Editor: **Tore Dybå**
SINTEF
tore.dyba@sintef.no

What Do We Know about Software Development in Startups?

Carmine Giardino, Michael Unterkalmsteiner, Nicolò Paternoster, Tony Gorschek, and Pekka Abrahamsson

STARTUPS ARE NEWLY created companies with little or no history of facing high volatility in technologies and markets. In the US alone, 476,000 new businesses are established each month,[1] accounting for nearly 20 percent of job creation.[2] As such, startups are an important factor in the economy. However, the environment of startups is dynamic, unpredictable, and even chaotic, forcing entrepreneurs to act quickly, fail fast, and learn faster to find a market niche and acquire a sustainable income. Sixty percent of startups don't survive the first five years, and 75 percent of venture capital funded startups fail.[3] Most of this is due to the high risk of startups, missed market windows, and other business reasons. To what extend engineering practices impact this high failure rate is still unknown given the premature state of research.

We present a detailed investigation and collection of all known empirical software engineering sources related to startups and their engineering practices, as well as an analysis of how accurate and reliable this available evidence is.[4] We see this as a first critical step into a largely unknown area—the world of software engineering practices in startups.

> High uncertainty and rapid evolution are the two key characteristics for startups.

What Is a Startup, Anyway?

In the past, the term "startup" had different meanings. Looking at the recurrent themes (Table 1 offers a complete list) adopted by researchers and practitioners, a startup is a small company exploring new business opportunities, working to solve a problem where the solution isn't well known and the market is highly volatile. Being newly founded does not in itself make a company a startup. High uncertainty and rapid evolution are the two key characteristics for startups retrieved by the studies, which better differentiate them from more established companies.

We retrieved and evaluated empirical evidence by using the systematic mapping study approach (see the sidebar).

Startup Software Development

"Done is better than perfect" and "move fast and break things" are slogans you might read when entering a startup workspace. What stands behind those slogans is a summary of more than 200 working practices. We reviewed these to point out where gaps exist and future development and research are warranted.

Process Management Is Agile, Evolutionary, and Opportunistic

Process management represents all the engineering activities used to manage product development in startups. Because the flexibility to accommodate frequent changes is essential in the startup context, agile

 0740-7459/14/$31.00 © 2014 IEEE

TABLE 1

Recurrent themes in software startups.

Theme	Description
Lack of resources	Economical, human, and physical resources are extremely limited.
Highly reactive	Startups are able to quickly react to changes in the underlying market, technologies, and product (compared to more established companies).
Innovation	Given the highly competitive ecosystem, startups need to focus on and explore highly innovative segments of the market.
Uncertainty	Startups deal with a highly uncertain ecosystem under different perspectives: market, product features, competition, people, and finance.
Rapidly evolving	Successful startups aim to grow and scale rapidly.
Time pressure	The environment often forces startups to release fast and to work under constant pressure (terms sheets, demo days, investors' requests).
Third-party dependency	Due to lack of resources, startups heavily rely on external solutions to build their product: external APIs, open source software, outsourcing, COTS, and so on.
Small team	Startups start with a small number of individuals.
One product	Company activities gravitate around one product/service only.
Low-experienced team	A good part of the development team is formed by people with less than five years of experience and often recently graduated students.
New company	The company has been recently created.
Full organization	Startups are usually founder-centric, and everyone in the company has big responsibilities, with no need for upper management.
Highly risky	The failure rate of startups is extremely high.
Not self-sustained	Especially in the early stage, startups need external funding to sustain their activities (venture capitalist, angel investments, personal funds, and so on).
Little working experience	The basis of an organizational culture isn't present initially.

methodologies have been considered the most viable process—they embrace change, allowing development to adapt to the business strategy.[5] Fast release with an iterative and incremental approach shortens the lead time from idea conception to production with fast deployment.

A variant to agile is the lean methodology,[6] which advocates the identification of the riskiest parts of a software business and provides a minimum viable product to systematically test and plan modification for the next iteration. In this regard, prototyping is essential to shorten the time to market.

To allow better prototyping activities, evolutionary workflows are needed to implement "soft-coded" solutions in the first phases until the optimal solution is found. Despite the number of methodologies that embrace fast prototyping in development, none of the processes are strictly followed by startups. Yet, the uncertainty and fast-changing needs of startups drive them to opportunistically tailor minimal process management to their short-term objectives and adapt to the fast-paced learning process of their users to address market uncertainty.

Software Development Is Driven by Customers who Act as Designers

Startups are under constant pressure to rapidly demonstrate that they're developing a solution that fixes a real problem.[7] They're constantly optimizing the problem/solution fit. To achieve it, startups must discover the real needs of their first customers, testing business speculations

EMPIRICAL BODY OF EVIDENCE

A systematic mapping study is a method to structure the empirical evidence in a particular field of interest.[1] We identified 43 studies that investigate different aspects of startups and their software development processes. We also estimated the strength of evidence in this field by assessing the rigor and relevance of the studies (see Figure A).[2] *Rigor* refers to the precision and thoroughness of reporting a study's design, validity threats, and results. *Relevance* refers to the realism of the environment in which the study is performed and to the potential of transferring results to practitioners.

Our rigor and relevance assessment suggests that the empirical evidence on the startup phenomenon is still rather premature. A minority—10 of the 43 mapped studies—provides transferable and reliable results to practitioners (sector A). Similarly, 10 studies provide low rigor and relevance (sector C). More studies (23) exhibit moderate industry relevance, but with low scientific rigor (sector B). From this observation, we conclude that it's challenging to conduct research in an environment in which a lack of resources is a dominant characteristic. Researchers need to identify efficient means to collaborate with and study startups.

References

1. K. Petersen et al., "Systematic Mapping Studies in Software Engineering," *Proc. 12th Int'l Conf. Evaluation and Assessment in Software Eng.* (EASE), 2007, pp. 1–10.
2. M. Ivarsson and T. Gorschek, "A Method for Evaluating Rigor and Industrial Relevance of Technology Evaluations," *Empirical Software Eng.*, vol. 16, no. 3, 2010, pp. 365–395.

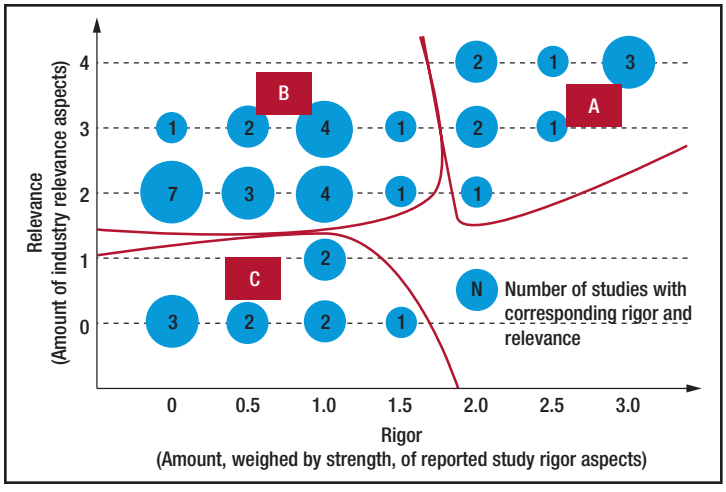

FIGURE A. Rigor and relevance of primary studies.

only by defining a minimal set of functional requirements.[8]

Several authors acknowledge the importance of involving the customer/user in the process of eliciting and prioritizing requirements according to their primary needs. However, the market-driven nature of those requirements also demands alternatives. For example, startups can use scenarios to identify requirements in the form of user stories and estimate the effort for each story. However, polishing requirements that address an unsolicited need a waste of effort. Requirements elicitation methods are moving toward testing the problem and understanding if the solution fits real needs before the product goes to market (the so-called customer development process).[7]

In the startup context, customers often steer requirements, and developers must be ready to embrace change from day one. The use of architecture and design patterns to make features modular and independent is crucial when functionality is continuously updated or removed. Therefore, employing architectural practices and frameworks that enable easy extension of the design can dramatically benefit the alignment between the product and market uncertainty.[9] This requires some upfront effort but can prevent the growth of product complexity.

Scientific evidence also points to the advantages of constant code refactoring. Reimplementing the whole system might be costly and risky if it must be immediately scalable to a growing number of users. Therefore, some quality assurance is needed for the functionality that brings the most value to customers. The use of ongoing customer acceptance through focus groups made up of early adopters can provide a time-efficient way to discover major bugs. But solutions are still scarce for easily accessible automated testing frameworks and the more practical user interface-testing approaches.

The Team Is the Catalyst of Development

Time pressure and lack of resources often lead startups to adopt a loose organizational structure without traditional management hierarchies.[10]

Empowerment of team members represents the main viable strategy for enhancing performance and success.[11] The team must be able to absorb and learn from trial and error quickly enough to adapt to new emergent practices. Working on innovative products requires creativity—an ability to adapt to new roles and face new challenges every day, working overtime if necessary.

Indeed, in building a startup company, the team needs expertise to counterbalance its lack of resources. In addition, having previous experience in similar business domains and exhibiting entrepreneurial characteristics (courage, enthusiasm, commitment, leadership) are important parts of a startup employee's skillset.

Nevertheless, the absence of structure might hinder important activities, such as sharing knowledge and team coordination, especially when the company grows. In this case, collocation is essential to facilitate informal communication and close interactions between team members.

Tools Can Accommodate Product and Management Changes

Startups can take advantage of the newest technologies and development tools without having to worry about legacy or previous working experiences.[12] But the selection of a technology requires some domain- or product-specific requirements, which are typically unknown in the early stages.

In general, startup employees prefer using those technologies that can quickly accommodate change in the product and its management.[13] Examples include general-purpose infrastructures, such as configuration management, problem reporting, tracking, and planning systems, and scheduling and notification systems. Easy-to-implement tools, such as whiteboards and technologies that can handle fast-paced changing information, will lower a startup's training and maintenance costs. To mitigate the lack of resources, startups often appear to take advantage of open source solutions when possible, which also give them access to a large pool of evaluators and evolving contributions.

Nevertheless, the absence of structure might hinder important activities.

Startup companies seek to generate revenue and obtain funding to continue the development, which means that software quality isn't their most critical concern. To quickly validate the product, they tend to use agile and lean methods in an ad hoc manner.[14]

Evidence suggests that engineering activities must be tailored to the startup context to allow flexibility and reactiveness in development workflows. Decision makers in startups confront continuous unpredictability; the relationship between cause and effect can only be perceived in retrospect.[15] Applying rigorous methodologies to control development activities isn't effective because no matter how much time is spent on analysis, it isn't possible to identify all the risks or accurately predict what practices are required to develop a product.

On the other hand, flexible and reactive methods designed to stimulate customer feedback increase the number of perspectives and solutions available to decision makers. Developers need the freedom to choose activities quickly, stop immediately when the results are wrong, fix the approach, and learn from previous failures. In line with the lean startup movement, we would expect methodologies and techniques tailored from common agile practices to specific startups' cultures and needs; failures should be completely acceptable or even preferred in favor of a faster learning process.

Reported common practices, which ride the wave of rapidly evolving technologies and markets, are as follows:

- use of well-known frameworks to quickly change the product according to market needs;
- use of evolutionary prototyping and experimentations via existing components;
- ongoing customer acceptance through early adopters' focus groups;
- continuous value delivery, focusing on core functionalities that engage paying customers;
- empowerment of teams to influence final outcomes;
- use of metrics to quickly learn from consumers' feedback and demand; and
- use of easy-to-implement tools to facilitate product development

and handle fast-paced, changing information.

Today's startups are at the forefront of applying new technologies in practice. The growing startup phenomenon opens uncharted opportunities as well as challenges in research. "Startuppers" need more transferable and reliable results concerning the diversity of context and viewpoints in the adoption of practices dealing with high uncertainty.

References

1. R.W. Fairlie, "Kauffman Index of Entrepreneurial Activity," Kauffman Foundation, 2014.
2. R.W. Fairlie, "State of Entrepreneurship Address," Kauffman Foundation, 2014.
3. C. Nobel, "Why Companies Fail, and How Their Founders Can Bounce Back," Harvard Business School, 2011.
4. N. Paternoster et al., "Software Development in Startup Companies: A Systematic Mapping Study," *Information and Software Technology*, 2014; DOI: 10.1016/j.infsof.2014.04.014
5. G. Coleman and R. O'Connor, "An Investigation into Software Development Process Formation in Software Start-ups," *J. Enterprise Information Management*, vol. 21, no. 6, 2008, pp. 633–648.
6. E. Ries, *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*, Crown Business, 2011.
7. S. Blank, "Why the Lean Start-Up Changes Everything," *Harvard Business Rev.*, vol. 91, no. 5, 2013, p. 64.
8. S.-C. Li, "The Role of Value Proposition and Value Co-Production in New Internet Startups: How New Venture e-Businesses Achieve Competitive Advantage," Portland Int'l Center for Management of Engineering and Technology (PICMET), 2007, pp. 1126 –1132.
9. S. Yogendra, "Aligning Business and Technology Strategies: A Comparison of Established and Start-up Business Contexts," *Proc. Int'l Eng. Management Conf.* (IEMC), 2002, pp. 2–7.
10. Y.-W. Yu et al., "Entrepreneurial Success for High-Tech Start-ups: Case Study of Taiwan High-Tech Companies," *Proc. 6th Int'l Conf. Innovative Mobile and Internet Services in Ubiquitous Computing*, 2012, pp. 933–937.
11. E. Carmel, "Time-to-Completion in Software Package Startups," *Proc. 27th Hawaii Int'l Conf. System Sciences*, 1994, pp. 498–507.
12. S.M. Sutton, "The Role of Process in Software Start-ups," *IEEE Software*, vol. 17, no. 4, 2000, pp. 33–39.
13. M. Crowne, "Why Software Product Startups Fail and What to Do about It," *Proc. Int'l Eng. Management Conf.* (IEMC), 2002, pp. 338–343.
14. S.W. Ambler, "Lessons in Agility from Internet-Based Development," *IEEE Software*, vol. 19, no. 2, 2002, 66–73.
15. C.F. Kurtz and D.J. Snowden, "The New Dynamics of Strategy: Sense-Making in a Complex and Complicated World," *IBM Systems J.*, vol. 42, no. 3, 2003, pp. 462–483.

CARMINE GIARDINO is a PhD student in computer science at the Free University of Bolzano. His research interests include software development in startup companies, focusing on the alignment between business strategies and development activities. Contact him at cgiardino@unibz.it.

MICHAEL UNTERKALMSTEINER is a PhD student in software engineering at the Blekinge Institute of Technology (BTH). His research interests include coordination of requirements engineering and testing, information retrieval to support decision making, and software repository mining. Unterkalmsteiner received an MSc in software engineering from BTH. Contact him at mun@bth.se.

NICOLÒ PATERNOSTER is a startupper at www.woodwallets.io. His research interests include bitcoins service development and makerspace development. Paternoster received an MSc in software engineering from Blekinge Institute of Technology and Free University of Bozen-Bolzano. Contact him at hi@adva.io.

TONY GORSCHEK is a professor at the Blekinge Institute of Technology (BTH) and Chalmers. His research interests include requirements engineering, technology and product management, process assessment and improvement, and practical innovation. Gorschek received a PhD in software engineering from BTH. Contact him at tgo@bth.se.

PEKKA ABRAHAMSSON is a full professor of computer science at Free University of Bozen-Bolzano. His research interests include empirical software engineering, agile development, startups, and cloud computing. Abrahamsson received a PhD in software engineering from University of Oulu in Finland. Contact him at pekka.abrahamsson@unibz.it.