Early-Stage Software Startup Patterns
Strategies to building high-tech software companies from scratch

Daniel Cukier, University of São Paulo – Department of Computer Science
Fabio Kon, University of São Paulo – Department of Computer Science

This pattern language describes different strategies to build an innovative software startup. We propose patterns to address multiple problems that an early-stage startup has to face. While some patterns are more related to technical issues, other focus on human and management aspects of new, creative software companies. The pattern language can be used as a set of tools for high-tech entrepreneurs.

Categories and Subject Descriptors: K.6.3 [**Computing Milieux**]: Management of computing and information systems—*Software Developement*

General Terms: Startup Patterns

Additional Key Words and Phrases: Startups, entrepreneurship, Startup pattern language

ACM Reference Format:

Cukier, D. and Kon, F. 2015. Early-Stage Software Startup Patterns: Strategies to building high-tech software companies from scratch. HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 22 jn , , Article (October 2015), 11 pages.

1. INTRODUCTION

Search for known solutions to known startup problems in the Software Startup Patterns catalog.

In 2014, Daniel started a new company with a friend. He had some experience in working as CTO in another startup and also as technical manager in different companies. He also had a Masters in Computer Science, knowledge on how to build web systems, and had read several books on Entrepreneurship. Even with all this experience and technical expertise, they had a lot of difficulties to build their own company. Many times, Daniel found himself in situations where he did not have any idea of how to act or what decision to make. They had to learn a lot of new Startup concepts. Certainly, they were passing through experiences similar to those faced by many other entrepreneurs in the past (and to be faced in the future). They wish they could have somewhere to look and learn known solutions to their problems.

A startup is an "organization formed to search for a repeatable and scalable business model" [Blank and Dorf 2012]. Besides Startups exist in many different industries, we focus this article on Startups whose main product and revenue streams are based on software. These organizations are run by open-to-risk entrepreneurs who, along years of Startup creation, pass through many difficulties in fields where they are not specialists. The range

Author's address: Daniel Cukier and Fabio Kon, Deparment of Computer Science, University of São Paulo, Rua do Matão, 1010, Cidade Universitária, São Paulo, SP, 05508-090, Brazil; email: danicuki@ime.usp.br, kon@ime.usp.br

of topics go from simple problems like “where the company’s office will be?” to decisions of “how much equity do we give to employees?” or “which operating system do developers should adopt?”

Startups are usually located within a context of an Ecosystem [Kon et al. 2014]. There are many examples of ecosystems in different stages of development, such as Israel [Senor and Singer 2011], New York [Cometto and Piol 2013], or many other [Frenkel and Maital 2014]. By identifying opportunities in the market, an entrepreneur creates a Startup. The entrepreneur gets support from family (and friends), who are part of a society and culture that influence the entrepreneur behavior. Universities and research centers provide knowledge in technologies that enable the Startup, by preparing the entrepreneur and providing him networking possibilities. Universities or established companies may run incubators or accelerators that trains and instruments the startup with methodologies. Private funding bodies like angels, VCs mentor and invest on Startups, which can also get financial resources from governmental programs through R&D funding agencies or tax incentives. Within a complex environment, startup owners face a range of different challenges and problems.

Startups have a lot of problems to solve along their life and it is difficult for entrepreneurs to find answers to all questions by themselves.

Low budget, time pressure, few resources available and instability are every-day problems to Startup entrepreneurs. Moreover, since these organizations are very small in the beginning, there are not many people inside the organization for who entrepreneurs can ask for help.

Therefore:

Use the startup pattern language to look for known solutions for startups problems.

The pattern language organizes, in a structured catalog, common problems that entrepreneurs face and help newcomers to apply known solutions to known problems in the same context. The pattern language also shows the relationships among the patterns and guides practitioners on which pattern to apply next. By using patterns to solve common problems, entrepreneurs can reduce waste of time, which is the most precious asset they have.

The patterns language can be visualized in Figure 1. Patterns more related to human aspects of Software Startups are located on the top, while other patterns more connected to technical characteristics lay on bottom. The crosscutting patterns which tackle with both human aspects and technical are in the middle of Y-axis.

The X-axis is related to the startup lifetime. Some patterns can be applied in the first days of the Startup **Ideation and Concept**. This is the initial phase when the team is not defined and business model is not clear. A later moment in the Startup existence is when the team is already **committed** and the business model is **validated**, with some customers and confirmed revenue. Then the Startup starts to show a clear and measurable grow, entering in a **Scaling** and **Establishment** phase to become successful organization. Other patterns suit better on these later stage phases.

The present Pattern Language is not the first attempt to create a Pattern Language for Software Startups, as we can see in [Eloranta 2014], [Hokkanen 2015] and [Goldman and Gabriel 2005]. However, the patterns presented here original in the literature until now. New patterns can be added to the catalog as long as entrepreneurs understand that their problems are not unique to their reality and that, by sharing their experiences in an accessible and easy to use way, they can contribute to the evolution of the startup ecosystem they live in.

Due to space limitation, this article will describe only four patterns, leaving other patterns for future work.

[1] http://www.startupcommons.org/startup-key-stages.html

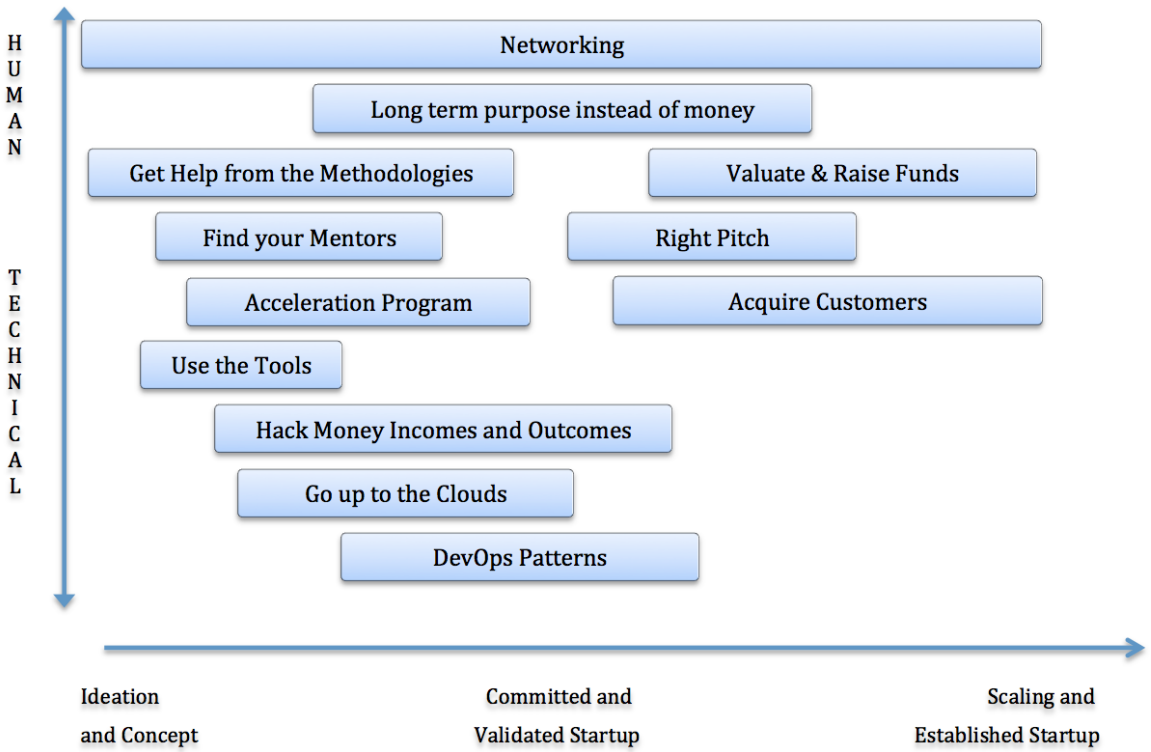

Fig. 1. Early Stage Software Startups Pattern Language

2. PATTERNS

2.1 Get help from the methodologies

Use existing methodologies that promote entrepreneurship best practices and will make you less likely to commit the most obvious mistakes.

A successful Israeli startup in the field of Web analytics started their operations by focusing only on the algorithms they used. While the technology was doing well, the company was not growing until they decided to work systematically with the business aspects by adopting a more disciplined way of looking at their market. Then, business grew and they were forced to hire more programmers. At that moment, they realized that their way of developing software wouldn't scale and would produce very bad quality code. Then, they adopted Scrum to manage their development team and, in less than two months, they saw big improvements in their productivity.

In the past decades, practitioners have learned, in the hard way, that there are many recurrent mistakes made by entrepreneurs such as ignoring the product-market fit and focusing only on technology, forgetting to look at the business side of things. Some of these experienced practitioners became educators and consultants and started to write down their experience in the form of books defining methodologies that startups can follow to minimize mistakes.

While, absolutely no methodology or technology is failure-proof and we know there are no silver bullets [Brooks 1987], they can help entrepreneurs avoid some of the common mistakes. As an entrepreneur, you should study and

Fig. 2. Startups working in the acceleration program in University of São Paulo

be aware of existing methodologies such as Customer Development by Steve Blank [Blank and Dorf 2012], Lean Startup [Ries 2011], Disciplined Entrepreneurship [Aulet 2013]. Pick the one you like the most or a combination of them and follow it in your daily startup activities.

Specifically in the case of Software Startups, it is fundamental to adopt a software development method that can deal well with extreme uncertainty and requirements that can change in a daily basis. Agile Methods such as Extreme Programming [Beck 2000] and Scrum [Schwaber 2009] were designed for such situations and have been used successfully by thousands of companies world-wide.

Inexperienced entrepreneurs tend to make the same mistakes over and over again, repeating the same flaws that led to the failure of other startups that they are not even aware of.

Some young entrepreneurs tend to be a bit arrogant and believe that they know everything and don't need to learn from others. In rare cases, the same problem occurs with older professionals; although this is much more infrequent as more experienced professionals normally learned the hard way that everyone needs help from others. Studying existing methodologies sometimes can be boring and requires a lot of investment in reading multiple books, attending courses, and enrolling in acceleration programs. However, this investment is small when compared to the great benefits that arise from developing the business in a more systematic way.

Therefore:

Invest time and energy to study the well-known startup development and agile software development methodologies and follow, in your daily work, the best practices championed by these methods.

Everybody in a startup founding team should know very well at least one modern method for startup development, be it Customer Development, Lean Startup, Disciplined Entrepreneurship, etc. To acquire this knowledge, it is important not only to read books describing the methods and their consequences but also attending courses with

practical activities and exercises. Finally, to really absorb the ideas it is fundamental to apply their concepts in real situations with real problems. By simply reading a book, we learn a little; but it is only when we actually apply the methods in real situations in our own context that we really incorporate the new knowledge. It also helps to interact with other people in similar situations via online forums, meetups, and study groups.

It is important to note that there are significant limitations on what this pattern can achieve, though. Learning from others' mistakes is helpful but won't make the entrepreneur failure-proof. Many times the entrepreneur faces new situations that he/she cannot relate to what is in the books. The context changes, the situations change, and the parallel with what the methodologies teach is not clear. Thus, even the most "well-educated" entrepreneur will make mistakes and fail many times. But the best learning comes exactly from one's own mistakes. Reading about a mistake in a book is one thing, but feeling on one's own skin the pain of making a wrong decision is something an entrepreneur will never forget in his/her entire life.

Agile software development requires a unique programming culture that is radically different from what used to be taught at universities in Computer Science and Computer Engineering courses up to the late 1990s. Unfortunately, this old culture is still common in thousands of software development companies nowadays. Fortunately, a significant portion of the new generation of software developers is being educated with a more agile mindset based on strong collaboration with customers, test-driven development, incremental design and development, and continuous adaptation to changing conditions and requirements. A software startup is by definition an environment of high uncertainty where requirements change very rapidly, thus a robust agile development method is essential. Startup founders must embrace agile methods, hire programmers with this culture whenever possible and invest on training and education when the required level of adherence to agile practices is not observed. It is fundamental that a CTO proficient in agile methodologies guide the startup software development efforts.

The level of rigor and number of agile practices that must be applied vary largely across the lifetime of a software startup. In its very early stages, when there are only one or two developers, very little discipline is required; maybe only practices such as incremental development and customer collaboration are really helpful. When the startup grows to 4 to 6 developers, practices such as automated testing, pair programming, refactoring, stand-up meetings become more important. When the startup goes beyond 10 developers, a larger set of agile practices and a more disciplined organization become very important to manage properly the technical debt [Brown et al. 2010] [Kruchten et al. 2012] of the product under development.

Also, entrepreneurs must remember that, many times, buying off-the-shelf software components and reusing available open source software is much more cost-effective and help reduce time-to-market. Thus, a startup should avoid developing software from scratch when there is no need for it because an existing solution can be reused.

A startup that applies these disciplined principles both for developing its business and for its software will make less mistakes over the course of its life and will be more likely to succeed. A systematic approach for finding the right markets and developing the software that really addresses a real pain of the customer will help the startup in finding the product-market fit more rapidly. This does not guarantee that the founders will not make mistakes and that, even if they don't, that the company will necessarily succeed. But, even in this case, if the business "need" to fail, it is better that it fails as soon as possible, preferably before spending large amounts of money and energy in investment.

2.2 Hack money incomes and outcomes

Find creative ways to avoid unnecessary costs and get as much free resources as you can.

In the beginning of our startup, we didn't have an office. My partner and I worked from our home, so we did not have to pay for an office space. But after some time, we were starting to grow the development team. We needed a place to meet with everybody. We rented a place at a very low cost. We tried to find co-working offices, but in our region it would cost more than having our own space. Then we needed to buy furniture. We discovered that new

Fig. 3. The day we bought furniture in an auction website

furniture prices were very expensive. We found a great alternative: asked some relatives for donation and also looked in auction websites. We could set up everything we needed for the office with less than US$500.

"In our company we have policy that you can spend company's money on whatever you like, but you should treat it as it was yours. If you need something, it is ok to buy it, if you would pay for it yourself, too. It is kind of test that your really need that." – Veli-Pekka (entrepreneur).

Startups usually have very little money. In the beginning, they do not have revenue streams and need to find ways to have money for their first month's costs. The challenge is to keep costs as low as possible and the incomes sufficient for these costs. Furthermore:

There is no revenue in the beginning of startups and costs are high. It is a challenge to know exactly where to spend the few resources a startup has.

Spending money wisely is as important and earning money as fast as possible. The best uses for money in a startup are those related to developing the product and marketing. Administrative costs must be as low as possible. Salaries also should be as low as possible (see Section 2.3). Even if the startup has received an investment, spending the money on the right things can be the difference between success and failure.

Therefore:

Eliminate unnecessary costs and get free or cheap resources as much as you can.

There are many ways to save money in your startups. Moreover, often there are great alternatives in your ecosystem to raise inexpensive resources. By focusing your expenses on essential things such as product development, customer development, and marketing, you maximize your resources and accelerate the business

model implementation, anticipating revenue and increasing your chances of success. Furthermore, if you use "free" or "cheap" money offers, you are in advantage compared to your competitors.

Here are some ideas:

—If some people work from home, you don't need to have a big office (or even have an office).

—You also won't have to waste money with equipment and furniture.

—In some cities there are free co-working spaces for startups, use them!

—Save money by offering equity shares instead of high salaries (see Section 2.3).

—Buy used equipment and furniture at auctions websites

—Search for government or universities programs that foster startups development (ex: startup Chile, Startup Brazil, PIPE-FAPESP, USA NSF-ISBIR, Israel OCS, etc.)

—Use cloud providers programs for startups to save costs with infrastructure. Almost all providers such as Google, Rackspace, Microsoft, Amazon, and others offer free tier until US$100.000 for one year.

2.3 Long term purpose instead of money

Differentiate from big companies to attract talents.

When Daniel started the company with his partner Juliano, they did not have money to pay high salaries for tech talent in São Paulo, one of the most expensive cities in Brazil. Their startup was a high-tech innovative platform in the music industry, so they needed the best developers to create complicated algorithms. On a first try, they offered bellow average salaries, but even if some developers were interested in the company's challenges, they did not accept the job. When the founders started to offer equity, they attracted exactly the people they wanted to their team: people with passion and long-term commitment with the company. Moreover, people willing to give up high salaries in exchange of being part of the company's construction and purpose.

When working as CTO at Elo7, a successful Brazilian startup, Daniel organized a developer's retreat, a week when the engineers would work in a relaxed environment, out of the office. They had pool, sauna, video games and food at their disposal, as well as a working table and Internet access. Besides some communication problems with the main office, the workweek was very productive and the team members had a lot of fun, feeling very connected and engaged after this special and unforgettable moment.

Most startups, especially in the beginning of its existence, do not have resources to pay a salary compatible with market values. Startups need to have other ways to attract talent and motivate people to work on its project. Furthermore, hiring the right people in the early days is crucial for the Startup culture development. The criteria for hiring are very talented people with passion and commitment for the business.

It is very hard to hire talents when you are competing with big companies admired by people.

Big companies have dedicated staff for recruiting and they usually offer high salaries and many benefits to employees. Moreover, known high-tech companies have are admired by people for their achievements. On the other hand, employees in big companies can feel themselves no one in the middle of thousands. They are restricted to companies existing rules and rigid processes.

The first substitute to money is purpose. Startups want to change the world for the better. Show your vision to candidates and let them dream with you. People want to feel pride of their work and like to be part of a big thing.

Another attractive in Startups is the growing potential. In big companies, employees do not have the chance to grow and evolve in career as fast as they can in a Startup. The first Startup employees have the chance to

Fig. 4. Developers working at the swimming pool in a developer's retreat

become managers or directors in a matter of a couple of years. They have not only the chance to grow, but also the freedom to dictate the rules and build the company from scratch following their own beliefs. In terms of professional growth, young Startups tends to be more attractive, besides being more risky.

Beside Startups do not have money in the beginning, they can offer the opportunity to earn a lot of money in future, if the business succeed. Startup founders, especially those without investment, can reserve pool of the company stocks and give them to the first employees.

Therefore:

Offer company equity, freedom and job with purpose in exchange of high salaries.

When you offer engagement on a common purpose instead of money, you attract the right people to work with. Workers will behave like founders and will give much more energy to the startup. D. Pink argues that people are mainly motivated by autonomy, mastery and purpose [Pink 2009]. You can offer all three in your Startup. If you are creating an innovative product that will change the world for the better, you have purpose. When you have a participative culture and flexible work environment, you offer autonomy. Startups with dynamic and learning environment permit people to master on their field. You can also offer flexible work schedules or home office days.

To convince candidates to accept equity and motivate them, you need to show the benefits of this option in the interviews process. The interviews must be very engaging. You must show passion and belief on the project. Candidates must feel that you are reliable and strong. In these moments, your reputation counts a lot. Networking (see Section 2.5) is very useful not only to help you to build the reputation, but also to find potential talents in your local community.

If you want to attract people interested in Startups, post job announcements on forums or groups that you believe are aware of the Startup environment. Express your enthusiasm in the job post text. Make it clear that you are looking for people that want to work in startups, people who understand and like this environment. This will save you a lot of time, not only on explaining the Startup context in interviews but also by keeping away the wrong candidates.

2.4 Go up to the cloud

Prioritizing existing cloud solutions for features non-core to your business

When Daniel started to work in Elo7, the company website infrastructure was installed in two bare metal servers. Local SMTP server application was responsible for mail delivery. Server logs were stored in local drive. Managing and scaling bare metal servers was difficult, so everything was migrated to cloud instances. The mail delivery was decoupled from the application server and migrated to a service provider. Server logs were also migrated to a cloud PaaS, so developers would not need to log in the application servers to search in log files. These and other changes to the cloud lead to a lower cost and more flexible systems architecture, enabling the Startup to grow fast and solid.

Startups do not have time to build their own physical server infrastructure. They need to focus on their own product or service development. Nowadays, there are a lot of services that provide almost everything a Startup needs to setup its business online. There are hundreds of offerings on Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) layers. Moreover, most providers offer very low costs for low usage, as well as free tiers for beginners.

Today it is possible to build a technology company just by sticking together existing solutions. There are thousands of different Software as a Service (SaaS) platforms, most of them offering free tier for small companies to solve a wide range of problems a company has, from file sharing applications to accounting software and communications tools. Almost everything is available freely online.

Software running into production is much more complex than running on developers machines. There are many cloud providers that offer simple solutions to deploy software in a mater of a mouse click, without needing any knowledge on system administrations. These Platform as a Service (PaaS) environments save a lot of precious time for startups, which in the beginning don't have any specific requirement that justifies building its own infrastructure.

Building a physical server infrastructure is hard, time consuming, and expensive. Developing commodity software is also a waste of time. Startups do not have time to spend on tasks other than their core business.

Startups usually have few software developers to do the entire job. It is impossible for a small team to be competent in all technologies. In a software startup, its product codebase is the most important place developers should spend they hours, learning about the business domain and creating inexistent and innovative solution to the startup customers. Moreover, every line of code deployed into production increases the maintenance costs. If developers spend time maintaining software non-core for the business, they have less time to work on core functionalities. It is a matter of cost benefit between building and buying. Building software non-core to your business will make you loose focus from your customers.

Therefore:

Use ready-made infrastructures over building your own solution. Prefer SaaS to PaaS, prefer PaaS to IaaS, prefer IaaS to 'building your own infrastructure', except for features core to your business.

The more non-core software you delegate to the cloud, the more time your team will have to work on what really matters: your customer. When you spend more time on your business software, you deliver more value and

learn faster from your clients. By continuously delivering software to your customers you are bound to become competitive and profitable.

If you need a communication tool to keep the startup remote team together, you can use HipChat or Slack. For documents and spreadsheets creation, there is Google Drive, Microsoft Office 360. Virtual machines infrastructure monitoring can be achieved with New Relic. Systems log processing and analysis can be done with Loggly or Splunk [Cukier 2013]. Do remote pair programming with Screen Hero.

Most of these SaaS, PaaS and IaaS solutions were a startup someday. Off course if your startup product is a cloud system, developing a cloud service is core to you, so you won't delegate these core functionalities to third party companies. But even cloud service providers use third party software. IaaS companies use SaaS. SaaS companies use IaaS, and so on. The DevOps for Startups pattern (see Section 2.9) can help you to choose the right balance between different cloud layers [Lwakatare et al. 2015]. A bad consequence of choosing the wrong cloud solution is that you can get stuck to a specific provider, so take care, prefering standard and replaceble solutions instead of proprietary hard to migrate architectures.

2.5 Networking

This pattern is about the importance of social networking when building a startup.

2.6 Use the tools

Shows the importance of using existing automation tools for creating landing pages, mock-ups, A/B tests, etc.

2.7 Find your mentors

Explains why it is important to find mentors to guide you along the startup development.

2.8 Reflect about your business

Pattern describing tools like Model Business Canvas that helps you to think and reflect about your business.

2.9 DevOps for Startups

Point to Cukier's patterns about DevOps [Cukier 2013] and their importance to help on the Startup growth and scaling.

2.10 Right Pitch

Even the best products and ideas need a good selling pitch. This pattern discusses best practices on that.

2.11 Acquire Customers

This pattern discusses many tools that help on Customer Acquisition, like SEO optimization, email marketing, web crawlers and robots, social media, etc.

2.12 Valuate and raise funds

What are the investment options for Startups? What are the different investor types? How much money to raise and how much equity to give to investors? This pattern helps on this topic.

REFERENCES

AULET, B. 2013. *Disciplined entrepreneurship: 24 steps to a successful startup*. John Wiley & Sons.

BECK, K. 2000. *Extreme programming explained: embrace change*. Addison-Wesley Professional.

BLANK, S. AND DORF, B. 2012. *The startup owner's manual*. K&S; Ranch.

BROOKS, F. 1987. *No silver bullet*. April.

BROWN, N., CAI, Y., GUO, Y., KAZMAN, R., KIM, M., KRUCHTEN, P., LIM, E., MACCORMACK, A., NORD, R., OZKAYA, I., ET AL. 2010. Managing technical debt in software-reliant systems. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*. ACM, 47–52.

COMETTO, M. T. AND PIOL, A. 2013. *Tech and the City: The Making of New York's Startup Community*. Mirandola Press.

CUKIER, D. 2013. Devops patterns to scale web applications using cloud services. In *Proceedings of the 2013 companion publication for conference on Systems, programming, & applications: software for humanity*. ACM, 143–152.

ELORANTA, V.-P. 2014. Towards a pattern language for software start-ups. In *Proceedings of the 19th European Conference on Pattern Languages of Programs*. ACM, 24.

FRENKEL, A. AND MAITAL, S. 2014. *Mapping National Innovation Ecosystems: Foundations for Policy Consensus*. Edward Elgar Publishing, London, UK.

GOLDMAN, R. AND GABRIEL, R. P. 2005. *Innovation Happens Elsewhere: Open Source as Business Strategy*. Morgan Kaufmann.

HOKKANEN, L. 2015. Three Patterns for User Involvement in Startups. *EuroPLoP*.

KON, F., CUKIER, D., MELO, C., HAZZAN, O., AND YUKLEA, H. 2014. A Panorama of the Israeli Software Startup Ecosystem. Tech. rep.

KRUCHTEN, P., NORD, R. L., AND OZKAYA, I. 2012. Technical debt: from metaphor to theory and practice. *Ieee software* 6, 18–21.

LWAKATARE, L. E., KUVAJA, P., AND OIVO, M. 2015. Dimensions of DevOps. In *Agile Processes, in Software Engineering, and Extreme Programming*. Springer, 212–217.

PINK, D. H. 2009. *Drive: The Surprising Truth About What Motivates Us*. Riverhead Hardcover.

RIES, E. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Publishing Group.

SCHWABER, K. 2009. *Agile Project Management with Scrum*. Vol. 2009. O'Reilly Media, Inc.

SENOR, D. AND SINGER, S. 2011. *Start-up nation: The story of Israel's economic miracle*. Random House Digital, Inc.