Editor: **Christof Ebert**
Vector Consulting Services
christof.ebert@vector.com

Practices and Tools for Software Start-Ups

Gislaine Camila Lapasini Leal, Rafael Prikladnicki, Christof Ebert, Renato Balancieri, and Leandro Bento Pompermaier

From the Editor

Start-up companies continuously fight for survival. Driven by uncertainty, many decisions are taken ad hoc without analyzing tradeoff. While this allows flexibility, it also creates burdens for the future and technical debt, such as using the wrong tools and technology. Gislaine Camila Lapasini Leal, Rafael Prikladnicki, Renato Balancieri, Leandro Bento Pompermaier, and I have investigated start-up behaviors regarding which software development tools and practices they use. This article provides a summary and some guidance for start-ups. I look forward to hearing from both readers and prospective authors about this article and the technologies you want to know more about. —*Christof Ebert*

SOFTWARE START-UPS HAVE become increasingly important to the economy and are a key to innovation. However, these companies face many challenges, such as balancing short-term and future needs with respect to technology, processes, and tools.

A typical start-up follows three to four stages, namely: initiation, stabilization, growth, and an optional initial public offering.[1] Initiation is the time from the initial idea toward concept of the product and a first sale. The stabilization phase begins when a consumer receives the initial product and lasts until the moment the product is stable enough to be sold to a new consumer without causing overhead in product development. The growth phase shapes the prerequisites for stabilization and ends with a decision point toward being acquired or getting new capital and thus evolving to a normal small- or medium-sized enterprise.

Most start-ups terminate within the first two years for various reasons, mostly because the market is not behaving as it was hoped, the product was immature, or there were simply not enough competencies.[1,2] One hidden obstacle is within the development practices and tools, which not only hinder product evolution but might also create so much hassle that teams fight their software more than delivering new products. Sometimes the growth is too fast and necessary software decisions are postponed to a point where the product explodes. Netscape was an example of an overly successful start-up that eventually failed due to insufficient software practices and tools.[3]

To systematically distill practices for start-ups, we conducted an empirical field study. The starting point was a survey of Brazilian start-ups that had software development among their main activities. We selected the

Digital Object Identifier 10.1109/MS.2019.2946783
Date of current version: 24 December 2019

0740-7459/20©2020IEEE

Brazilian start-up scene because of its rich and diverse culture, with many new start-ups appearing each year, paired with large numbers of international collaborations. We obtained the list of start-ups through organizations such as the Brazilian Association of Start-ups and invited a total of 627 fledgling software enterprises to participate in this study using an online form for data collection. Over two months, data from 100 different start-ups were collected, which provides sufficient statistical significance for further analyses. Evaluating their respective distribution indicates that we have representative coverage. Figure 1 shows the year when the responding start-up was founded. As with any region, we can see the exponential growth of software start-ups during the past decade with dampening driven by regional economic slow-downs. We deliberately only looked to those surviving the first two years, that is, older than mid-2017, as the others would produce too much noise with their often very low maturity and lack of vision.

Of the companies participating in the survey, 26% were in the initial phase, while the remaining ones were divided into stabilization (25%) and growth (49%) phases. From the data collected, we found that in most cases start-ups take about a year to change from the initial to the stabilization phase and an average of six years to move into the growth phase. The survey indicated that 54% of the start-ups are in an innovation ecosystem, in other words, an incubator, a coworking space, an accelerator, or a science and technology park, and 34% of the start-ups mentioned having external investors.

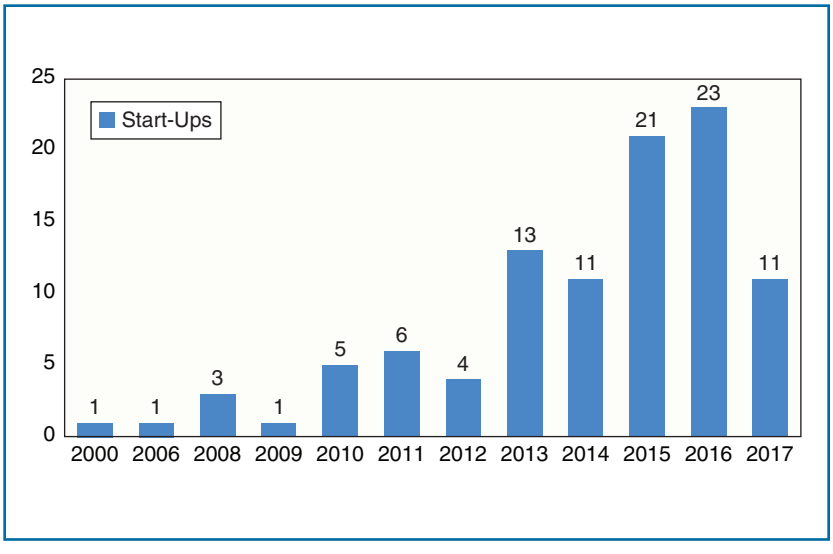

FIGURE 1. The number of responding start-ups founded per year.

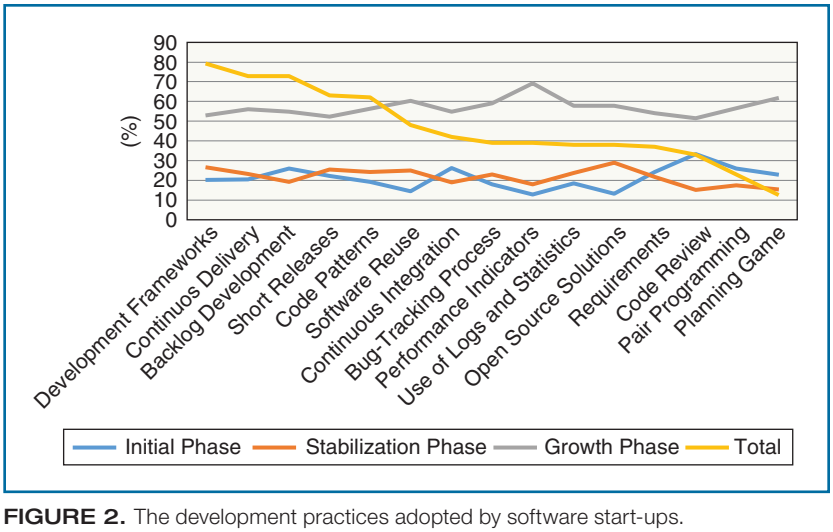

FIGURE 2. The development practices adopted by software start-ups.

Software Start-up Survey

As to be expected, a clear majority of 86% of the start-ups indicated that they have had to adopt agile methodologies into their software development process. Agile methodologies allow for delivering value to the customer in a shortened development cycle with constant customer feedback. This is corroborated by a recent study that mentioned more than 70% of software start-ups adopt agile methodologies.[3]

Figure 2 presents some software development practices adopted by software start-ups. This figure shows the total citation of each software development practice and the percentage of each practice adopted based on its phase. For example, the use of development frameworks was mentioned by 79 (79%) of those surveyed. However, 42 of those 79 start-ups (53%) are in the growth

phase, while 21 (27%) are in the stabilization phase, and 16 (20%) are in the initial phase.

Start-ups like utilizing agile methodologies, as it is the best method to survive in a volatile, uncertain, complex, and ambiguous (VUCA) world. Scrum can be easily adapted and scaled by embedding the existing processes that cover functional safety and systems engineering, software architecture, and testing in the Scrum team. The risk of ad hoc agile is high, and Vector Consulting faced many client projects in which agile was seen as "anything goes," which is a deadly sin, specifically for start-ups. Thus, rules for critical systems in Scrum should be developed and strictly applied. "Definition of done" should be the only exit criteria which is in lieu of the process expectations.[1] Team meetings are key to success to agile, so transparency and participation should be ensured. To ensure consistency of products across domains, teams should be centered around architectures and integration. Integration focus allows new ventures to scale the approach across the distributed teams and sites.

Start-ups like utilizing agile methodologies, as it is the best method to survive in a volatile world.

We argue that the use of frameworks is among the most common practices because it streamlines the development process by removing the need to implement something that has already been made. Literature mentions that this is one of the most important practices for start-up success, especially for those that are in the initial phase, since it allows more time to be invested in the feature development of the product, reducing the time to market.[4] However, most of the start-ups adopt this practice in more mature phases.

Continuous delivery is also frequently used as it allows for frequent delivery and the customer feedback in shorter cycles. Backlog development is also a practice frequently mentioned and well described in the Scrum framework. It allows a team to list all desirable features of the product, allowing the development team to keep an up-to-date record of the prioritized features and break in the tasks to be developed during each iteration.

Short release cycles were mentioned by 63% of the start-ups and is corroborated by a recent large survey of 1,526 software start-ups that showed almost the same adoption percentage for this practice.[5] The same study also corroborates the fact that speed-related practices such as short release and continuous delivery are used more often than quality practices, such as code review and pair programming.

Let us further drill down to specific tools as we evaluated with start-ups (Table 1). Usability test is the most commonly used validation (mentioned by 59% of the start-ups), followed by user test (49%), unit test (44%), acceptance test (38%), and integration test (37%). Usability tests and user tests allow a rich feedback from the customer. Unit testing allows one to verify that the system meets the requirements that were defined during the specification, as well as acceptance tests. Integration tests usually occur after the unit tests and allow verification if the modules or parts of the system will work when combined. However, when looking at the general responses, the various tests were adopted by between 40 and 60% of the surveyed start-ups, depending on the type of test, and this could be risky for the success of such endeavors.

As for version and configuration control tools, GitHub (48%) and Bitbucket (44%) are the most used, followed by GitLab (17%), Subversion (13%), and other tools (10%). Several start-ups use more than one tool. Only one start-up indicated it was not using version and configuration control tools. It is worth noting that all version control tools are decentralized, which allows first-class access to all developers. In other words, it is not necessary to be a direct contributor to access the versions. Atomicity in changes and an automatic way of merging them are also advantages that help to maintain the consistency of the project, besides preventing the occurrence of errors and outdated code in any of the workstations.

Regarding project management, Trello is the most popular tool, adopted by 66% of the surveyed start-ups, followed by Atlassian Jira (19%) and Redmine (9%). There were several other tools indicated, varying from Google Docs to specific agile project management tools such as ScrumHalf.

Development platforms vary from start-up to start-up. In our survey, respondents cited Visual Studio, Eclipse, Xcode, JetBrains, NetBeans, and PyCharm explicitly. More than half of the respondents mentioned other platforms, including Atom, RadStudio, Android Studio, Unity 3D, and Rstudio, among others. It is difficult to

argue about the use of certain suites or development platforms, since the choice usually depends on particular factors of the organization or even the developer.

Depending on the programming language used by the start-up, only a few specific tools can be used, and even within these options, the choice of tool can vary according to the personal preference of each developer. Budget constraints and target audience can also influence decision making. In the case of the Xcode tool, for example, there are no other options if the focus of the start-up is on software development for Apple products. Similarly, PyCharm is restricted to the Python programming language.

Testing is vastly dominated by unit test, such as Junit and others, including some automation with continuous integrations pipelines, like Jenkins. Test methodology is not so advanced, primarily looking to functional testing. Such positive methods aim to ensure that all of the functional requirements of the system are taken care of. Negative testing methods that look to quality requirements are not as common. Quality requirements, for example safety and cybersecurity, are typically implicit and are not explicitly specified.

> Test methodology is not so advanced, primarily looking to functional testing.

For knowledge management we saw the mention of Atlassian Confluence, Freshdesk Collective Knowledge, and OpenKM. The number of start-ups that do not use such tools is relatively high, however. Several other tools have been indicated for this category. Storing the software knowledge allows a company to maintain a history of product development, allowing it to visualize the state of its features over time. The opportunity to look at past decisions is important for making future ones, since they can indicate what was expected of the product and what direction it is currently taking. From this, adjustments can be made to bring the product back to its original path or to insert improvements to ensure the start-up meets the needs and demands of its

Table 1. A brief overview of some of the low-cost tools used by start-ups in engineering and team collaboration.

Tool	Focus	URL	Agile support	Collaboration support	Cost (License)	Drag/ drop	HW and SW	Software engineers	Systems engineers
Papyrus	Modeling	http://www.eclipse.org/modeling/mdt/papyrus/	Yes	Yes	Low (OSS)	High traceability	Both	Yes	Yes
Enterprise Architect	Modeling	https://sparxsystems.com/	Yes	No	High (proprietary)	Low traceability	SW only	Yes	Yes
Jira	CM, tracking	https://www.atlassian.com/software/jira	Yes	Yes	Low (proprietary)	No traceability	SW only	Yes	Yes
Pivotal Tracker	Story templates	https://www.pivotaltracker.com	Yes	Yes	Low (proprietary)	Little traceability	SW only	Yes	Yes
Capella	Sytem design.	http://www.polarsys.org/capella/	Yes	Yes	Low (OSS)	High traceability	Both	No	Yes
CORE	Complex system models	http://www.vitechcorp.com/	Yes	Yes	Low (OSS)	Medium traceability	Both	Yes	Yes
Modelio	Modeling	http://www.modeliosoft.com/	Yes	Yes	Low (OSS)	Low traceability	SW only	Yes	Yes

HW: hardware; SW: software; OSS: open source software.

customers. Yet, despite their benefits, there are still many start-ups that do not adopt such tools.

> Storing the software knowledge allows a company to maintain a history of product development.

Where Do We Go From Here?

The number of start-ups is steadily growing. Their practices are mostly agile but not necessarily aligned with a sustainable agile method and tool support. Software engineering practices and tools developed originally with the focus on bigger organizations are often too heavy for start-ups, leading to less systematic ways of utilizing them.

Collaboration matters. Good tools help to keep track the of a project and organize the team's progress. But they are not there to impose strict schedules or roles, only to provide support, making self-management and convergence of goals easier for the distributed developers by providing the programmers with tracking ability while working collaboratively on these parts. Often a project is split up into short cycles which helps to gradually converge onto the result. Cycles swing between planning and code sprints. While the goal is to keep the cycle short, including a good amount of developer's feedback while planning allows the distributed teams to adjust and focus.

Start-ups work with agile methodologies and need fast progress tracking. Here the graphic dashboards of

ABOUT THE AUTHORS

GISLAINE CAMILA LAPASINI LEAL is an adjunct professor in the Production Engineering Department at Maringá State University, Paraná, Brazil. She is also part of the postgraduate programs in Computer Science and Production Engineering at the same university. Contact her at gclleal@uem.br.

RENATO BALANCIERI is an adjunct professor in the Computer Science Department at the Paraná State University in Apucarana, Brazil. He is also part of the postgraduate program in computer science at Maringá State University. Contact him at renato.balancieri@unespar.edu.br.

RAFAEL PRIKLADNICKI is an associate professor at the School of Technology and the director of the Science and Technology Park (Tecnopuc) at the Pontifical Catholic University of Rio Grande do Sul, Porto Alegre, Brazil. He is also the chair of the *IEEE Software* Advisory Board. Contact him at rafaelp@pucrs.br.

LEANDRO BENTO POMPERMAIER is an assistant professor in the School of Technology at the Pontifical Catholic University of Rio Grande do Sul and leads the start-ups area at Tecnopuc. Contact him at leandro.pompermaier@pucrs.br.

CHRISTOF EBERT is the managing director at Vector Consulting Services. He is a Senior Member of the IEEE and is the editor of the "Software Technology" department in *IEEE Software*. Contact him at christof.ebert@vector.com.

most tools help represent the progress of the teams and achievement of the goals. Allowing the teams to follow each other's progress visually, makes them better able to stay on track and avoid conflicts.

Start-ups have so many topics at hand—most prominently to survive in business—that they often lack the courage to select the tool most appropriate to the process and needs of the company. Different engineers have different requirements, and in new ventures there is often no clear decision-making process. The number of tools is just overwhelming, and they pick what they know, not necessarily what is best. Our recommendation regarding tools thus is to focus on export mechanisms which are key to migrate at a laser point to another environment. We also recommend avoiding products known as *white elephants*, often promised by big vendors, as they can be expensive and difficult to get rid of. Our recommendation is to federate tools and ensure transparency of content to all software engineers. Federation-like integration allows not only to connect tools with different technologies, and eventually replace some pieces, but also to connect with changing teams and ecosystems, as is normal in the evolution of a start-up. Much can be learned from the Bauhaus design school that currently celebrates its 100th birthday in cities like Weimar, Tel Aviv, and New York. At the beginning of the 20th century, they had already understood the concept that "cooperation rules the world." The ability and willingness to cooperate while using structure and sharing knowledge should be our ambition in ramping-up tools and practices for start-ups.

> The opportunity to look at past decisions is important for making future ones.

Acknowledgments

We thank Bruno Henrique Cavalcante, Rafael Chanin, Abhishek Ashok Nilugal, and Hammad Shahid for their contributions to this article. Vector also wants to thank several start-up clients who provided insight to their projects. This work is partially funded by the Foundation for Research Support of the State of Rio Grande do Sul (17/2551-0001/205-4) and the Brazilian National Council for Scientific and Technological Development.

References

1. M. Crowne, "Why software product startups fail and what to do about it? Evolution of software product development in startup companies," in *Proc. 2002 IEEE Int. Engineering Management Conf. IEMC'02*, pp. 338–343.
2. E. Mkpojiogu, N. L. Hashim, A. Al-Sakkaf, and A. Hussain, "Software startups: Motivations for agile adoption," *Int. J. Innovative Technol. Exploring Eng.*, vol. 8, no. 8S, pp. 454–459, June 2019.
3. I. Cat and C. Ebert, "Technical debt as a meaningful metaphor for code quality," *IEEE Softw.*, vol. 29, no. 6, pp. 52–55, Nov. 2012.
4. C. Giardino, M. Unterkalmsteiner, N. Paternoster, T. Gorschek, and P. Abrahamsson, "What do we know about software development in start-ups?" *IEEE Softw.*, vol. 31, no. 5, pp. 28–32, 2014.
5. J. Pantiuchina, M. Mondini, D. Khanna, X. Wang, and P. Abrahamsson, "Are software startups applying agile practices? The state of the practice from a large survey," in *Agile Processes in Software Engineering and Extreme Programming. XP 2017. Lecture Notes in Business Information Processing*, vol. 283, H. Baumeister, H. Lichter, and M. Riebisch, Eds. New York: Springer-Verlag, 2017, pp. 167–183.