A combined agile project management approach for mobile application development

Aljaž Daković
Address: Krpanova 9, 2000 Maribor, Slovenia
E-mail: aljaz.dakovic@hotmail.com

13.12.14, Maribor
TABLE OF CONTENTS

1 Introduction ........................................................................................................................ 4
2 Short overview of agile project methodologies Scrum and Extreme Programming .......... 4
  2.1 Scrum ........................................................................................................................... 4
  2.2 Extreme Programming (XP) ........................................................................................ 6
3 Integration of agile project methodologies Scrum and Extreme Programming for the purpose of managing mobile app development projects ................................................. 8
4 Managing mobile app development projects whit combining Scrum and XP ................... 9
  4.1 Preparing Product Backlog ........................................................................................ 10
  4.2 Sprint planning .......................................................................................................... 10
  4.3 Preparing Sprint Backlog ........................................................................................... 11
  4.4 Daily Scrum meeting ................................................................................................. 12
  4.5 Managing the development process .......................................................................... 12
  4.6 Sprint review and demo ............................................................................................. 13
  4.7 Sprint retrospective .................................................................................................... 13
5 Conclusion ........................................................................................................................ 13
6 References ........................................................................................................................ 15
7 Acknowledgements .......................................................................................................... 15

TABLES
Table 1: Comparison of Scrum and Extreme Programming (XP) methodology .................... 7
Table 2: Organizational structure and methodological basis for mobile app development projects ................................................................................................................................. 8
Table 3: Product Backlog example ....................................................................................... 10
Abstract:

The development of mobile applications is a new and extremely fast evolving and growing industry. The dynamics of growth and development is provided by mobile applications development projects that extend the global range of projects and, with their specific requirements, confronts the profession of project management with new challenges. Because of the fast growth of mobile application industry it is important that these challenges are addressed with new project solutions and methodologies.

Mobile application development projects peculiarities are reflected in the relative short-termism and low complexity with regard to the number of individuals involved in the project system of these projects from the beginning of their life cycle until the completion of the implementation phase. For those reasons and with certain contextual specifics of these projects, the applicability and effectiveness of traditional concepts, in particular, project management methodologies, are quite limited. In recent years, suitability of agile project methodologies is often highlighted for projects with similar characteristics as of mobile applications development projects. Some of the more popular agile methodologies were put to the test in real life mobile application development projects and the resolute of their implementation is discussed within this article.

The article presents an original adaptation and integration of the two agile methodologies - Scrum and Extreme Programming, creating a suitable and validated basis for managing those projects.

Key words: project management methodologies, agile project management, agile methodologies, mobile applications, Scrum, Extreme Programming
1 Introduction

Recently we are witnessing a bright of a new industry called mobile applications. A year after Apple announced its smart phone; Steve Jobs took the stage and announced third party apps for their bellowed iPhone. Third party developers where not only allowed to create apps for iOS but for the first time where also allowed to distribute them directly apps through the OS\(^1\). Mobile applications (MA) had existed and had been sold before the introduction of Apple’s App Store but MA where very complex to develop and the lack of a wide distributing channel meant MA development was economically unviable. Today the biggest changes are still low hardware capabilities of devices. To counter makers of mobile OS’s are releasing advanced SDK\(^2\)’s. With this tools developers can quickly and cost-effective integrate new functionality in to their MA. Since then we are witnessing a staggering growth of MA industry. According to ABI Research the mobile app industry was worth 27 billion in 2013.

Mobile app development projects are popping up everywhere. Because of the new challenges mobile app development projects bring new methodologies and project management concepts are being developed. Standard project management practices that had worked great on desktop app development projects fall short because of specifics that are associated with MA development projects. A faster more agile way of doing things is needed.

We suggest integrating two currently popular agile methodologies Scrum and Extreme Programming for efficient organizing and managing of mobile app development projects. By integrating Scrum and XP we can overcome some of the shortcomings of classics waterfall methodologies.

2 Short overview of agile project methodologies Scrum and Extreme Programming

2.1 Scrum

Scrum is an iterative, incremental framework for projects and product or application development. It was first introduced in 1990 and was intended for complex product development projects.

Scrum structures development in cycles or iterations called Sprints. Each Sprint includes the process of planning, development and controlling. Iterations also make it possible that feedback is gathered and dispersed at the end of itch cycle, which can significantly decrees the amount of risk concerning a development project. Sprints take place one after the other. Sprints are of fixed duration – they end on a specific date whether the work has been completed or not, and are never extended.

At the beginning of each Sprint, a cross-functional team selects customer requirements from a prioritized list (Product Backlog). They commit to complete the items by the end of the Sprint. During the Sprint, the chosen items do not change. Every day the team gathers briefly to replan its work to optimize the likelihood of meeting commitments. At the end of the Sprint, the team reviews the Sprint with stakeholders, and demonstrates what they have built.

---

\(^1\) OS – Operating system

\(^2\) Software development kit - typical set of software development tools that allow the creation of applications for a certain software package.
People obtain feedback that can be incorporated in the next Sprint (Schwaber K., Sutherland J 2011, 6).

A project team as defined by Scrum usually consist of a product owner, the development team and a Scrum master. It is characteristic for a Scrum project team to be interdisciplinary and self-organizing. Self-organizing means that the team itself is reasonable for finding the best way to organize work during the project. The team must be interdisciplinary to the point that team members have all the necessary skills and knowledge to successfully complete all tasks that are a part of the development project.

In Scrum, there are three primary roles of team members:

- **Product Owner** is responsible for maximizing return on investment by identifying product features, translating these into a prioritized list, deciding which should be at the top of the list for the next sprint and continually reprioritizing and refining the list. In Scrum it is crucial that there is only one product owner and that the project team have constant access to the product owner if there are any misunderstanding that need to be addressed. It is also very important that the individual in the role of product owner has all the necessary privileges to sign off or approve changes to the product.

- **Development Team** builds the product that the customer is going to use. The team in Scrum is cross-functional and includes all the expertise necessary to deliver the potentially shippable products after each Sprint. It is also self-organizing with a very high degree of autonomy and accountability.

- **Scrum Master** helps the development team learn and apply Scrum to achieve business goals. The Scrum Master does whatever is in his power to help the team be successful. Scrum Master also communicates regularly with the product owner and helps the product owner to maintain and upgrade the product backlog. Scrum Master is the only member of the development team that has any contact with the outside organization in the case the Scrum team is a part of a larger organization or company.

**Pros of Scrum**

- **Focus** – During a sprint there are no outside distraction and the product team can focus only on a handful of tasks that they set out to accomplish during the sprint.

- **Agility** – In a Scrum project the goals and the end product that the project is aiming to accomplish can change at any point during the development. Because of short iterations it is easy to add new features or goals at the end of each iteration.

- **Self-learning organization** – At the end of each Sprint there is a retrospective where the team gathers and then defuses new knowledge gained during the sprint.

- **Creativity and innovation** – Scrum sets guidelines on how team members should interact with each other. All Scrum guidelines base on better interaction between team members. This sparks creativity and innovation because team members are searching for solutions together.

- **Motivation** – Team members on a Scrum project have a high degree of autonomy. They have the power to set their own workload and the team itself is tasked with finding the solutions to the problems that a specific product development project has. In Scrum overtime is strongly unadvised due to the possibility of endless development cycles. Overtime can lead to team members burning out.
Cons of Scrum

- Preparing and educating a new team on using Scrum can lead to a very late start due to Scrum's many new concepts and work processes. Also, positive effects of Scrum such as higher productivity are usually not apparent with new teams.
- During a Sprint, the development team is left to organize its own work. The lack of a formal leader is usually a shock for team members that are used to different kind of organization structures.
- A Scrum development team should internally have all the necessary knowledge and experience to successfully finish a project. If a team encounters a problem that is outside their scope, than the project can get delayed.

2.2 Extreme Programming (XP)

XP is a lightweight, efficient, low-risk, flexible, predictable, scientific, and fun way to develop software. It is distinguished from other methodologies by (Kent Beck 1999, 9):

- Its early, concrete, and continuing feedback from short cycles.
- Its incremental planning approach, which quickly comes up with an overall plan that is expected to evolve through the life of the project.
- Its ability to flexibly schedule the implementation of functionality, responding to changing business needs.
- Its reliance on automated tests written by programmers and customers to monitor the progress of development, to allow the system to evolve, and to catch defects early.
- Its reliance on oral communication, tests, and source code to communicate system structure and intent. Its reliance on an evolutionary design process that lasts as long the system lasts.
- Its reliance on the close collaboration of programmers with ordinary skills.
- Its reliance on practices that work with both the short-term instincts of programmers and the long-term interests of the project.

The main goal behind XP is to reduce the underlining costs that are associated with high risks of sudden change that is common in development projects. Changes occur during the development phase of a project usually due to the fact of shortage of information at the beginning of a new development project and then gaining new insights and knowledge as the project progresses.

Pros of XP

- **Agility** – XP allows for changes to project goal and product features during the project life cycle. The iterative nature of XP allows changes that can be easily implemented with new iterations.
- **Working Prototype** – At the end of iteration in XP the product should be at the point it can be launched to the market. This is referred to as a working prototype. The first iteration should produce a basic working product and subsequent iterations should expend the feature set of the product getting it closer to the end initially planned product.
- **Quality** – XP proposes development techniques that are meant to drastically decrease the probability of code errors.
- **Transparency** – XP is known for implementing a great amount of transparency to a project which makes it easier for managers and organizations to oversee projects.
• **Feedback** – When developing a new product there is always the risk that the market will reject the end product. XP minimize this risk by forcing the development team to test their product at the end of iterations with potential costumer.

**Cons of XP**

• XP demands that the contractor of the project is an active team member of the development team and a constantly available to all team members of the development team.
• XP is only applicable if all team members can work at the same location. If this is not the case, XP is not advised.
• XP itself is a relatively young and still evolving methodology. So it is hard to recommend it for very complex development projects.
• XP is only suited for companies, organizations or project teams that adhere to a certain organizational culture. The organization has to be open to employee creativity, freedom and self-control and innovation.

Scrum and XP have common practices, but each metrology has some important fetchers of its own, which make it more stubble for mobile app development projects. By comparing both methodologies in Table 1, we can outline which practices and fetchers are more appropriate for mobile app development projects.

**Table 1: Comparison of Scrum and Extreme Programming (XP) methodology**

<table>
<thead>
<tr>
<th>Comparison criteria</th>
<th>Scrum</th>
<th>XP</th>
<th>Better suitable for mobile app development</th>
</tr>
</thead>
<tbody>
<tr>
<td>Organizational structure</td>
<td>Expects self-organization and self-management of the project team. Defines organizational milestones and events.</td>
<td>Expects self-organization and self-management of the project team.</td>
<td>Scrum</td>
</tr>
<tr>
<td>Team structure</td>
<td>Project team consists of:</td>
<td>Project team consists of:</td>
<td>Scrum</td>
</tr>
<tr>
<td></td>
<td>• Product Owner,</td>
<td>• Client,</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Scrum Master,</td>
<td>• Developers.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• Developers.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Optimal size of project team</td>
<td>4 - 6 team members</td>
<td>8 - 10 team members</td>
<td>Scrum &amp; XP</td>
</tr>
<tr>
<td>Project location</td>
<td>Recommend the team works from one location. There are some practices that show how to organize teams in separate locations.</td>
<td>Requires a tem to works from one location.</td>
<td>XP</td>
</tr>
<tr>
<td>Project life cycle</td>
<td>No clear project end. The project continues until it makes sense to continue for the client.</td>
<td>No clear project end. The project continues until it makes sense to continue for the client.</td>
<td>Scrum &amp; XP</td>
</tr>
<tr>
<td>Iteration cycle length</td>
<td>1 - 4 weeks</td>
<td>1 - 6 months</td>
<td>Scrum</td>
</tr>
</tbody>
</table>
Methodology applicability
Scrum can be used on all kinds of projects in large or small organization. Suitable only for software development projects in smaller organization with a specific organizational culture.  

Organization culture
Does not propose changes to the culture and values of the organization. Outlines in detail organizational changes and changes in values.  

Development processes
Lacks techniques and best practices needed for development. Describes in detail techniques and best practices of software development.  

Scrum

XP

Table 1 makes it abundantly clear neither agile methodology is perfectly suitable for mobile app development projects. A new methodology or an adaption of existing one is needed. We propose integration and slight modifications of both Scrum and XP.

3 Integration of agile project methodologies Scrum and Extreme Programming for the purpose of managing mobile app development projects

Using Table 2 we model recommended methodologies depending on the organizational structure for mobile development projects. Looking at the organizational structure as a whole helps us better understand the managerial challenges of a MA project especially on the organizational and operative level. The formation challenges can be successfully met using a combination of Scrum and XP.

Table 2: Organizational structure and methodological basis for mobile app development projects

<table>
<thead>
<tr>
<th>Organizational level</th>
<th>Tasks / Activities</th>
<th>Methodology</th>
</tr>
</thead>
</table>
| **Strategic level**  | Defining business goals and vision of the project.  
It is critical that a decision about platforms the proposed mobile app should run on is made. |  |
| **Organizational level** | Assembling a cross-functional project team, defining organizational structure and rules of operation. | Scrum |
| **Operational level** | Structuring development in an agile way. | Extreme Programming |
| **Level of quality control** | Obtaining appropriate certificates needed to list app on different app stores.  
User feedback. |  |

Source: own
On the highest strategic level management has to set a clear vision and business goals. Agile projects are structured in a way that delivers highest business value first. Goals are also needed so it can be determined what kind of mobile app will be developed. Mobile apps can be divided in three groups: Native apps, Web apps and Hybrid apps.

Once business goals have been set and it is clear what kind of MA is needed to accomplish set goals, the project team can be assembled. When assembling the team all business as well as development challenges need to be taken into account. The project team has to be small and has to be self-containing. The team has to be able to work together at one location. At this phase we recommend use of Scrum methodology for assembling and organizing a project teams.

During the development phase of the project it is crucial that Scrum events take place as predicted by the methodology. Scrum workflow is optimized for team focus and motivation. Where Scrum starts to fall short is not the organizational structure that the methodology proposes but the lack of concrete agile software development practices that are needed in the development phase. That is why we propose use of XP. In Table 1 we already established that the methodologies are very similar and can be successfully combined. XP defines agile development practices that work very well in the Scrum framework.

We would not be really talking about an agile methodology if did not consider short iteration cycles that allow for user feedback. Quality control has to be at the heart of the whole development so it has to be present at all levels. We define quality control as the feedback real users give at the end of each product cycle once they interact with the product completed in that cycle. User feedback is then considered in the planning stage of the new development cycle.

4 Managing mobile app development projects with combining Scrum and XP

Application of the integrated approach which combines agile methodologies Scrum and XP, as presented in this article was carried out and tested on two successfully completed mobile app development projects. The authors of this paper have been using the combined approach for a little less than a year. It is worth pointing out that we have never tried managing MA development projects using traditional software development methodologies that utilize the waterfall model. That is why it is not the goal of this paper to claim that the proposed combination of Scrum and XP is the only or the best way to manage MA development projects.

During the application of the proposed combination of Scrum and XP on the before mentioned projects we experimented with different combinations of Scrum and XP practices working together. After a few less successful attempts due to the initial lack of experience with Scrum and XP projects we discovered a balance between using both techniques on MA projects.

In the following chapter we describe the way we approached and integrated both Scrum and XP best practices into MA development projects.
4.1 Preparing Product Backlog

Product backlog is an essential and probably the most important document in according to the Scrum methodology and is also a prerequisite before starting a project. Product backlog is simply a prioritized list of feathers needed to achieve projects business goal. Product backlog is prepared by the product owner. Each feather is usually referred to as a story which presents a description of the functionality of the product. XP similarly to Scrum requires clients to come up with stories, but the stories are defines much broader. They are called "user stories". User stories represent a scenario or typical use case of how a user will interact with a particular feature of the product. The authors of this article suggest the use of user stories as defined in XP.

In managing MA projects we used a simple Excel spreadsheet as the main Product Backlog (PB) as shown in Table 3. PB was accessible by all team members at all time. We used the tool Dropbox to synchronize PB at all times.

Table 3: Product Backlog example

<table>
<thead>
<tr>
<th>MA name</th>
<th>ID</th>
<th>STORY NAME</th>
<th>IMPORTANCE</th>
<th>TIME</th>
<th>DEMO</th>
<th>COMMENTS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>User profile</td>
<td>30</td>
<td>8</td>
<td>Create a new user profile by going in to settings and entering all fields and a user profile picture.</td>
<td>User profile pictures should be 150 X 150px. Do not apply any styling to input fields.</td>
</tr>
</tbody>
</table>

Source: own

For each story in the spreadsheet we had prepared a physical card that helped with the visualization of work.

In Scrum methodology PB is the propriety of the Product Owner and as such now other team member is allowed to change it. In our experience we found is much batter and productive if all team members have the power to change PB but they have to consolidate with the Product Owner first. In our example all team members added suggestion to PB and then the Product Owner would approve ore discard the suggestions. For this reason we added a Comments field to the Excelle spreadsheet that allowed for new ideas and changes to be explained.

Well-prepared product backlogs are therefore more than just prioritized lists. Stories should describe business goals through scenarios which describe typical or expected user behaviour of the finished product. The technical aspects and the best implementation of product feathers should be left to the development team.

4.2 Sprint planning

Planning the next sprint is the most important event in Scrum methodology. Organization and execution of this event deeply affects the performance of work during the next sprint, and thus the success of the entire project.

The majority of the meeting is devoted to evaluation of user stories. The product owner assigns initial variables to each story. Each story contains three variables: scope, time estimate and importance, which are highly dependent on each other. Scope and importance are set by
the product owner. Estimate is set by the team. Scope defines how detailed a user story should be. Importance is assign in regarding to the ability of a user story to help achieve business goals. The higher the importance the sooner a user story should be implemented. Time estimate is the ideal time it would take to develop a user story.

During sprint planning, these three variables are continuously changed by team and the product owner. If a product owner wants to include more user stores in the next sprint he or she can change the scope or importance of a story. Bought Scrum and XP suggest that developers should determine their own work load. That is why time estimates must be determined by developer so they will be prepared to committing to the deadlines.

During the sprint planning meeting we had found the use of story card that we previously mentioned as a physical representation of the PB very useful. Story cards contain the three before mentioned variables, the name of the story and a description of the user story. During the meeting story card turn ideas and tasks in to something tangible that team member can hold on feel. This has an impact on creativity and team members are more engaged in the planning process.

For determining the time how long a particular story should take to implement we also recommend using a visualization technique. The technique is called planning poker. Each team member gets a set of thirteen playing cards. For every user story each team member chooses a playing card with a number that represents the time he or she thing is needed to implement the story. Time is represented with points and not the actual time values as is predicted by Scrum. For each story all team members simultaneously show the card that they think represents the time needed for successful implementation. This ensures that all answers are impartial and are not influenced by other team members. If there are big differences in the estimates between team member farther discussion about the story is needed. This is a clear sign that team members have different understanding of the goal that is trying to be accomplished with the story. After discussing the story we repeat the wound of poker. We keep repeating until all the estimates are not the same.

4.3 Preparing Sprint Backlog

Sprint Backlog is a prioritized list of user stories included in the next sprint. In the preparation phase user stories get broken down into tasks. Traditionally sprint backlog is a document which must be accessible to all team members. It represents a reference document for all developers involved in the sprint and helps unify team’s vision about the sprint. Team members should be able to see sprint backlog any time. We've noticed that this cannot be achieved with an electronic or paper format of a sprint backlog. By using a wht board or simply a wall instead of paper or an electronic format as a sprint backlog we achieved better sprint results.

To create the wall format we create a table with four columns. Each story card is placed on a new row with in the table. Story cards are sorted by their importance ranking from top to bottom. Once a team member stars work on a story he or she physically places the task into the next column. Task and stories move from the most left column to the “done” (3ed) column. The very last column is wider as it is meant for a burn down chart. The burn down chart visually shows the rate and trend of progress. The horizontal excise represents time and the vertical excise represents the number of stories included in the sprint. After each daily stand-up scrum meeting a team member adds a new point in to the chart depending on the team’s progress. Underneath the burn down chart we integrated two very important concepts: unplanned items and next items. Next items are all story cards that were not selected for this
particular sprint. This is done as motivation and a reminder that the development team cannot rest if they start to feel they will complete all sprint tasks before the end of the sprint.

Using a visual representation of the sprint backlog changes a team’s work dynamic in a positive way. Suddenly there is a focal point where team members can brainstorm and exchange ideas. All team members instantly see updates to the project and communication between team members increases.

4.4 Daily Scrum meeting

Scrum methodology defines the requirement for a daily 15-minute stand-up meeting. This is an important event, which should take place every day at exactly the same time and in the same space preferably in front of the wall with the sprint backlog.

The purpose of Scrum meeting is that each team member answers three questions:

1. What have you done since the last Scrum meeting?
2. What obstacles did you meet?
3. What will you do until the next Scrum meeting?

We experienced best result in Scrum meetings when team members used story cards and physically updated the sprint backlog as they answer the three main Scrum meeting questions. As the last agenda of the meeting a team member should update the burn down chart. We recommend that after every meeting a different team member updates the chart. This ensures all team member understand the chart and know the rate of progress.

4.5 Managing the development process

Both presented agile methodology requires that management distance itself from the development. But XP also describes in great detail agile software development techniques. This techniques guaranties best results when they all work in synergy. It is worth mentioning that some of XP development techniques are very complex and are hard to adopt spicily for inexperienced teams. Dilemma occurs if it is even worth adopting XP if the team will not be able to achieve expected potential productivity due to lack of experience. From our experience adopting even some of XP’s development techniques will have an impact on productivity and quality of MA. Using XP the development process becomes more structured and transparent.

In this paper we will only mention agile XP development techniques. An in depth analysis on how to adopt this techniques in MA development is too extended to include in this discussion.

XP recommend using the following agile development techniques (Beck 1999, 47):

- Small releases - Put a simple system into production quickly, and then release new versions on a very short cycle.
- Metaphor - Guide all development with a simple shared story of how the whole system works.
- Simple design - The system should be designed as simply as possible at any given moment. Extra complexity is removed as soon as it is discovered.
- Testing - Programmers continually write unit tests, which must run flawlessly for development to continue. Customers write tests demonstrating that features are finished.
Refactoring - Programmers restructure the system without changing its behaviour to remove duplication, improve communication, simplify, or add flexibility.

Pair programming - All production code is written with two programmers at one machine.

Collective ownership - Anyone can change any code anywhere in the system at any time.

Continuous integration - Integrate and build the system many times a day, every time a task is completed.

40 hour week - Work no more than 40 hours a week as a rule. Never work overtime a second week in a row.

On-site customer - Include a real, live user on the team, available full-time to answer questions.

Coding standards - Programmers write all code in accordance with rules emphasizing communication through the code.

4.6 Sprint review and demo

At the end of each sprint, especially in the first few sprints, it may happen that the end result is very uninteresting. The question may arise is it even necessary to do a final demo. We advise that whatever the outcome of a sprint it is always necessary to carry out the final demo. Here are some reasons for carrying out a demo at the end of each sprint (Kniberg 2007, 75):

- The development team gets some much deserved thanks which affect the overall team motivation and satisfaction.
- Other Scrum teams can attend the demo and see the results. This is especially important on larger projects where there is ore then one Scrum team.
- A final demo usually provokes priceless feedback.
- A final demo is also a social event which gives the development team a chance to meet new people that can offer a fresh prospective on the project.
- Doing a demo forces the team to actually finish work and release it (even if it is only to a test environment).

4.7 Sprint retrospective

After the end of each sprint it is necessary to carry out a sprint review. The goal of the meeting is to review events that took place during the sprint. Team members express what they though worked well and what mistakes of a sprint were. The end result of the meeting has to be a proposal of improvements for the upcoming sprint.

5 Conclusion

Clear advantages and compatibility of agile methodologies Scrum and XP when applying them to mobile app development projects where the basis for the idea of adapting and integration both methodologies. Joint use of methodologies Scrum and XP as proposed in this article was used on several mobile app development project by the authors. It is necessary to emphasize that the purpose of the article is not to advocate the proposed joined use of agile methodology as the only appropriate methodological basis for managing mobile app development projects. That is why we do not compare or analyze other possible methodologies.
The goal of this article is simply to show a possible way of combing two very well documented and popular agile methodologies in a way it maximizes project efficiency based on mobile app development project characteristics. We conclude that using Scrum’s best practices as an organizational framework which is then expanded with the detailed development practices of XP is a suitable way of structuring mobile app development projects. Furthermore we explain how and when to use certain best practices outlined in agile methodologies Scrum and XP. As Scrum’s author J. Sutherland likes to put it: “No two Scrum projects are the same”. We fully understand that some of the practices explained in this article cannot be applied to all mobile app development projects, but we hope we at the very least provided some inspiration for new ways of solving challenges mobile app development brings to the table.
6 References


7 Acknowledgements

1. Dr. Igor Vrečko for outstanding mentorship and guidance.