockchain technology, is another emerging fundraising mechanism for startups at very early development stages. Decentralization and deeper personalization are new types of support needed by software startups. However how to utilize these mechanisms to support software startups is yet to be understood fully.

- How to better train current and future software startup founders?

Many software startup founders lack necessary knowledge to initiate their startup journey and blindly follow only gut feelings and/or a trial-and-error approach. No validated learning is obtained and accumulated to guide their practice. On the other hand, training and education offerings are diverse across different organizations and institutes, making it difficult to share best pedagogical practices.

These challenges are intertwined and have to be tackled collaboratively by all startup ecosystem stakeholders. Besides, there are more fundamental questions that the researchers interested in software startups need to answer before diving into the battle against these challenges. For example, what are software startups exactly? Are they fundamentally different from other types of startups? what research disciplines are relevant to obtain necessary and updated knowledge from? These fundamental questions need to be answered before we could start tackling the challenges listed above.

3.26 Understanding new development trends, exploring large data, and pushing the boundaries of innovation

Karl Werder (*University of Cologne – Germany, werder@wiso.uni-koeln.de*)

License  Creative Commons BY 3.0 Unported license
© Karl Werder

Software is an important element of the digital transformation. Software producing organizations (SPO) lead the way in product innovations and new ways of working. Let me share my elaborate on my perspective towards software producing organizations:

First, SPOs leading **new development trends**. Having XP, Scrum, or DevOps, SPOs continuously challenges their operating modes and find new innovating techniques to conduct development and design activities. These, as in the case of agile development, inspire industries beyond the software industry, partly due to their proven benefits to both, management and developers [1], and due to their applicability to all organizational sizes, from startups to large international enterprises.

Second, SPOs providing grounds for highly impactful research insights based on **large data**. Using a central repository is a standard tool in SPOs. These provide deep insights into the organizations development processes and enable researchers to answer questions that previously have been left unanswered [2]. Using this rich data source enables researchers from IS and SE domain to provide research contributions beyond their own field of research towards other parts of social sciences, such as project management, team research, leadership, and human resource management. It also helps scholars to better understand large scale software development practices.

Third, SPOs pushing the boundaries of **innovation**. A key assumption of innovation research is that innovation is a well-bounded phenomenon, focusing the investigation of fixed products. However, given software ecosystems and technological innovations such as block chain, innovation is much more fragmented and less defined. Open innovation facilitates the collaboration of manifold people without limitation to a product or timeframe, (e.g. [3]), as in contrast to (new) product development.

In my research, I focus on enhancing understanding the software development process and the people involved in it.

References

- 1 Karl Werder and Alexander Maedche. Explaining the emergence of team agility: a complex adaptive systems perspective. *Information Technology & People*, 31(3):819–844, 2018.
- 2 Karl Werder. The Evolution of Emotional Displays in Open Source Software Development Teams: An Individual Growth Curve Analysis. In *2018 IEEE/ACM 3rd International*

- Workshop on Emotion Awareness in Software Engineering (SEmotion)*, Gothenburg, SE, 2018. ACM.
- 3 Karl Werder, Remko W. Helms, and Slinger Jansen. Social Media for Success: A Strategic Framework. In *Pacific Asia Conference on Information Systems 2014*, page Paper 92, Chengdu, PRC, 2014. AISeL.

3.27 Improving handling business model changes for software-intensive organizations

Krzysztof Wnuk (Blekinge Institute of Technology – Sweden, krw@bth.se)

License © Creative Commons BY 3.0 Unported license
© Krzysztof Wnuk

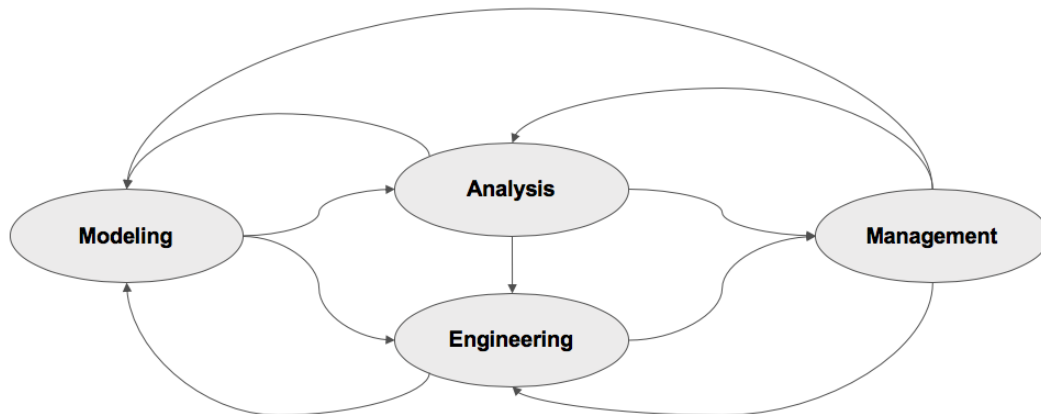
Software-intensive companies are recently undergoing significant transformations and are struggling with the alignment of business and technology change. Until recently, these companies handled increasing size and complexity by: 1) clearly distinguishing between the planning and realization layers for company strategy, product portfolios and individual products; and 2) handling change mainly in the realization layer and ensuring that the planning layer remains reasonably stable. Frequent changes into the realization layer were efficiently handled by various engineering paradigms and principles, e.g. software architecture, Software Product Lines, variability and configuration management, just to name a few.

Whatever change arriving to the strategy, portfolio or product level could be either earlier anticipated or received sufficient accumulation time in the realization layers, both of which helped to handle substantial size and complexity of software-intensive products.

However, the core of the recent transformation is that the speed of changes in the planning layer increases substantially and, in many cases, reaches the speed of changes in the realization layer.

A primary driver for this transformation is the digitalization of the business environment. For example, at the product level, the introduction of agile development and user communities, user groups for high-valued customers, and similar forums allow customers to interact directly with product development. Such frequent interaction transforms much of the traditional product road-map planning work, into a continuous software release including both feature-based upgrades and bug fixes. At the Product Portfolio level, the need to innovate and increase the value creation drives a transition towards Product Service Systems (PSS) and use-models.

New services are mixed with software products and re-usable components to create new product and solution offerings that can either be delivered as a cloud or as a more traditional product sale. Finally, at the strategy level, companies create or get engaged in various business ecosystems where they reinvent the value and work together with other ecosystem stakeholders to co-produce value. The same pattern is seen in most industries today, e.g., Porsche launch an open innovation platform aiming at taking the lead in an electric car business ecosystem, similarly as Amazon once turned their struggling online bookstore into a global shopping ecosystem by launching an online, cloud-based Web Service platform (AWS) in 2002. By the re-launch of AWS in 2006, with their Elastic Compute Cloud (EC2), Amazon popularized the term 'cloud computing' and became an active leader in a new IT industry. Similar examples are also recognized by Bharadwaj et al. as they coin the term digital business strategy as the fusion of business strategy and IT strategy. They point at limitations



■ **Figure 1** We identify four domains in which the software ecosystems research challenges can be categorized.

of traditional business models as “we need richer models that delineate inter-dependent ecosystems that evolve more rapidly than what we have seen in traditional settings”.

4 Working Groups

4.1 Working Group on the Software Ecosystem Research Agenda

Paul Grünbacher (Johannes Kepler Universität Linz – Austria, paul.gruenbacher@jku.at)

Jens Förderer (University of Mannheim – Germany, foerderer@uni-mannheim.de)

Zhi Jin (Peking University – China, zhijin@pku.edu.cn)

Samuel Fricker (FHNW University of Applied Sciences and Arts Northwestern Switzerland – Switzerland, samuel.fricker@fhnw.ch)

Rahul Basole (Georgia Institute of Technology – USA, basole@gatech.edu)

Slinger Jansen (Utrecht University, the Netherlands, slinger.jansen@uu.nl)

License © Creative Commons BY 3.0 Unported license

© Paul Grünbacher, Jens Förderer, Zhi Jin, Samuel Fricker, Rahul Basole, Slinger Jansen

Ten years after the Software Ecosystems research agenda at ICSE [1] it is time to take score and establish a research direction for the next decade. The domain has grown significantly, both in publications and interest, and overall there are signs of maturation of the domain. As the community is increasing its research effort, it is relevant to articulate themes that give direction to the research, avoid redundancy, and provide novel research avenues. In this report we express themes, challenges, research questions, and propositions, based on ten years of literature and research agendas in the field.

A software ecosystem is a set of actors functioning as a unit and interacting with a shared market for software and services, together with the relationships among them. Software ecosystems are pervasive and software producing organizations increasingly realize that it is the ecosystem that makes them and their technologies successful [2].

Digital business has become an essential pillar under most economies and it has been a driver of innovations for several decades. The introduction of new technologies and convergence of the Internet of Things, cloud technologies, and artificial intelligence, lead to a

	Definition	Issues	Recommendations
Modeling	Act of defining the primitives, hierarchies, the granularity, actors, structure, arrangement, behavior in the ecosystem.	No common language	harmonization of existing languages
		No scalable modeling method	Build a typology/classification of ecosystems and the actors involved
		Insufficient understanding of tipping points and genesis events	Advance a hierarchical view of platforms, develop a hierarchical modeling approach
		Little understanding of dynamics, adaptive, and emergent behavior	Simulate evolutionary models, perhaps using independent agents Predictive modeling
		Insufficient understanding of ecosystem boundaries	Develop a model of the boundaries involved in an ecosystem (parallel to the onion model)
		modeling of human and crowd behaviour	leverage agent-based modeling and innovation/diffusion techniques
Analysis	Define, collect, analyze metrics	metrics interpreted differently (what is centrality etc) --> data-dependent, no consensus regarding the interpretation	converge on measures, go to a standardization organization, adopt and compare existing measures, bring communities together and formulate recommendations, have regular exchanges
			Conceptualize ecosystem health
		Situational awareness and sense-making	Create visual representations at all levels
		Analysis of metrics over time	Leverage data science, machine learning techniques
		Data access and availability	Build a library of (temporal) SECO knowledge
Engineering	The definition and application of governance mechanisms for software ecosystem development.	lack of design knowledge for starting SECOs, difficulty of studying ecosystems, limited number of attempts for launching an ecosystem	- derive principles, e.g. from existing examples - simulation techniques and Quasi experiments for SECOs - allow in-laboratory research of SECOs - a catalogue of knowledge discovery mechanisms
		insufficient understanding of architectures that enable ecosystems	Create a software architecture pattern catalogue for software ecosystems (e.g., extension mechanisms)
Management	Orchestration of the ecosystem	Lack of understanding of the relationship between ecosystem health and technology adoption	Derive principles and theories about adoption and health
		Insufficient insight into value flows and human centered design	Create a value model for ecosystems Study Community/ecosystem canvas
		Lack of understanding regarding the theoretical mechanisms by which organizational actions/governance mechanisms (competition, cooperation, knowledge sharing etc.) shape ecosystem outcomes	Empirically test cause consequence models
		Insufficient insight into renewal of ecosystems	Create a collection of governance mechanisms

■ **Figure 2** During the Dagstuhl event these challenges were identified and categorized over the four domains.

myriad of new possibilities, but require an ecosystem approach for extensive adoption. These technologies are rapidly adopted, in large part due to the “ecosystemification” of the digital business; software producing organizations depend on each other to enable faster adoption of new technologies supplied by new entrants in the market.

Society, organizations, and economies are experiencing and anticipating fundamental changes that are shaped, embedded, and influenced by ecosystems. Ecosystems are social, technical, and economic systems that are large, multi-level, complex, dynamic, adaptive, emergent and global in nature, and concern a wide range of stakeholders (managers, policy-makers, and society), each with different perspectives and incentives. An interesting finding is that ecosystems cannot be created, but must be cultivated and fostered.

The complexity in scale, scope and dynamics makes systematic modeling, analysis, engineering, and management challenging. It requires multi-disciplinary perspectives in research, such as computer science, economics, management, information systems, innovation sciences, engineering and policy. Given its economic and societal relevance, successful ecosystem research requires collaboration by scholars, practitioners, and individuals. The value and impact of engineered ecosystems is manifested through mobilization, participation, and facilitated collaboration enabling growth, innovation, and welfare.

One of the big challenges of the field is its multi-disciplinarity. It includes work about open source from software engineers, works about automotive platforms from management and information scientists, and works about visualizations from computer scientists. We reiterate that this domain is relevant from different perspectives, and should thus be considered multi-disciplinary.

The overview of challenges in Figure 2 functions as an inspiration for a future research agenda, which is currently under development.

References

- 1 S. Jansen, A. Finkelstein, and S. Brinkkemper. A sense of community: A research agenda for software ecosystems. In *Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on*, pages 187–190. IEEE, 2009.
- 2 Slinger Jansen, Michael A Cusumano, and Sjaak Brinkkemper. *Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry*. Edward Elgar Publishing, 2013.

4.2 Working Group on Software Business and Technology Lifecycle Artifacts

Sjaak Brinkkemper (Utrecht University)

Armin Heinzl (University of Mannheim)

Robert Heinrich (Karlsruhe Institute of Technology (KIT))

Alexander Maedche (Karlsruhe Institute of Technology (KIT))

Kari Smolander (Lappeenranta University of Technology)

License © Creative Commons BY 3.0 Unported license

© Sjaak Brinkkemper, Armin Heinzl, Robert Heinrich, Alexander Maedche, Kari Smolander

In Software-intensive Business many different artifacts along the software lifecycle are created. This covers both, business-oriented artifacts such as business models, roadmap, user stories or business process models as well as technology-oriented artifacts, such as technical architecture diagrams, class diagram or test cases is created.

Artifacts of the business domain depend on artifacts of the technology domain and vice versa. Furthermore, there are dependencies between artifacts used along the lifecycle of development, deployment and operations of software-intensive systems. Understanding and making these dependencies explicit in the entire development and management of large-scale software systems is important. However, currently there neither a classification of relevant artifacts nor an explicit description of their dependencies along the entire lifecycle available. Thus, there is no possibility to trace links between the artifacts and there are no comprehensive tools supporting navigating through different artifacts and propagating changes from one artifact to another. This leads to limiting focus on a small subset of artifacts and neglecting substantial side effects between the artifacts.

The working group addressed this important issues and suggested a set of activities in order to solve this problem. Specifically, an initial classification of existing artefacts (business and technology) covering the whole life-cycle was created. Furthermore, ideas were discussed on how to apply the classification, e.g. for monitoring development and operations, systematic life-cycle data collection, as well as interconnecting, tracing and optimizing of all artefacts. In a follow-up activity, the goal is to come up with a first conceptualization including a classification of artefacts and their dependencies in the field of Software-intensive Business. The conceptualization will formally be represented by (partial) metamodels and their composition. Initial prototypical tool support for visualizing and navigating through the conceptualization will be provided.

4.3 Working Group on Software-Intensive Business Research: Definition and Roadmap

Karl Werder (University of Cologne, Germany, werder@wiso.uni-koeln.de)

Sjaak Brinkkemper (Utrecht University – the Netherlands, s.brinkkemper@uu.nl)

Jan Bosch (Chalmers University of Technology – Sweden, jan@janbosch.com)

Michael A. Cusumano (MIT Sloan School of Management – USA, cusumano@mit.edu)

Georg Herzworm (University of Stuttgart – Germany, herzwurm@wius.bwi.uni-stuttgart.de)

Helena Holmström Olsson (Malmö University – Sweden, helena.holmstrom.olsson@mau.se)

Sami Hyrynsalmi (Tampere University of Technology – Finland, sami.hyrynsalmi@tut.fi)

Hans-Bernd Kittlaus (InnoTivum Consulting & ISPMA (International Software Product Management Association) – Germany, hbk@innotivum.de)

Thomas Kude (ESSEC Business School – France, kude@essec.edu)

Tiziana Margaria (University of Limerick – Ireland, tiziana.margaria@ue.ie)

Efi Papatheocharous (RISE SICS – Stockholm, SE, SE)

Diomidis Spinellis (Athens University of Economics and Business – Greece, dds@aueb.gr)

Pasi Tyrväinen (University of Jyväskylä – Finland, Pasi.Tyrvainen@jyu.fi)

Xiaofeng Wang (Free University of Bozen-Bolzano – Italy, xiaofeng.wang@unibz.it)

License © Creative Commons BY 3.0 Unported license

© Karl Werder, Sjaak Brinkkemper, Jan Bosch, Michael A. Cusumano, Georg Herzworm, Helena Holmström Olsson, Sami Hyrynsalmi, Hans-Bernd Kittlaus, Thomas Kude, Tiziana Margaria, Efi Papatheocharous, Diomidis Spinellis, Pasi Tyrväinen, Xiaofeng Wang

In today's digital environment companies are forced to participate in a process often coined digital transformation. As a result, companies expect to stay relevant and harness digital technologies for their competitive advantage and sustainable value creation. A central element of this transforming process is the software that companies develop, purchase or customize in order to support their business. These challenges stretch beyond the information and technology industry, as businesses use digital technology to compete in traditional industries. Popular examples are omnipresent, with Uber revolutionizing the taxi industry, AirBnB forcing new legislations to protect the hotel industry, and Spotify becoming a single source for music with a monthly subscription model. The Dagstuhl Seminar “Software Business, Platforms, and Ecosystems: Fundamentals of Software Production Research” organized by Pekka Abrahamsson, Jan Boch, Sjaak Brinkkemper, and Alexander Mädche took place from 29th of April until 02nd of May. The seminar's objectives were i) to strengthen cross-community research efforts, ii) to increase accessibility of research data and results, iii) to exchange on current and future research developments and discussions, iv) to initiate project ideas between scholars and with industry that evolve into project proposals.

4.3.1 Toward a definition

A Software-intensive Business creates, captures, and delivers value through digital technologies. Software-intensive Businesses create value through the development of new software technologies. When operating a platform, they often capture value through their established network of partner. When a software is shipped to and operated by a customer, the value is delivered. The academic community around Software-intensive Businesses investigates two aspects, i) arrangements and methods, and ii) responses to environmental changes. When investigating arrangements and methods, the community distinguishes between phenomena within and between organizations. Example organizations are software firms, start-ups, data

businesses and other Software-intensive Businesses. Within such organizations, product management, business models, agility, and DevOps are example arrangements and methods of interest. Between organizations, ecosystems, platforms, and OSS communities can be the subjects of investigation.

Given the manifold environmental changes a Software-intensive Business is subject to, the community researches three sources of change. First, the general and overarching trends and changes stemming from political, economical, societal, technological, environmental, and legal changes. Second, changes in their competitive environment, which may stem from a competitor dominating the market, employee shortages, or from characteristics of an industry segment. Third, customer trends, such as the digitalization, consumerization of information technology, and changing values lead to changing requirements. Hence, we formulate the following definition for Software-intensive Business research:

*The scientific field of **software-intensive businesses** studies sustainable software-based value creation, capture, and delivery a) through **arrangements and methods** i) within organizations (e.g. product management, business models, agility) (e.g. established, software firms, startups, data businesses, other firms), and ii) between organizations (e.g. ecosystems, platforms, app stores, OSS communities) and b) in response to **environmental changes** related to i) political, economical, societal, technological (e.g. cloud, IoT), environmental, and legal (regulation, GDPR, IPR); ii) competitive environment (e.g. market dominance, employee shortage, industry segments); and iii) customer trends (e.g. digitalization, consumerization, values).*

4.3.2 Toward a research agenda

As a result of the Dagstuhl seminar, the participants identified a 3x3 focus matrix. The x-axis of the matrix depicts the unit of analysis, i.e., a software system, a human system, or an ecosystem. The software system can be investigated as a whole or in parts, such as component or modules. The human system refers to a software organization, a development project or team team, or an individual, such as a developer or user. The y-axis represents the value, the lifecycle stages, and enablers we need to understand. Innovation can stem from technological innovation, business-driven innovation, or design innovation by creation new means of human-computer interaction in order to create and deliver more value. Innovation can also relate to the speed in which ideas and features are turned around, often referred to time-to-X, such as time-to-market, time-to-release, etc.. Lifecycle refers to the different stages a Software-intensive Business can find itself. Beginning with its inception as a startup, progressing toward a mature company, managing its rich ecosystem or responding to a crisis and need for a transformation. Enablers are prerequisites that help us to better understand the impact or facilitation of different factors, that some may call success factors.

	Software System	Human System	Ecosystem
Value	Deliver more value to its users	Reduce time-to-market, time-to-release	Provide value through an established network
Lifecycle Stages	From prototype to stable release	From student to expert	From beta to market dominance
Enablers	Knowledge base and research platform	Trainings, workshops and phd courses	Data mining and data analytics

Using this matrix, we suggest six areas that require further research in the future: 1) definition and reuse of core concepts, 2) Software-intensive Business lifecycle, 3) future business models, 4) benefits of new technologies, 5) driving innovation, and 6) enablers that support these research trends.

- **Definition and reuse of core concepts:** In order to advance our understanding of the nomological net in the field, core concepts need to be identified and reused. The community investigates concepts, such as ecosystems, platforms, development methods and tools, and product management, to name only a few. While the investigation of new concepts explore new research avenues for the community, the difference to established and better understood concepts needs to be clear. When investigating established concepts, the community progresses towards a deeper understanding of such concepts, their antecedents, outcomes, and boundary conditions. Hence, more research is needed that reflects the status quo in regards to core concepts of the community and simultaneously suggests the quo vadis for the research efforts of the investigated concept.
- **Software-intensive business lifecycle:** historically, the community centered around the term software business. Hence, it is not surprising that research shed more light on the business related activities. These were often limited to well-established software businesses in order to better understand how these differ from other businesses. While we have a better understanding of the differences and unique characteristics of the software business, more research is needed that investigates different lifecycle stages of these businesses. Example are research into software start-up, the effective management of platforms, or the management of crisis situations.
- **Future business models:** Traditional business model focused on the sale of a software license and the corresponding service. While there has been a major shift in the sales of software products towards a service oriented approach, as for example through software as a service. Further business models have evolved. For example the case of Uber, being a provider of a digital platform that users' approach in order to be linked with a taxi driver nearby. These examples suggest that the business models of Software-intensive Businesses keep changing as they are reinventing themselves. Hence, more research is needed in order to understand the driving forces behind these transitions and sometimes pivoting processes.
- **Benefits of new technologies:** Software technologies evolve at a astonishing rate. Internet and mobile technologies facilitated an increasing access and utilization of software. On the one hand internet technologies facilitated the introduction and use of software as a service concepts in which software installations become obsolete. Mobile devices result in the omnipresence of information technology in today's lifestyles and continuous access to messaging, social media, and finance applications. Yet, more recently, technologies such as machine learning, artificial intelligence, big data, internet of things and blockchain have been introduced amongst others. More research is needed in order to investigate their role in managing Software-intensive Businesses.
- **Driving innovation:** Given the increasing rate of change, software-intensive businesses need to find new ways to collect data, analyse such data, and derive meaningful conclusions. In response, continuous experimentation has been suggested in which the software provider tests different version of the software over time in order to analyse the data and to understand what works best. Little do we know about continuous experimentation with software and its possible extension to other subjects, such as business related decisions. Hence, more research is needed.

- **Enablers that support the trends:** Given these research trends to not exist in a vacuum, more research is needed that investigates enablers supporting these trends. For example, we need to understand whether existing measurement instruments still apply to these trends, If not, what adjustments to we need to make in order to assure reliable measurements? How can we assure that the next generation of Software-intensive Business scholars have the right skills and tools to progress the research with the rigor and granularity needed to advance the field?


4.4 Working Group on Health Measurement of Open Source Projects and Ecosystems

Slinger Jansen (Utrecht University)

Paul Grunbacher (Johannes Kepler University)

Efi Papatheocharous (RISE, Sweden)

Diomidis Spinellis (Athens University of Economics and Business – Greece, dds@aueb.gr)

License  Creative Commons BY 3.0 Unported license

© Slinger Jansen, Paul Grunbacher, Efi Papatheocharous, Diomidis Spinellis

Open source projects and ecosystems can be studied due to the public availability of their data. The main reasons for studying this data is to collect operationalizable metrics that can be used for the improvement of the project or ecosystem. We can for instance use these metrics to do prediction, study adoption rates, and perform scenario modeling.

Presently, in literature, the reigning health factors that are acknowledged are Robustness, Productivity, Niche creation. It is also common to look at ecosystem health from two dimensions: the partner/network level versus the system/project level. Each dimension provides a unique perspective on open source health and enables improvement in a different manner: one focuses on the activity within the platform, whereas the other focuses on the activity outside of it.

Typically, in open source ecosystem health research the metrics are characterized along several axes: they are evaluated for availability, collectability, generalizability, comparability, user friendliness, etc. Examples of metrics are interactions between developers, clones, branches, and numbers of commits. We also find that metrics that are typically easy to collect are not very meaningful. Also, the need arises for a meaningful compact subset of metrics, instead of throwing the kitchen sink at evaluation projects. Also, we suspect that “typical” developer behaviors can be extracted from the correlations between different metrics. Finally, we find that the goal-question-metric approach is insufficiently employed in the study of the health of ecosystems.

One of the bigger challenges in assessing ecosystem health is the myriad of perspectives on ecosystems. For instance, we can look at network health versus economic health. Furthermore, ecosystems themselves are made up of ecosystems, and we need to establish beforehand what the best manner is of decomposing an ecosystem.

4.5 Working Group on Research Data for Software Intensive Business

Slinger Jansen (Utrecht University)

Diomidis Spinellis (Athens University of Economics and Business – Greece, dds@aueb.gr)

Krzysztof Wnuk (Blekinge Institute of Technology, Greece)

License  Creative Commons BY 3.0 Unported license
© Slinger Jansen, Diomidis Spinellis, Krzysztof Wnuk

One of the largest challenges in the Software-intensive Business domain is the collection of data for research purposes. Mostly, the data is generated by proprietary entities, who are already challenged with the task of making the data publicly available. Software intensive businesses typically also do not see the use in sharing their data, as they reveal information about products, teams, and persons.

In intense collaborations organizations tend to be much more willing to share their data with the researcher, and often also with the entire research community. There are different ways to achieve this. First, researchers must avoid strategic topics for the company, such as new product strategy and infighting. Secondly, people within companies love sharing success stories. Thirdly, it is generally easy to anonymize the case data. Fourthly, it is possible to set up consortia in a distinct market, which enables collaboration between the companies and provides researchers with a trove of data. Finally, researchers can use historical data that is no longer problematic for the organization. There are also open data sources, such as economic meta-data, app stores, open source repositories, code review servers, continuous integration servers, and testing servers that provide data.

As a community we must find ways to curate data and publish it to colleagues. Furthermore, we need to provide researchers with incentives to publish case studies.

Participants

- Pekka Abrahamsson
University of Jyväskylä, FI
- Rahul C. Basole
Georgia Institute of Technology –
Atlanta, US
- Jan Bosch
Chalmers University of
Technology – Göteborg, SE
- Sjaak Brinkkemper
Utrecht University, NL
- Christoph Bussler
Oracle Labs – Redwood Shores,
US
- Michael Cusumano
MIT – Cambridge, US
- Jens Förderer
Universität Mannheim, DE
- Samuel A. Fricker
FH Nordwestschweiz – Windisch,
CH
- Paul Grünbacher
Johannes Kepler Universität
Linz, AT
- Robert Heinrich
KIT – Karlsruher Institut für
Technologie, DE
- Armin Heinzl
Universität Mannheim, DE
- Mika Helenius
TIVIA – Espoo, FI
- Georg Herzwurm
Universität Stuttgart, DE
- Helena Holmström Olsson
Malmö University, SE
- Sami Hyrynsalmi
Tampere University of
Technology, FI
- Slinger Jansen
Utrecht University, NL
- Zhi Jin
Peking University, CN
- Hans-Bernd Kittlaus
Inno Tivum Consulting –
Rheinbreitbach, DE
- Thomas Kude
ESSEC Business School – Cergy
Pontoise, FR
- Alexander Mädche
KIT – Karlsruher Institut für
Technologie, DE
- Tiziana Margaria
University of Limerick, IE
- Efi Papatheocharous
RISE SICS – Stockholm, SE, SE
- Balasubramaniam Ramesh
Georgia State University –
Atlanta, US
- Guenther Ruhe
University of Calgary, CA
- Kari Smolander
Aalto University, FI
- Diomidis Spinellis
Athens University of Economics
and Business, GR
- Pasi Tyrväinen
University of Jyväskylä, FI
- Xiaofeng Wang
Free University of Bozen-Bolzano,
IT
- Karl Werder
Universität Duisburg – Essen,
DE
- Krzysztof Wnuk
Blekinge Institute of Technology –
Karlskrona, SE

