An Experiential Report on Using a Software Factory in a Rural State to Promote Entrepreneurship

Clemente Izurieta
Gianforte School of Computing
Montana State University
Bozeman, Montana, USA
clemente.izurieta@montana.edu

Sharlyn Gunderson-Izurieta
Gianforte School of Computing
Montana State University
Bozeman, Montana, USA
sharlyn.izurieta@montana.edu

***Abstract*— Teaching software engineering and developing a sense of entrepreneurship in rural communities is challenging, particularly when students need to develop hands-on experience in the context of a realistic work environment. A Software Factory was established at Montana State University in 2014 as an innovative approach to teach entrepreneurship and software engineering. The physical space of the Software Factory emulates a real-world environment that facilitates an intimate experience for undergraduate students to interact with professional organizations such as commercial companies, startups, non-profit organizations and schools. Many computer science students located in rural states face challenges when trying to obtain professional experiences. The Software Factory provides a self-sustaining and meaningful way of bridging this gap by pairing teams of undergraduate students with viable established or startup high-tech companies. In this experiential report, we present a compilation of results from 16 collaborations over four years, the challenges faced, the lessons learned to date, and our plans for future improvements.**

Keywords—Undergraduate Education; Software Factory; Entrepreneurship; Experience Report

I. INTRODUCTION

The Software Factory at Montana State University (MSU) [5] began in August 2014 after a visit by the primary author to the University of Helsinki [4], where an established and proven factory had been in operation for several years. After modifying the *experience* of the Software Factory to fit within the academic model of a US university, we identified space, prepared a self-directed student experience, identified potential partners in the high-tech industry, and selected critical practices and software that would aid students in the management of their work and their interactions with external organizations. This effort was undertaken to address the challenges faced by students in rural States, such as Montana, that seek meaningful hands-on software engineering and entrepreneurship experiences.

We began this effort after noticing a significant increase in requests for help with developing technologies from entrepreneurship undergraduate students, and as a response to improve the knowledge and experience necessary to succeed in a new enterprise. According to Forbes [17], 9 out of 10 startups do not succeed for any number of reasons, and many can be mitigated by providing hands-on experiences. This can be challenging in rural States, where opportunities to participate in maker-spaces, or think tanks is limited.

The focus of the Software Factory at Montana State University is to provide a physical ideation space for developing hands-on experiences in the formation and growth of technology based enterprises.

II. SOFTWARE FACTORIES

The Software Factory is an undergraduate pedagogical laboratory under the Software Engineering Laboratory (SEL) in the Gianforte School of Computing at Montana State University. It is an educational facility located in a specially furbished room that is meant to emulate the open spaces that many modern software companies use today. The idea is to stimulate entrepreneurship and research, and create a space where technologies can be ideated and developed by students. It is a platform that provides the necessary resources to deliver real products in the form of software prototypes. The Software Factory promotes student learning, sharing and growing of entrepreneurial ideas and collaborations. The Software Factory brings together entrepreneurs, professional software developers, practitioners and managers with undergraduate students, thus enabling unique cooperative projects that can serve as incubation points for new ideas or in the case of more established organizations, a collaborative extension to high-tech companies where students can help develop non-critical path software prototypes.

The idea of a Software Factory approach for MSU was developed by working in close collaboration with the University of Helsinki; however, methodology changes were required in order to accommodate for the different academic schedules of MSU students as well as the different high tech environments between a rural town and a large urban city like Helsinki. Further, since we were not allowed to offer the Software Factory experience as a 6-credit hour course (the equivalent of the 7-11-week experience in European universities), we developed our experience as an option to a 4-credit capstone project lasting 2 semesters in the final senior year. One of the goals of MSU's Software Factory is to establish a self-sustaining (i.e., self-funding) center that serves as a pedagogical laboratory for students without easy access to internship opportunities that their urban student counterparts may have. Students at MSU's Software Factory [1] engage in activities that promote "*1) Growth:* Develop

National Science Foundation (NSF) Grant# 1658971

software prototypes that support new business ventures, or complement existing software products from public, private and non-profit groups, *2) Learning:* Development of computer science students in the context of a real business environment, and *3) Sharing:* Development of intensive, hands-on collaborations between companies and students (through projects) to explore the deployment of new ideas, research, and knowledge sharing."

The activities promoted by the Software Factory embody the goals originally proposed by the Software Factory community in Europe [14] [15] which is especially active in Finland, Italy, and Spain. At MSU, the Software Factory opens up opportunities in a risk-free environment to address *Growth* by positioning the Software Factory as an ideation space in the rural state of Montana. Students learn how to operate in a business setting with an agile/lean approach to delivering a product –hands-on-entrepreneurship. This is extremely valuable to students and partner organizations alike. In rural states like Montana, companies struggle to compete for top talent from regional universities, and have found that by partnering with the Software Factory, they are able to "interview" students for the duration of the project and also, they are able to court and recruit students much earlier. Further, in many cases, our students have produced software prototypes that the sponsoring company has taken to production. The *Learning* component allows students to experience firsthand, the immersive experience of dealing with customers, changing requirements, developing a business sense, and the dynamics of group collaboration in a physical setting similar to what (startup) software companies offer. Internships in high-tech companies in rural states are competitive, and experiencing entrepreneurship in a startup is almost non-existant, so this allows more students to consider their local communities and realize similar opportunities to learn how to operate in real-world working environments. This is important when rural states are trying to stop the *brain drain* of their young talent. This is something that cannot be taught in classroom settings and the student's testimonials have been positive. Finally, the *Sharing* component facilitates an exchange of best practices and lessons as well as the usage of tools that are used professional software development organizations. For example, GitHub [6] and Kanban [7]. GitHub is a source control and source versioning and archiving mechanism, and Kanban is a scheduling system that allows for the tracking of multiple tasks as they move from one stage of development to another. Software engineering teams have adopted this approach (originally used in the automotive production plants of Toyota) to track deliverables of components. A board is the central component of the system as it allows all participants in a project to visualize progress. Online versions of the software allow for many metrics such as number of tasks, project speed, and tasks per developer, to be tracked by software. The method used by the Software Factory at MSU is a variation called Scrumban [8] [9]; where we borrow some aspects of Scrum techniques and pair them with the visual power of Kanban.

The Scrumban approach has worked extremely well with our students as well as our partners. Students and partners have access to the same board; which allows for easy communication as well as tracking of tasks.

Figure 1 shows a group of students interacting with a customer in the physical space of the Software Factory. Customers are required to interact with the students by either visiting the students in person, or by meeting via any video telecommunications technology.

Fig. 1. Software Factory physical space

III. Experience and Collaboration

Since 2014, we have partnered with 16 different organizations. Table I lists the partners and the nature of the collaboration. Over four years we have served 40 undergraduate students and collaborated with four types of organizations. Regardless of the type of participating organization, students are given full ownership of their projects and are required to operate as if the project represents their business and as if the sponsor represents their customer. A mentor or responsible professor does not get involved unless a significant issue cannot be resolved and is beyond the control of the students.

The four types of partnerships include commercial, startup, and university collaborators and bootstraps. Commercial organizations work in either of two modes. They either incorporate a group of students into an existing project, or they provide requirements for a new project that needs to be designed and developed anew. Most Commercial organizations act as sponsors of projects by providing a dollar gift amount to the Software Factory. Vendor Commercial organizations provide support in the form of tools. For example, Kanbanize [2] [13] helped us by providing access to their Kanban set of management tools and various analytics. Another form of commercial partnership occurs through Startup organizations. These organizations are sponsored by the Blackstone [3] group at MSU. Blackstone is a launchpad facilitator (non-profit University unit) charged with helping establish new student led ventures by helping with marketing and business plans, branding, goals, and any related tasks associated with new businesses. The nature of the relationship with the Software Factory is to help bootstrap new businesses. Finally, we have two types of University relationships: collaborators (i.e., the University of Helsinki) helped as consultants while we established our Software Factory, and bootstrap projects; where the Software Factory partners with

other university departments to help develop new prototypes. In all the sponsor and bootstrap relationships, students are required to interact (on a weekly basis) with an experienced engineer from the sponsoring organization.

A special type of partner is the National Science Foundation (NSF). They support cohorts of 8 students every summer (2017-2019) to develop software prototypes in the context of their research experience. These students' partners are faculty members; however, the same tools and processes as every Software Factory project are followed.

TABLE I. SOFTWARE FACTORY PARTNERS (SHADED ROWS INDICATE PROJECTS WHERE DATA WAS COLLECTED WITH KANBANIZE™)

Project Year	#Students in Project	Partner Name	Organization Type	Nature of relationship
2014	N/A	Kanbanize	Commercial	vendor
2014	N/A	Blackstone	Non-profit	sponsor
2014	4	Zoot Enterprises	Commercial	sponsor
2014	N/A	Univ. Helsinki	University	collaborator
2015	5	Printing For Less	Commercial	sponsor
2015	4	S2 Corporation	Commercial	sponsor
2015	2	Sharelift	Startup	bootstrap
2016	2	Golden Helix	Commercial	sponsor
2016	2	Workiva	Commercial	sponsor
2016	2	Blackmore Sensors	Commercial	sponsor
2016	4	Story Squares	Startup	bootstrap
2016	4	Numo	University	bootstrap
2016	2	Blueprints	University	bootstrap
2017	3	Hewlett-Packard Co	Commercial	sponsor
2017	4	Cowboy Crickets	Startup	bootstrap
2017	2	Precision Agriculture	University	bootstrap
2017	8	NSF	Non-profit	sponsor

IV. PROJECTS

Every project uses Kanban style approaches to manage tasks and progress. In Table I, the shaded rows represent those projects where Kanbanize [2] was used to collect information. Partners *Printing For Less* and *Blackmore Sensors* used their own Kanban software (proprietary from their company) and data was not shared because of privacy issues. Projects from *Hewlett-Packard Co., Cowboy Crickets,* and *Precision Agriculture* are currently in progress, and not enough information is available.

A. Summary Statistics

We provide some descriptive statistics that we felt were important to communicate, as they represent the expected outputs of students that are working under a full time load during a school year. These statistics help mentors, partners and principal investigators develop a sense of expectations from the various projects and partnerships.

Table II shows the average number of days required to complete 85% of the tasks for each project. Normalized for the number of students in a project (assuming 2 students per project), all projects managed between 26 and 35 work items with two notable outliers: Sharelift managed 98 work items and Story Squares managed only 12. Both of these projects were commercial bootstrap projects where the involvement of the corresponding customer counterparts varied significantly. The former was well acquainted with Agile Kanban approaches, whilst the latter learned as they evolved their project. It is important to note that many students, as well as new startup ventures are not familiar with Kanban approaches to management, and enforcing disciplined and systematic recording of tasks in a Kanban board can be difficult. Even in cases where students are familiar with the process, they have not had the opportunity to put it into practice.

TABLE II. SOFTWARE FACTORY PARTNERS CYCLE TIME TO COMPLETE 85% OF REQUESTED WORK ITEMS

Partner Name	#days to complete 85% of tasks	#work items	#Students in Project
Zoot Enterprises	23	52	4
S2 Corporation	42	53	4
Sharelift	45	98	2
Golden Helix	74	24	2
Workiva	56	32	2
Story Squares	67	24	4
NuMo	45	70	4
Blueprints	57	25	2

Figure 2 provides a summary of the Daily Work In Progress (WIP) for each group that reported information. In agile software environments you can setup limits to the amount of work that can occur; thus helping with the identification of inefficiencies in a workflow. We allow students to break down tasks into work items that can be completed in less than two days. If a task takes longer then they need to consider dividing it into more tasks. This approach has worked well with partners and students.

Each project is scoped over two semesters, where the first semester is spent researching and learning the problem; whilst the proposed design is implemented during the second semester. Thus, most of the *churn* is observed after approximately workday 90. Prior to work day 90, most teams are either learning the technologies of the partner organizations or preparing the hardware, software and work environment to develop the prototype, thus low activity can be observed. In the cases of projects sponsored by *Golden Helix*, *Story Squares*, and *Zoot Enterprises*, the teams did not begin recording data in Kanban until the end of the first semester, thus the number of work days reported for each of these projects was reduced, and you can observe high activity levels begin to occur sooner than in other projects.

V. CHALLENGES

A. Rural States

Advancements in technology have allowed unprecedented connectivity between universities. Yet, despite these significant improvements in connectivity, students in rural

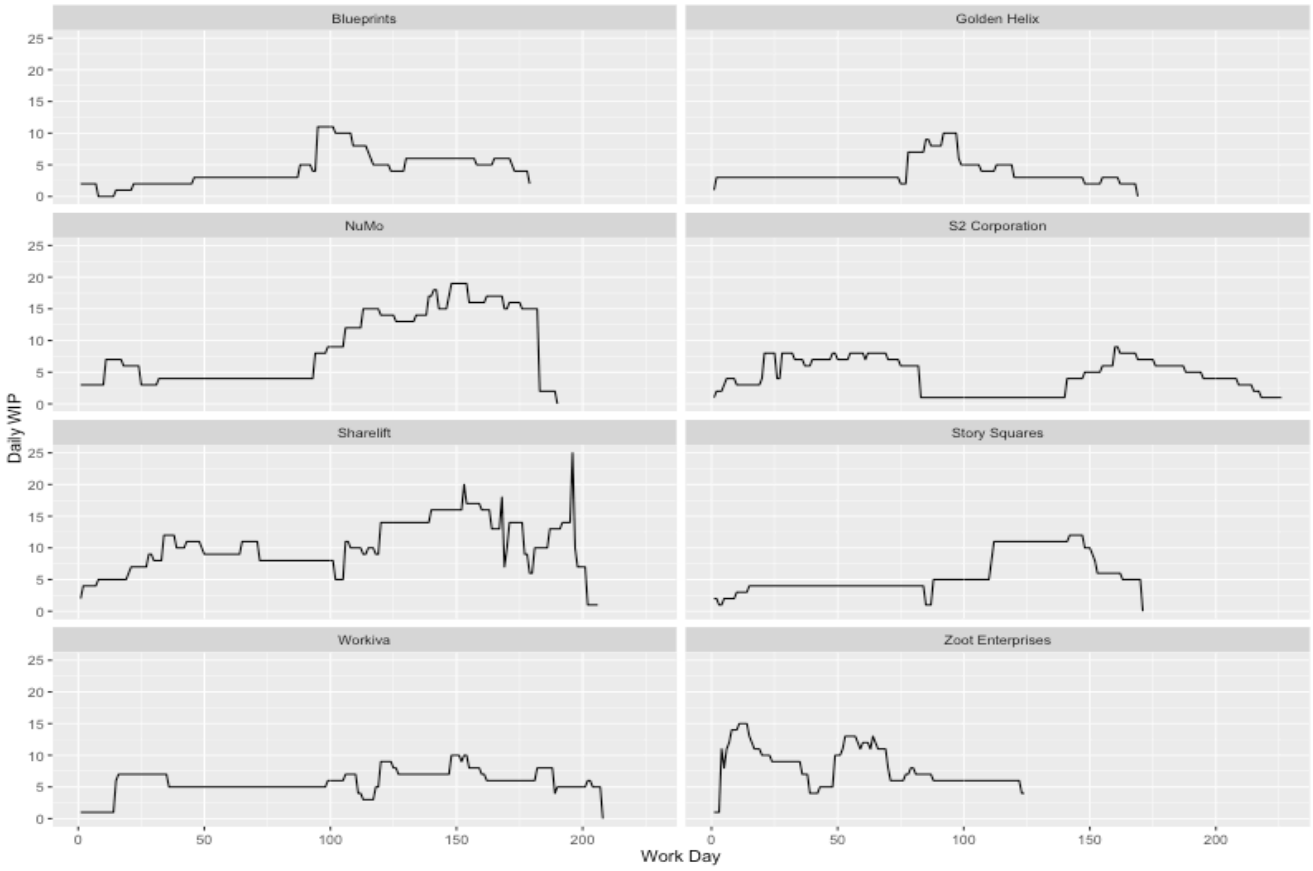

Fig. 2. Daily Work in Progress (WIP) for each student project

states lack the opportunities afforded to student in urban areas when it comes to internships and experiencing entrepreneurship. The latter has become a necessity when students look for full time employment or more importantly when students consider the possibilities of starting their own companies. The latter presents an interesting problem because many students believe that bootstrapping a software company is simple (given the relatively low costs of entry) and may not consider the potential drawbacks. Through the Software Factory, we are helping to mitigate this problem by providing meaningful experiences with local and global companies. Our experience with students that have participated in Software Factory projects since 2014 has been significant. All 40 participating students received offers of employment from the sponsoring companies, and although only 2 students have decided to pursue startups, they all commented how much they learned about entrepreneurship and the skills necessary to give themselves a chance to succeed. Further, two of the startup businesses we partnered with mentioned the importance of the Software Factory. The resulting prototypes provided to the startup partners boosted their portfolios when they sought additional funding from angel funds, venture capitals and commercial funding.

Rural states are handicapped when it comes to providing adequate training for entrepreneurs. The Jake Jabs College of Business at Montana State University provides an *Entrepreneurship and Small Business Management* minor and legislative initiatives in every rural state want to retain these highly capable students. Lessons learned in the Software Factory include the recognition that a successful business can be (physically) located anywhere, and this is especially true in startup businesses related to software.

B. Selection of Students

The selection of qualified students for sponsored projects is a challenge. Students must be self-organizing [16] with an aptitude for working independently and a tolerance for uncertainty and adaptation to (at times) high pressure in the form of continuously changing requirements. All students interested in participating are required to submit a two-page curriculum vita that is reviewed by both mentor professors as well as the collaborating organization. As shown in Table I, projects have ranged anywhere from 2 to 5 students. The latter being the most challenging with expected group dynamics at play. Although many studies [11] [12] have addressed optimal team size in Agile environments, ultimately the sizes of the groups are determined by the sponsoring organizations.

C. Project Support

In order to establish a self-sustaining laboratory, we created a business model where large and established commercial organizations are asked to contribute a monetary donation. Amounts are modest and vary between $3K - $5K per project. Smaller startup organizations are typically sponsored by Blackstone [3] with smaller donations in the range of $1K per project. NSF funding was obtained at a much higher (small grant) level; however, these monies are tied to the research and development of the NSF program, and are not used nor allowed in Software Factory expenditures. Despite these challenges, we have found that the modest contributions from industry partners have provided enough funds to maintain the program while providing significant benefits to all parties involved: students, sponsors and mentors/professors.

D. Intellectual Property

A common question from potential collaborators is the issue of intellectual property (IP), and establishing collaborations with many third parties implies a need to address how to handle certain privacy rights. The Software Factory is used as a pedagogical laboratory and environment to help provide an experience to undergraduate students and has no interests in the intellectual property associated with stakeholders. The focus of the Software Factory is on ideation rather than commercialization.

We have established the following rules, which continue to evolve:

• Tag source code, designs, documentation, data, products and processes as private, confidential, or public depending on the nature of the relationship with a stakeholder

• Understand collection, handling and dissemination of data commensurate with the stakeholder expectations

• Obtain explicit written consent from all participants in the Software Factory (students and stakeholders) to make sure rules and policies are clearly communicated

• Establish legal agreements with authors to transfer the rights to the university when appropriate

• Create draft proposals for every project that include a research agreement, an agreement on transfer of intellectual property, and data file descriptions

After projects reach a certain maturity date, and depending on the stakeholder, data and software elements are either archived or destroyed.

VI. Lessons Learned

We continue to evolve our pedagogical experience and this process has naturally allowed us to look back and review what has and has not worked.

A. Failures

Failures are part of the learning experience for both student participants as well as the mentors and advisors of the Software Factory. We have found that teams of 2-3 students are optimal in commercial sponsor relationships. In larger teams (>=4) we did experience communication hardships between students. In one specific case, the communication hardships extended to the sponsoring organization with an almost complete collapse in the required weekly meetings. Although the crisis was ameliorated through scope pullback, the cause of the failure was the disparate skill levels of the students, which led to engagement discrepancies. This trickled back to the sponsor and manifested itself through missed deadlines and missed functional requirements.

Post mortem interviews with participating students revealed that problems were not caused by skill levels as much as they were by the constantly changing requirements from the sponsoring organization. This generated frustration on part of the students, which trickled to communication problems. It was also an opportunity for students to learn first hand how to approach these situations and resolve the problems by engaging with the customer. Many students revealed an aversion to wanting to engage with the customer because this was a new experience for them and for fear of making the situation worse, when in fact the customer expected that the students would push back when unreasonable requests were made.

B. Entrepreneurial Education

Although many of our students participated in projects with startup companies, many also participated in projects with established organizations or Fortune 500 companies. In the latter, we observed an interesting dynamic. The established companies pushed their processes and tools with our students; however, after some negotiations we requested they use our management approaches using Agile and Kanban style tools to develop software. Testimonials from these projects indicate that students are able to innovate more by not being tied to an established method from the partner company. This was echoed by the engineers working with our students. Although the tools may have been different, the underlying process was the same. For example, the Kanban approach to organizing tasks was still viable regardless of the technology used, but students appeared to react better when the focus of the challenges lied in the development of the core problem, not the peripheral tools used to help the process.

Another interesting observation is that students adapt very quickly when facing communication problems by finding their own solutions. Rather than asking for tools to facilitate engagement, they have a ready made tool bag of technologies from the mobile world. They adopt and delete mobile applications with surprisingly fast turn over ratios. As soon as an application stops providing convenient functionality, it is dropped for a different application.

C. Structured Formats

Whilst the freeform approach to the projects has been a success, and students as well as stakeholders have commented on the importance of the lack of structure when running the projects, some less experienced students can benefit from a more structured and handheld approach. It is true that in a startup organization, its members need to learn how to be a *jack-of-all-trades*, however this approach does not work when the students are unprepared. We have introduced a small period of time where we provide tutorials on technologies and

processes such as Kanban, Scrumban, and paired programming. We have also asked that stakeholders provide small primers on their communication styles and technologies prior to beginning the Software Factory projects.

D. Physical Spaces

Feedback from students reiterates the importance of a physical environment that provides a sense of ownership condusive to innovation. A space that allows a sense of creativeness has been one of the best selling tools to attract potential students to Software Factory projects. The space is owned by the students for the entire academic year in their final year of their undergraduate degrees and they experience the sense of ownership needed in successful entrepreneural environments.

VII. Threats to Validity

There exist two threats to the validity of this work we want to address. We follow the definitions of Wholin et al. [18] to address the content and the external validity of the Software Factory as a viable means to deliver entrepreneurship education and experience.

Content validity refers to how complete the observations cover the content domain. With respect to the Software Factory, we are in the process of learning those variables that adequately provide information and feedback that adequately capture measuring the quality of the entrepreneurship education experience that student participants are obtaining. We believe that we are still in the exploratory phase of developing the Software Factory, and these markers will slowly reveal themselves. Our goal is to maximize the entrepreneurship experience for students, thus keeping track of the correct independent variables to measure this experience is important.

External validity refers to the ability to generalize results. Although we have successfully run 16 projects with 40 students, this is still a relatively small sample that makes it difficult to statistically generalize our results. However, we are building consensus that this approach is viable and successful. Generalizability also includes scalability and we conjecture that this will be difficult to achieve without losing quality. It is important that the entrepreneurial experience is not diminished and scaling these experiences to more students requires significant resources.

VIII. Conclusion and Future Improvements

Montana is the fourth largest state in the US, with a population of just over 1 million. This creates a unique challenge when students seek entrepreneurship education and internship opportunities in-state. The student experiences at MSU's Software Factory facilitate an academic sense of entrepreneurship that can be practiced *in-vivo*. This is not feasible in a traditional classroom, and the testimonials from both students and participating companies continues to make us improve this laboratory experience for students in rural settings. Although projects in the Software Factory are software related, opportunities to develop other entrepreneurial skills have not yet been explored, and we hope that collaborations with the Jake Jabs College of Business as well as the College of Architecture can help us develop experiences that expand the Software Factory in many other dimensions. Additional opportunities to collect high quality data (i.e., benefits, expectations, tool selection, etc.) from participating organizations are planned through the use of surveys and interviews.

Acknowledgments

The Software Factory establishment was made possible by Zoot Enterprises [10]. Support for student projects is made possible by the organizations listed in table I. Finally, support for an REU Site was made possible by NSF grant 1658971.

References

[1] C. Izurieta, M. O'Bleness, M. Trenk, Gunderson-Izurieta S., "The Effectiveness of Software Development Instruction Through the Software Factory Method for High School Students," 123rd Conference in Engineering and Education, ASEE, New Orleans, June 26-29, 2016.

[2] Kannanize [Online]: http://www.kanbanize.com

[3] Blackstone [Online]: http:// http://www.montana.edu/launchpad/

[4] The University of Helsinki Software Factory [Online]: http://www.softwarefactory.cc/

[5] Montana State University Software Factory [Online]: http://www.bobcatsoftwarefactory.com/

[6] Github. [Online]: http://www.github.com

[7] The Kanban Method [Online]: http://www.toyota-global.com/company/vision_philosophy/toyota_production_system/just-in-time.html

[8] Ikonen, M., Pirinen E., Fagerholm, F., Kettunen, P. and Abrahamsson, P., "On the impact of Kanban on Software Project Work: An Empirical Case Study Investigation," in the 16th IEEE International Conference of Complex Computer Systems (ICECCS), 2011, pp. 305-314.

[9] Kniberg, H., Skarin, M., Kanban and Scrum: making the most of both. USA: C4Media Inc. 2010.

[10] Zoot Enterprises. [Online]: https://zootsolutions.com

[11] Putnam D., 2015. "Team Size Can Be the Key to a Successful Software Project." Quantitative Software Management Inc. QSM. [Online] Available: http://www.qsm.com/process_improvement_01.html

[12] Renger M., Kolfschoten G.L., Vreede G.J. "Challenges in collaborative modeling: a literature review and research agenda." International Journal of Simulation and Process Modelling (2008).

[13] Kanbanize. "Kanbanize in the Software Factory at Montana State University," 2014. [Online] Available: https://kanbanize.com/kanbanize-at-the-software-factory-of-montana-state-university/

[14] D. Taibi et al. "Free Innovation Environments: Lessons learned from the Software Factory Initiatives," International Conference on Software Engineering Advances, IARIA, 2015, ISBN: 978-1-61298-438-1

[15] F. Fagerholm et al. "Patterns for Designing and Implementing an Environment for Software-startup Education," 43rd Euromicro Conference On Software Engineering and Advanced Applications, SEAA, Aug. 30 – Sept. 1, 2017. Vienna, Austria. doi:10.1109/SEAA.2017.67

[16] X. Wang, I. Lunesu, J. Rikkila, M. Matta and P. Abrahamsson, "Selforganized Learning in Software Factory: Experiences and Lessons Learned". In Agile Processes in Software Engineering and Extreme Programming. pp. 126-142, 2014.

[17] N. Patel, "90% of Startups Fail: Here's what you need to know about the 10%," Forbes. Jan 16, 2015. [Online]: https://www.forbes.com/sites/neilpatel/#1925a09f181e

[18] C. Wohlin, P. Runeson, M. Hst, M. C. Ohlsson, B. Regnell, and A. Wessln, Experimentation in Software Engineering. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012. [Online]. Available: http://link.springer.com/10.1007/978-3-642-29044-2