Software Project Management Combining Agile, Lean Startup and Design Thinking

Bianca H. Ximenes(✉), Isadora N. Alves, and Cristiano C. Araújo

Department of Informatics (CIn), Universidade Federal de Pernambuco (UFPE), Recife, PE, Brazil
{bxhmm, ina, cca2}@cin.ufpe.br

Abstract. This paper describes a project management model named Converge, that combines Agile, Lean Startup and Design Thinking with the aim of producing user-centered software and sustainable innovation through empathy with users. The model is based on previous works combining the aforementioned methodologies and adjusted considering needs that arose from teams inside the lab, observed empirically. In order to test the method's validity in a real project, an undergraduate team part of an experimentation lab followed the proposed model to guide the development of a homonymous data storage app.

The app was built in 8 weeks and, at the time of release, 80 % of testers considered it a better solution compared to ones they already used. Overall test results suggest that it is productive to combine the methodologies. The model met its aim since it guided the development of a novel software solution highly regarded by users.

Keywords: Project management · Agile methodologies · Lean Startup · Design Thinking · Model · Software development · Innovation · Case study

1 Introduction

The software development industry has gone through great changes in the last decade. The shift from computer to mobile devices, alongside the popularization of ubiquitous computing, internet of things and wearables have expanded markets and changed the industry. All these factors combined offer many opportunities in innovative new market niches.

But how can software development teams lead an innovative process, create a five-star product and still deliver in time? Methodologies such as Agile (using techniques such as Extreme Programming, Scrum, Kanban) have become more common and attempted to change this scenario, preparing teams to be more adaptive and coming to closer contact with clients and customers. However, some issues are still left unresolved or not satisfactorily handled. Griffith [1] appoints the most prominent reason for software project failure as being building something for which there is no market – essentially, products nobody wants. This is most likely due to the fact teams become too focused on technical aspects and entrapped in their own product vision, as seen in [2], which also highlights the products built do not solve real problems. Furthermore, [3] also indicates no go-to market strategy and lack of competitive research

A. Marcus (Ed.): DUXU 2015, Part I, LNCS 9186, pp. 356–367, 2015.
DOI: 10.1007/978-3-319-20886-2_34

as common problems. Finally, [4] explains the major pitfall is the teams' inability to put themselves in the users' shoes, truly understanding the way they think and what they need, a concern echoed by all other authors.

Two recent movements that have become popular in the technology and software industries are Lean Startup and Design Thinking. Lean Startup brings a market focus to project development and was conceived inside a software development company. The Lean Startup movement begun with Eric Ries, who defined a startup as being "a human institution designed to create a new product or service under conditions of extreme uncertainty" [5]. Its philosophy is to transform teams into efficient learning units that do not build products nobody wants to use. Design Thinking, made popular by Tim Brown [6] conveys that the methodology employed by designers when approaching problems can be applied in all areas of knowledge in order to achieve innovation. True innovation lies in working out a creative solution that is desirable for the consumer, economically viable and technically feasible. Thus, Design Thinking is a tool that helps develop new products, services, processes and strategies. These two movements address common problems identified by specialized literature regarding IT project failure, in both the business and the user aspects. Besides, they encourage an environment where it is possible to transform uncertainty into new products, generating innovation.

For all reasons presented above, the idea of combining them in the software development scenario and making a unique model with the state-of-the-art methods for software and startup developments, and user experience design with the aim of fostering innovation appeared to be worthy of further investigation.

2 The Converge Model

The model had its inspiration in previous works combining Agile, Lean Startup and Design Thinking, as well as empirical observations inside the experimentation lab. Converge model is applicable to development teams in need of creative solutions. In this scenario, the solution is not yet known; in reality, several times the problem itself is complex and not yet defined. Such characteristics are typical of Design challenges (which address wicked problems) [6, 7] and common in the startup scenario as well, which deals with extreme uncertainty according to the definition present in the Lean Startup philosophy [5]. Its graphic representation is depicted in Fig. 1.

Agile provides the necessary structure to coordinate software development and deployment. It reminds every one of the time constraints and breaks tasks into small portions members can approach and tackle effectively; Scrum meetings keeps the team bounded and informed of each other's tasks. Agile's iterative approach to software development also provides the necessary basis for developers to change and rebuild fast, in case it proves necessary – a culture in which it is admissible to pivot [8]. All team's tasks are described and fit into weekly or 2-week sprints maximum, be the member a developer, a designer or from any other area of expertise that comes to be involved in the project. Since the main output of the lab is software, the organizational structure was conceived to support its development at best. Requirements exist, but they do not come from a single client; the team themselves establish the requirements based on potential customers pull and extreme users, crossing their references with their own product vision (a Lean Startup characteristic).

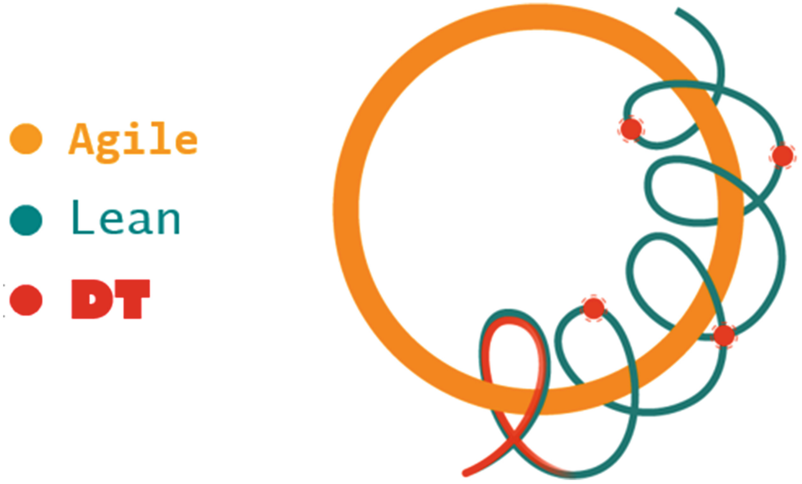

Fig. 1. The converge model graphic representation

A feature-by-feature implementation is followed strictly; some other Extreme Programming values such as pair programming and collective code ownership [9] are also part of the model fundamentals. A complete project would last a complete cycle – from concept to publishing - and lasts the sum of all sprints, hence the Agile being represented by the bigger outside circle, with the estimated duration of three months [10]. As it was previously mentioned, lab members were too young and inexperienced to constitute a self-organizing team. One of the authors worked as a Scrum Master, closely to a Project Leader (PL), responsible for the project vision. This Leader came from inside the lab and was supervised until he gained independence and understood the process well enough to operate as a Scrum Master himself, if needed. Each project has their own Scrum meeting the first day of the week, done separately. Tasks were registered and managed using Asana. Because of students working only part time in the lab and their schedules not always matching, there were no daily Scrum meetings.

Since the model is to be applied in innovative projects which are not always funded, the model is also concerned with approaching the product development issue, following Lean Startup precepts. Lean is based on the constant validation or rejection of hypothesis one has about an unclear problem, while trying to achieve an unclear solution [5]. Every week, the teams must go through a build-measure-learn loop, learning more about the product and maturing its concept, being able to adjust to demand and pivot strategies. It is essential to establish key metrics to assess learning – otherwise, loops do not exist. Lean focuses on building an MVP and arriving fast to market, which works especially well for mobile application development, since quantitative and qualitative consumer feedback is available at ease on application markets on all platforms. These help continue the learning cycles and product improvement.

In the first week, team defines their strategy and fill in the Lean Canvas [11] to have a holistic perspective of the project. Like Agile, Lean Startup (and development) also focuses on delivering working code often [12]. Unlike Agile, however, a Lean Startup team has a product vision [13], hence the Project Leader role advocated in the model – it is not only meant to facilitate training. This Project Leader has a lot to do with the Scrum's Product Owner [14]; however, Converge's PL is not the client. Instead, they must motivate the team to find users and clients (early adopters) for their products. The early adopter is not necessarily a single person or company, but rather a market niche. That is why software requirements in the Converge model are a mix of what the team believes and what they can find while talking to or observing potential users. Lean Startup is also important because, due to the exploratory aspect and the focus on learning, it teaches students to balance their views with what is feasible and desirable for the market (a concern present in DT as well.) Moreover, for Lean Startup, the user (who uses the software) and client (who pays for software, service of information) are not necessarily the same thing, forcing students to start to consider business models.

A direct consequence of this was that students were free to work on projects before the introduction of the Converge model and remained so; they had, nevertheless, to prove qualitatively there was a demand for that type of app and that they were addressing a real need. This was done following [11]: they had to validate problem and idea separately with at least 5 people outside the lab, independently. Finally, to prepare students to presentations with investors and to pitch their projects, a Results meeting was held weekly, on the last day of the week. In it, a team member from each project (usually, three different projects were running simultaneously in the lab) would present the problem, the solution, the activities and the working code produced in the week. At these meetings, all lab members helped their colleagues from other projects improve their presentations and establish the hypotheses each team would have to validate or reject the following week.

The loops presented in the model follow through the whole extension of the structure circle. They never stop, for hypothesis validation and learning are always present, guiding the decisions. Their repetition is meant to indicate their continuity.

For the last part, some challenges present in software development projects are neither technical nor market-related; they concern the ability to understand the user, their limitations and needs, and to come up with different creative solutions beyond technical issues. Those challenges are represented by the "knots" and may appear at any point of a project. When they appear, the main reason for the hold-up is diagnosed and Design Thinking techniques applied to overcome those problems and come up with solutions. Those techniques are applied in DT rounds, which fit in weekly sprints, and last 1–2 h. Therefore, in the model, to each knot identified, there is a correspondent DT round. Just like the loops, their existence throughout the whole process is indicated by their repetitive presence in the model.

Following the model, which is inspired by d.school [15] and depicted in Fig. 2, knots are catalogued in five categories: Empathizing, Problem Definition, Ideation, Prototyping and Testing. They may occur in any order and must be recognized and addressed.

The relevance of Design Thinking mindset and techniques are not only about overcoming project challenges, but also about adding value because it allows for real insight on users' needs. That is why Design Thinking is also present during the first week of the project, as seen in the model – techniques are applied to the problem identification and idea validation phases preached by Lean, but also to explore and understand the nature of the project and users' reality – empathize – which sometimes may be completely different from developers' experience. A key feature in the Converge model is that the team responsible for the idea and design is also fully responsible for the development of the final product. This is relevant as the budget and time constraints encompass the project as a whole, from conception to delivery to the app store. Besides, all team members take part in the creative process from the very beginning until the end – developers do not receive ready-made requirements from others, taking part in the making of the product concept, being able to innovate. This follows suggestions from [16]. The following subsections explain what is done in each phase:

Empathizing: This phase is important to reduce the time spent in discussion and hypothesizing inside teams. It was observed that the students had a tendency to make decisions without inputs from customers and users, based only on their own perceptions of the product. At times, when it comes to a dead-end, members will take a vote to decide what to do next. This practice helps them see the customer as the person who has the final word. It is very focused on research and interviewing, and meant to allow team members to learn about their competitors, monetization and market, besides user behavior. Most used technique in this phase inside the lab: interviews with extreme users [17].

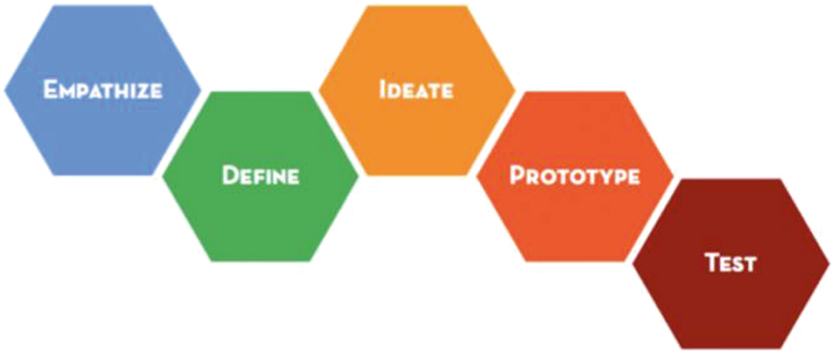

Fig. 2. Stanford's d.school Design Thinking flow

- **Problem Definition:** This phase is more commonly necessary when the team cannot find common ground and lacks alignment. It is noticed when there are no clear goals; team members define the project in different ways; functionalities implemented are of little importance to the product or do not complement each other. Defining the problem makes sure everyone knows what they want to achieve and work towards it, providing focus. Most used technique in this phase inside the lab: five whys method [18].

- **Ideating:** It is a very comprehensive phase, ranging from techniques to choose names of applications to the conception of a whole new project idea. Teams learn to think in new ways, be less judgmental and generate a great range of ideas to choose from. Every time someone in a company suggests a brainstorm, this actually means they are in need of an ideating phase. Brainstorm is, although widely accepted and renowned, simply one technique that corresponds to this phase. Most used technique in this phase inside the lab: triggered brainwalking [19].
- **Prototyping:** In Agile, prototypes are almost always a simpler version of the final code. This is important, but here the main focus are wireframes, mock-ups and paper prototypes. The main function of such prototypes is to allow quick testing of a tangible idea, even if it is low-fidelity. Prototypes provide quick feedback from potential customers and are fast to make as well as inexpensive. In the Converge model, the initial validations are done with paper prototypes and Rabiscapp[1] application, one of the lab's products, to assemble the screen flow. Prototyping knots also cover creating more than one prototype for split tests, allowing to better test hypotheses. Prototyping always happens in the first-week loop and afterwards as many times as prototyping knots are identified. Most used technique in this phase inside the lab: representation sketches [20].
- **Testing:** Every test round is an empathy round as well, for teams learn a lot by observing their users interact with the product. The difference is that testing has the development team necessarily offer a version of the product (be it on paper, code or any other applicable medium) for the user to interact. Testing reduces clashes inside the team. It makes it clear what works best for the user. Tests for a creative software project are much like requirement engineering for agile – the client chooses what they prefer. In the Lean Startup mindset, testing is what allows teams to assess whether they are working on a potentially successful product or failing. It validates the project existence. Every test round happens because there is a hypothesis to be proven or rejected. Therefore, whenever there is a hypothesis, there is a test knot. Teams must be trained to understand when they are making hypothesis and assuming things that should otherwise be tested. Most used technique in this phase inside the lab: interaction with application of Likert-scale questionnaire.

3 Validating the Model Empirically

This section narrates the development of a data storage app as a case study, exemplifying how the Converge model works in practice and describing project repercussions.

The First Immersion

First Week: In the beginning, all students in the lab became familiar with the current lab design challenge "How may we refine user's data collection and storage experience?". In order to empathize and understand users' needs, the first step was a 30-minute brainstorm. At first, students had to think of people to whom they had access

[1] To read more about and get to know Rabiscapp, access http://rabiscapp.net.

that they considered extreme users concerning data storage and collection. After coming up with some names, students had to explain to the group why they considered that person an extreme user, and the group decided whether they agreed or not. When the 30 min were up, there was a list of 6 people that were to be interviewed in the next 2 days. They were a young entrepreneur, an engineer that supervised construction works, a national energy agency manager, a public school director, an oral surgeon and the president of a junior enterprise. They were chosen because of the amount of data they had to gather and organize, the diverse types of medias and files they handled, the quantity of people of things the data they collected helped monitor and the high frequency in which that happened (many times every day). This characterizes an Empathy round. Students had two days to interview the people they identified as extreme users.

All interviews were recorded with users' consent. They were later transcribed and brought back to further discussion in the lab, on the 3rd week day. The initial questions were the same for all people, but everyone was free to elaborate if they thought they could get more relevant information. During 2 h, students became aware of the users' work routines, the type of data they had to store and consult, the instruments they used to help them, what they perceived as challenges in their daily lives and how easily they considered to storage, organize and retrieve data. This characterizes another Empathy round.

At that point, the team got wary: they realized interviewees had different needs and tasks to address. It seemed impossible to build an app with high UX and usability standards that tended to all users' expectations, particularly considering the time constraint suggested in the model (3 months). However, they understood that they were not yet aware of the problems underneath, and that those problems could maybe be the same.

It was then time for the Define round, which occurred on the same day. Using the 5-whys technique, students focused on the challenges and difficulties the users described trying to get to the root causes that made users experience the issues they mentioned. By the end of the round, they were able to synthesize common root causes for all challenges users mentioned, identifying 3 problems the app could address:

1. Data is geographically trapped.
2. Data is spread out.
3. Other people have my data.

On the 4th day, it was time for the Ideation round. In this phase, only half the original team continued to work on this project, and others resumed working on other ongoing products. Team used triggered brainwalking to write about how they thought an app could treat the underlying user problems they identified. At first, they thought about functionalities the app could have – natural for a technical team. Ideas were mostly to provide organization by tags, date, native search for content, offline storage of chosen files, automatically group similar files. After the first 20 min, a new constraint was introduced in the brainwalk to allow students to focus on the macro aspects of the project: they could not write down functionalities, but rather describe the ideal digital environment to organize users' data. In the end, students perceived people organized their lives in what the team named "projects", which could be personal or work-related, private or shared. All data related to a specific life "project" should therefore be stored together. The app would offer the same functionalities to everyone, but be able to adapt to each user's own reality, according to their personal activities, with varying degrees of granularity. For example, each one of a doctor's patients could be a separate project. A yearly company budget could also be a project, albeit of a different nature. A holiday trip planning, a wedding, could all be projects as well, and shared with as many people as deemed necessary by the user. The project would be a grand central for all data concerning that topic.

Having understood what the app should offer, it was time to translate the concept into an app interface, in a Prototyping round. The team used representation sketches mixed with the 6-3-5 design technique. This part was more focused on experience and functionalities. Common aspects proposed on screens were passed on to a prototype synthesizing best features (not by functionality, but by ease to use). Aspects deemed good by the team were also passed on. The main app flow consisted of 4 screens. None of what was built was final, and students understood that – it was a rough draft they would test with potential users.

On the 5th day, the team's designer started transforming the low-fidelity prototypes on high-fidelity screens, to be imported into Rabiscapp and tested the following week. The final project team was defined, consisting of two developers and one designer. The other students resumed working on other lab projects. Those three members started filling in the Lean Canvas – the areas of Key Metrics, Cost Structure and Revenue Streams were left blank, for the moment. Team also defined what would constitute the MVP and what could be added later; finally, they discussed the hypothesis they had to validate the following week, in the first learning loop.

The Learning Loops

Second Week: Ideally, Lean preaches the validation of the problem decoupled of the solution, in two separate moments. However, as the high-fidelity screens were ready, the team took them on Rabiscapp for a Test round with the extreme users previously interviewed. All 6 extreme users liked the project organization and functionalities, and considered the app flow intuitive. They pointed out some improvements such as enhancing header and text contrast, modifying visual assets as well as changing the color pallete. The best qualitative measurement, nevertheless, was the fact all users demanded new and more screens to test the complete app flow, thus playing the part of early adopters. The app solution and the hypothesis that users organize their lives in projects was validated. At that point, developers started to set up the database tables and enabling projects to be saved in the database. Besides, part of developers' work contemplated studying to overcome their technical limitations, such as implementing animations in Cascades, BlackBerry 10 native language.

Third Week: At this point, the team reassessed the Lean Canvas. They improved the Unique Value Proposition (adding "a structured Dropbox"). It was also necessary to propose a business model to generate revenue, but team did not have ideas because they did not know much about other data storage solutions. This characterized an Empathy knot – it was necessary to learn more about how to monetize in that market niche. Team listed all the competitors they identified besides the ones the extreme users cited as options they used to store data. Students conducted an extensive research and shared their findings amongst themselves. An Ideation knot followed suit, in which the team considered how to offer advantages when compared to their competitors, in a way that also allowed them to reach a break-even point. In the end, team decided Converge would sell monthly subscriptions of USD 5.99, which allowed for sending 10 GB of data each month, and unlimited storage. To do so, team also had to consider their cost structure. That way, two more fields of the Lean Canvas were also filled in. Besides, the business model still needed validation. In the development sprint, developers implemented the functionality of saving files and displaying the ones saved. They also created the different categories for saving files (images, documents, and notes) and a native text composer for the notes section. The designer altered the color pallete and made as visual assets, typography and icons uniform. The team put off validation to the following week. For the first time, Converge app took part in the weekly results meeting and the Project Leader pitched the project to the lab colleagues, who posed questions and gave suggestions for the team to explain and reply, dealing, on a lower scale, with the pressure of presenting a product.

Fourth Week: A Test round validation with the extreme users had them pull two new features – a file preview screen, expanding information available to help them decide whether they wanted to download the file; quick access buttons. Project's folders had icons displayed next to its name on the main screen to showcase which type of files were stored there inside it. The users wanted it to become a direct access to a specific type of file (i.e. images). They also highlighted the real-life elements look (e.g. cork-textured mural, pins, Polaroid frames for images, paper folders), remarking them positively. They were to be present on all screens. The well-known elements made them feel a sense of familiarity. Half the users declared they would pay the price for Converge. The other half said they did not subscribe to digital services, which is a rather common challenge for mobile services is Brazil. The team considered they needed further validation before rejecting or accepting their business model. The final comment in the validation was that the first screen picture and the app logo were not appealing. For that reason, a Prototyping round was set up with the members and other lab designers, to sketch different options until a final layout was reached. The development progressed, creating a synchronization protocol and implementing sync server $\rightarrow$ app and the other way around. Downloading files from server was enabled. The validated interface started to be implemented on code. The team pitched the project and shared their week findings in the results meeting at the end of the week.

Fifth Week: New screens were put forth on a Test round in Rabiscapp – some just created, like the file preview, some refactored, like the image upload screen. The extreme users/early adopters were extremely pleased; however, they felt the lack of file history could make it difficult to keep up with many updated, especially when projects were shared among many people. Beyond that, they were impressed with the app concept and flow. The history functionality was not part of the MVP. The app could be released without it, and have it added a posteriori. Notwithstanding, the team felt the history provided a unique opportunity to make Converge stand out from its competitors. The team's vision prevailed, and a Prototyping round took place in order to decide what the history would include. For that round, two people with the Converge user profile, from Business, were invited to join in. The final prototype result included complex features, such as being able to restore the last 5 deleted files, also present in history. Anyone could restore a file on the list, but the only ones able to delete would be the person who uploaded or the project manager (the person who created the project by default). Another added feature to make uploading files easier was an automatic filter. The system was able to identify the extension and send the file to the correct sub-folder in a given project. The upload button became a single one. This was aligned with the UVP of making it as easy as possible for users to share and store files. The server was set up on Heroku and the file storage on Amazon. Multiple accounts were enabled, and support to shared projects added.

Sixth Week: The first screen and history screen high fidelity prototype got ready, completing the entire app flow. Before implementation of the file history, file ownership and deletion/restauration logic, the complete flow was taken for a final Test round. All six extreme users rated the final result (showed in a high fidelity prototype, not in the real app) 5/5 points. The history screen and functionalities were greatly responsible for the high ratings. The development sprint contemplated providing support for tagging and displaying tags on files in-app, as well as editing or removing them; tags were enabled within the server, to be used when searching for a file; the screens for notes and documents preview were implemented following the validated design.

Seventh Week: At this moment, enough tests and iterations were conducted, allowing for launching. Key Metrics, however, were not yet defined. That required an Empathize round to decide how it was possible to measure whether Converge was fulfilling its UVP and delivering value to users, as well as retaining them. Number of downloads and subscriptions, number of projects and average number of users per project was something obvious to monitor. Besides that, team concluded that to assess whether Converge was accomplishing its UVP, users should use it every day. Daily users were considered active users. To validate their business model and prove it was sustainable, it was important to monitor the number of files uploaded per user (average), uploading frequency (average), and file size (average). Development sprint covered syncing tags. The history screen, with the recent activity list, was implemented alongside the validated interface. Push notifications and images' thumbnails on the main project page were implemented.

Eighth Week: The development sprint included implementing and supporting registration and login, as well as e-mail invitations for collaborators who were not already registered. Other details, such as "share from Converge" (export) and "save to Converge" (import) were added. The final task was implementing metric monitoring inside the app through Flurry and an IAP system. The final Test round before release was carried out on week 8. After 8 weeks of development, final tests with 17 potential users yielded positive results; they were all-new testers, no longer the extreme users who guided the team throughout the process. Even though none of the testing volunteers opted exclusively for mobile storage, in spite of Converge being a mobile-exclusive application, after testing 80 % of volunteers found Converge application provided a superior data storage experience than the one they currently employed and 40 % affirmed they would be willing to pay a monthly subscription to use it. The ones who did not agree with the price suggested that further development combining mobile and computer-based clients would make them more willing to pay for the subscription. This was handled as a possible project development post-MVP. The application was set up for sale in BlackBerry World in the following week, closing the Converge project development cycle and getting to market.

4 Conclusion

The complete cycle was pursued and closed in two months. In spite of the app produced being developed exclusively by undergraduates with little experience, the qualitative validation showed high acceptance of testers and gathered positive remarks. This corroborates the notion that the Converge model operates well in guiding software projects conception and development, until the product arrives to market.

On the other hand, there was not enough time to observe a market experience of the product and try to escalate it. It remains unassessed whether Converge can be adapted to projects already in market and help in quantitative product development.

The case study results suggest that it is possible to combine the methodologies and approach Design Thinking in rounds. The model seems to enable innovation and good product rapport with users. It was possible to make Computer Science undergraduates see beyond requirements and tasks, identifying mental models and real needs. Even though inexperienced developers and designers conducted the project, they were able to understand users and change their behavior by proposing a new way to approach data storage and providing a memorable experience.

The Converge model can be applied and tested in similar settings, those of experimentation labs and incubators. It benefits teams in need of creative solutions that can build and release the product into market fast and cheaply, as it is the case for the mobile content industry.

References

1. Griffith, E.: Why startups fail, according to their founders (2014). http://fortune.com/2014/09/25/why-startups-fail-according-to-their-founders/
2. Tobak, S.: 9 Reasons why most startups fail (2014). http://www.entrepreneur.com/article/231129
3. Deeb, G.: The unlucky 13 reasons startups fail (2013). http://www.forbes.com/sites/georgedeeb/2013/09/18/the-unlucky-13-reasons-startups-fail/
4. Aulet, B.: Disciplined Entrepreneurship: 24 Steps to a Successful Startup. Wiley, New York (2013)
5. Ries, E.: The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses. Crown Business, New York (2011)
6. Brown, T.: Change by Design: How Design Thinking Transforms Organizations and Inspires Innovation. Harper Business, New York (2009)
7. Brown, T.: Design thinking. Harv. Bus. Rev. **86**(84–92), 141 (2008)
8. Nelson, E.: Extreme programming vs. interaction design (2002)
9. Bird, J.: Code ownership – who should own the code? (2011). http://swreflections.blogspot.com.br/2013/04/code-ownership-who-should-own-code.html

10. Poppendieck, M., Poppendieck, T.: Lean Software Development: An Agile Toolkit. Addison-Wesley Professional, Boston (2003)
11. Maurya, A.: Running Lean: Iterate from Plan A to a Plan That Works. O'Reilly Media, Sebastopol (2012)
12. Widman, J., Hua, S., Ross, S.: Applying lean principles in software development process–a case study. Issues Inf. Syst. **11**(1), 635–639 (2010)
13. Nobel, C.: Teaching a lean startup strategy (2011). http://hbswk.hbs.edu/
14. Moe, N.B., Dingsøyr, T., Dybå, T.: A teamwork model for understanding an agile team: a case study of a scrum project. Inf. Softw. Technol. **52**, 480–491 (2010)
15. Virtual crash course in design thinking http://dschool.stanford.edu/dgift/
16. Lindberg, T., Meinel, C., Wagner, R.: Design thinking - a fruitful concept for IT development? In: Meinel, C., Leifer, L., Plattner, H. (eds.) Design Thinking: Understand - Improve - Apply, pp. 3–18. Springer, Heidelberg (2011)
17. Lafreniere, D.: Extreme user research (2008). http://boxesandarrows.com/extreme-user-research/
18. Serrat, O.: The five whys technique (2009)
19. Mattimore, B.W.: Idea Stormers: How to Lead and Inspire Creative Breakthroughs. Jossey-Bass, San Francisco (2012)
20. Beaudouin-Lafon, M., Mackay, W.E.: Prototype development and tools. In: Sears, A., Jacko, J. (eds.) Human Computer Interaction—Development Process, pp. 122–142. Lawrence Erlbaum Associates, Hillsdale (2003)