# Implementation of SCRUM methodology in programming courses

Matej Madeja*, Miroslav Biňas†
Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
*matej.madeja@tuke.sk, †miroslav.binas@tuke.sk

***Abstract*—This paper presents suggestions for creating a syllabus of programming courses, especially focusing on the development of mobile applications. At the same time, an agile method is proposed to implement the learning process to bring benefits for students. On the basis of a comparison of student results from previous runs of the particular course, this paper confirms the improvement of students' motivation by using SCRUM methodology with combination of automated testing towards classical learning methods. At the same time, authors present personal experiences and suggestions for use in other programming courses.**

## I. Introduction

Agile methodology is currently the most popular iterative software development method [1] in practice. This methodology has shown up the most feasible for software development purposes over the last few years, both from the customer's point of view and from the supplier's point of view. This is confirmed by companies around our university, as well as recommendations by international software giants [2], universities [3], [4] and many researchers in the world. Faster software development, bigger business value and lower development costs, i.e. higher yield, are the reasons why companies look for experienced developers who are experienced in SCRUM methodology and are able to use them in practice.

One way to learn the software development methods used in the practice for young developers and, from our viewpoint, the most beneficial, is to use a particular method within a university course. Our experience is that students come to a new practice environment after graduating from university and are forced to use development methods that they were not familiar at university. They feel a big gap between their approach to software development and approaches used by the real employer, based on previous research. From our experience, we started to use the version control system *git* a few years ago which is mostly used by companies around our university and today it is a rich experience for our students before their real job in a company. IT companies in our area demand experienced programmers [5]. It is the role of the university to ensure a continuous improvement of courses with respect to industry so to educate usable engineers for software companies.

Our goal is to effectively implement agile methods into programming courses. Because SCRUM is currently the most popular agile method in 2018 with 56% market share on its own and 70% when combined with other methods [6], we will focus on it. Although this research issue is currently being addressed by a number of researchers, it is difficult to implement SCRUM in a course with all the principles appropriate for use in a real development environment. One of the basic features of SCRUM is teamwork, on the other hand, if software development methods are not the main subject of teaching, and the student needs to learn a new programming language or programming for a particular platform, teamwork is not always suitable. According to our experiences in a student teams often work only a few students and others drive alongside them and do nothing.

Not every programming course is suitable for using agile methodology. Because of that it is necessary to choose a language and technology so that the student is motivated enough. Mostly the developer in the company is motivated by earnings. Motivating a student is often a problem, because only a few students in the group can be excited about the problem solving, others do the necessary minimum. For us, the agile method is a partial solution to this problem, while an appropriate course and level of requirements for students are needed to motivate as many students as possible.

Smart is today very commonly used word - smartphones, smart homes, etc., and because this topic is "in", students should also be interested. Mary Meeker's report [7] from May 2017 points that the average American adult spent 3.1 hours per day on a mobile device in 2016. We can assume that this is very similar in the whole world and this number is growing yearly. Among other things, the report shows that since 2011 Android is the most used mobile operating system and from April 2017 until today has the most market share among all operating systems in the world scale [8], followed by desktop Windows OS. This is the reason why MOOC (massive open online course) are so popular in this business (see [9], [10], [11]).

Responding to this situation, our university launched course *Application Development for Smart Devices* in 2015 with a major focus on the Android platform and today is attended by more than 150 students. We considered this course to be suitable for implementing the agile methodology at our university. Until today 3 semesters of the course were conducted, where the first 2 runs were followed by the classic non-SCRUM syllabus. In the last, i.e. 3rd run of the course, we used SCRUM method to organize students. To don't keep things out of reality, we have consulted the course syllabus with the *Wirecard* company, which mostly focuses on the development of applications for smart devices and employs many graduates of our university and is able to describe the largest student gaps.

This paper summarizes existing research of ways to apply agile methodologies to the learning process and suggests a new way of SCRUM implementation in the course background, with the main purpose of teaching programming. At the same time, it proposes appropriate tools for course material creation, evaluates the experience with proposed solution and describes suggestions for future use.

## II. State of the art

In March 2018, Sharp and Lang [12] presented their view of agile methods from a pedagogical point of view in programming courses. They analyze articles dealing with this issue, comparing articles used by Agile pedagogy to teach content that is not about Agile methodology itself and articles that use Agile pedagogy to teach students about Agile methodology. The authors offer the idea that agile methodology in the programming courses not only motivates students to work but also improves and makes attractive the course itself and improves its materials, too.

Linden, in her article [13] devoted to fitting SCRUM in self-regulated learning in an introductory programming course within the *Doubtfire* learning management system (LMS). She argues that despite the agile method used in the course it did not improve motivation of disinterested students and did not have any positive effect on the ratio of pass / fail rates. But in our case we do not want to increase pass / fail rates. The main motivation for the student will be a successful passing of the course and this will only be possible in case of an iterative work, so we will achieve that by mandatory submissions of sprints to the automated testing environment. By developing one application throughout the semester student will be more experienced in agile development. If the previous part of the assignment is not met, the student will still have to complete it, since it is followed by other parts of the application depending on the previous ones. From this point of view we use a totally different approach for motivation.

Mahnic [14] in 2010 implemented a SCRUM methodology in the way that students worked in a team on a real project. His test sample consisted of 31 students and one of his results was confirmation of the hypothesis that students preferred learning through practical project rather than formal lectures and that students liked working in the SCRUM team. In his article, however, he focused on perceptions about Scrum generally. We focus exclusively on use in the programming course, not general developer perceptions. Additionally, the number of students in our course is 5 times greater and as stated in the previous Section, we try to eliminate the students' misbehavior in the team by individual responsibility for the product. So we try to create a SCRUM environment but with the primary goal of teaching each student separately.

There are a few case studies that focus on integrating SCRUM methodology into Android courses. Reichlmayr's [15] in his conclusion claims that SCRUM helped the students in requirements engineering, project planning and tracking, testing and effective team collaboration. These characteristics are evident from the very nature of agile methodologies and these results have not been quantified, therefore, it is rather a guess. In our case, we do not want to quantify the efficiency of the students work, but we want to increase the qualitative level of the course and we expect a better level of student knowledge at the end. The level of knowledge of students can not be quantified unequivocally.

A very similar approach of SCRUM implementation into a course tried Rover et al. [16], which used SCRUM for real application for Android platform, but for a two-semester senior design course. In our case, most of the students will be beginners in Android programming. Another difference is that in their approach was intended to learn students SCRUM methodology using the Android platform in which they are already experienced. We try to do the other way, we would like to learn students program Android application with SCRUM methodology support. In conclusion for the future, the authors claim that it is still necessary to adapt course curriculum to agile methods, that is what we are very much concerned with in this article.

The field of programming teaching and assessment of student assignments is also devoted to many authors. Pecinovský in his book *OOP - Learn Object Oriented Thinking & Programming* [17] from 2013 and in his article [18] defines the *Architecture First* methodology, which states that the student should first understand the basic concept of a particular platform, then design the architecture of solution, and finally start writing the source code. According to Pecinovský, we have a lot of coders and a few software architects today.

## III. Application Makacs

In the first 2 runs of the selected course the same *Makacs* application was developed as an assignment by every student, focused on sports monitoring and tracking. The goal of the app was to familiarize students with the basic programming principles of Android platform:

1) the basics of creating an application design,
2) own and system activities,
3) work with intents,
4) running background tasks,
5) creation of own services and use of existing
6) *Google API* and *Google API* developement console.

Creating the main application *Makacs* in 2015 was within 4 labs, which means it was less difficult. Students had only a simple description of elements required for the main activities together with basic functionality. We left the other functionality for students' creativity because we wanted to observe as many different designs and ideas as possible, so we collected the most interesting ideas. Among other labs students had to implement smaller simple applications that together with *Makacs* formed their overall course result.

In 2016, we saw the greatest potential in *Makacs* app among other apps in the course, so we chose it as the only assignment for students for the entire semester, with the following requirements:

1) min. 5 activities, of which at least 1 will include calorie counting,
2) min. 1 service (recommended for counters - duration, pace, distance, calories),
3) min. 1 broadcast receiver,
4) min. 1 activity with own list implementation,
5) data persistence with SQLite[1]
6) communication with an external web service through the API,
7) min. 2 language variants,
8) min. 1 custom extension.

The assignment was more precisely defined to unify the evaluation conditions for all students. The app have to be targeted for Android devices with API 19+, have to fulfill stict requirements (eg. service implementation, main activity logic) and did not have any other implementation restrictions. It only had to meet the requirements and features had to be realistically usable. The originality of the solution was assessed, of course, by teacher's subjective opinion. At the same time, we tried to prepare a sample application for automated testing for 2017 run in the background. Main problem in 2016 was that most of the assignments were made on the last moment, as only one deadline was defined. This was one of the main teaching reason why we wanted to implement agile in the course.

## IV. Research questions and methods

Based on the Section II, we conclude that there are many solutions and insights on syllabus creating via SCRUM method. However, none of these studies focus on the implementation of SCRUM in a way that the student focuses primarily on the main subject of the course, programming, and perceives the chosen method of development only subconsciously, albeit modified to achieve higher learning priorities. As mentioned in Section II, most of the students have been working in a team, which improves their collaborative skills, but their programming skills decrease (see Section I), as programming works often do not distribute between themself and therefore they can not be evaluated equally and fairly. In addition to the design of the syllabus, we also implemented an automated testing environment, which is presented in more detail in our article from July 2018 [19].

The following research questions are addressed in this paper:

1) Is the use of the *Architecture First* method in the course appropriate and how to implement it?
2) How to include agile in the curriculum and how does it affect automatic testing?
3) What are the most common mistakes of students in developing mobile apps and how to avoid it?
4) What tools use to present requirements and how to communicate with students?

The answers and possible solutions to these questions are listed in the following sections.

### A. Architecture First

The selected course does require one prerequisite, Java beginner experience, which is included in the study plan by a mandatory Java course. Students attending the *Application Development for Smart Devices* course are mostly not experienced in developing smart devices.

In 2017 we selected as the main course assignment *Makacs* again. With regard to the requirements for programmers in practice around the university, one complex assignment is more convenient, as the student have to, in addition to programming, also address the sustainability of the code, even with bigger changes in requirements. The application had the same requirements as in the previous course runs (see Section III), but the assignment was moderately limited due to the automated tests executing in the background of the course as a supplement to the students' assessment.

During the first 2 semesters of the course (2015, 2016), submitting and testing of assignments was done only manually. The student submitted the assignment to teacher while submission was monitored by another one. This way we tracked how students understood course topics and their own solution. In this experiment, we found that a large proportion of students did not know the basic concepts of the Android platform and their context. Due to this fact we were looking for the cause in the first seminar of the course, where platform concepts have been explained only theoretically and students have met with them only during programming. Pecinovský's *Architecture First* methodology solutions appeared to be satisfactory.

We created an Android application called *TryMe* that leads the student as a guide to the introductory lesson of the course. Installed application (virtual or real device) does not have a launcher or any controls. The application can only be controlled using command line tools of Android OS, such as activity manager, package manager, etc., using the *adb shell*[2]. In this way, students will practically understand the basic concepts of Android apps, how to control them and how they communicate. Application includes activities and sending extra parameters between them, notification services and broadcast receiver. This app is used in *Sprint 0* (see Section VI).

### B. Agile development

During the first year of the course labs were mandatory, but after the first semester we noticed that the seminars were not used efficiently. In 2016 we have decided to realize only consultative seminars, with an emphasis on the creativity of the students in the assignment. This approach has quite proven, so we will try to keep it. However, if we wanted to test the

[1]https://www.sqlite.org/

[2]Android Debug Bridge, http://adbshell.com/

Table I
MOST COMMON STUDENT'S ISSUES IN MOBILE DEVELOPMENT.

| Issue | Failed Applications |
|---|---|
| Navigation & UX | 7 |
| Data loss on activity restart (e.g. device rotation) | 6 |
| Logical problems (e.g. usage of device variable data) | 3 |
| Unable to start or install app | 2 |
| Forgotten services at activity restart | 1 |

assignments by an automated system, we need to define more precisely the application functionality. And it would be good if the requirements will be povided to students in the way the surrounding IT companies use in practice. At the same time, in 2016 students often made a last-minute assignment because only the last deadline was mandatory. With new organization of the course, we would like to motivate students to work on a continuous basis and in shorter sprints [20].

Passing of the results of the student's work continuously is very similar to Continuous Delivery approach, which is closely related to agile methods of software development. According to Silverthorne [21], SCRUM makes up 56% of agile methods in practice. Scrum is popular especially thanks to its productivity, management and motivation for team members. Wirecard also confirmed the usage of this method in practice. Finally, in other courses at our university we use SCRUM (but in classical team members cooperation), so that were the main reasons why we decided to use SCRUM in the course, too.

After all, we want students to be experienced with all the basic Android application development issues, so students will not work in teams. It will not be pure SCRUM, yet we want to start using SCRUM artifacts, such as user stories, sprints, eposes, versions and themes (more in [22], [23]). The course has a duration of 10 weeks, with 5 seminars organized in 2-3 week sprints. As was mentioned, the first seminar is organized in the form of guide, the other seminaries will be in the form of sprint planning and retrospective.

### C. Most common students' mistakes

In generally the most common Android app failures are the following [24]:

1) Application failure when installing.
2) Application crash during execution.
3) Problems with scaling or deploying elements on the screen.
4) Unresponding application from unavailable sources.
5) Problems of viewing content in landscape or portrait mode.

These are all common issues for apps that people hate the most. In terms of teaching, however, many other problems arose. We chose 10 random student apps from the course in 2015 and 2016 and tested them manually. Testing took place on the basis of *Wirecard*'s recommendations and teaching experience. The results of testing are shown in the Table I.

Testing was carried out on 2 devices:

1) Nexus 5, API 25, Android 7.1.1, emulator.
2) Prestigio 5453 DUO, API 19, Android 4.4.2, real device.

For a detailed test procedure please refer to our article about automated testing evitoment [19]. According to this testing results we adapted the curriculum to avoid found mistakes.

## V. AGILE TOOLS AND COMMUNICATION

Although the app will not be created by a team but only by the students individually we can use the same tools to create materials for them. The terms wireframe, mockup and prototype are often confused and used without sufficient knowledge of the semantics of these terms. We choose only the right ones for teaching purposes.

### A. Prototyping tools

In a real-life development team, it would be useful to use the procedure described by Warcholinski [25], who present stages of the typical design representation development journey: $Sketch \rightarrow Wireframe \rightarrow Mockup \rightarrow Prototype$, but in the course we want to motivate students to creative activity. When delivering a detailed prototype with a precise design, we would be able to suppress this motivation. At the same time, the students expect the required functionality, which can be generally tested, so the prototype must simulate the expected behavior of the application.

In order to preserve student creativity, we will use wireframes with a combination of sketches. Sketches usually contain only a basic preview of the initial idea of the application, wireframes already show a simple structure of the application. As we want to give design freedom to students, we will create sketch-style wireframes, without particular layout requirements. For this purpose we will use the *Balsamiq*[3] tool to create wireframes which has the ability to create sketch-like mobile screen designs.

Another important part is to show students the basic functionality of the application that will be required, preferably in the form of live prototype. Since from a prototype is already expected a precise design, we will adapt it in the requirements of the course by demonstrating the prototype using wireframes. For these purposes, we will use the *InVision*[4] tool, which was the most accessible tool in our small-scale comparison [26].

### B. Communication with students

As the assignment will be presented to students in a short time period, they will not be able to ask questions about the development requirements immediately. That is why we needed to choose a suitable communication tool. Since we searched for the simplest solution that is free, we chose *Slack*[5]. In *Slack* students will be able to communicate together as a team creating multiple similar applications, advise to each other, make suggestions, and at the same time communicate with the product owner, who will be mostly a teacher or a developer from a real company.

[3]https://balsamiq.com/
[4]https://www.invisionapp.com/
[5]https://slack.com/

## VI. Sprints proposal

A sprint seminar will come to present a worker from an IT company or a teacher will be appointed. Only these seminars will have a presentation form, so the product owner will present his/her requirements and the student's task is to implement them. Other seminars will be a voluntary in form of consultation. The created curriculum content will contain 5 main topics:

### *A. Sprint 0: Introduction to the Android world (guided seminar)*

The first lab is realized in the form of instructions, as it can take place before the first lecture. We use the zero in the sprint numbering because it is not a sprint in the true sense of the word - it is only a teacher-guided introduction into *Android* development principles, specifically about activities, intents and services. The first meeting has a cognitive character and the duration of the sprint is 1 week.

By completing the lab tasks the student will be introduced to the *Android Studio* IDE and virtual devices using the *AVD Manager* (*AVD = android virtual device*). At the same time, the tutorial is focused on controlling virtual devices from the command line, especially using the *adb* (*Android Debug Bridge* tool).

To get a student familiar with the *adb shell* and the *Android* application basic components a simple *TryMe* application has been created to achieve *Architecture First* design. The installed application is without UI, so you need to use the *adb* tool to control the application and understand the meaning of the activities, services and intents. The application contains 1 activity with the a broadcast listener, 1 activity with an extra-parameter activity and 1 notification service.

Objectives: IDE, emulators, *adb*, device console, events, app components.

### *B. Sprint 1: Introducing app Makacs*

Since that lab every other will have a form of *SCRUM* sprint planning. *Product Owner* introduces the product in the form of a presentation and students have a live wireframe prototype that has been created the with *Balsamiq* and *InVision* (Section V). Example of a sample wireframe design is shown in the Image 2. Sprint has a duration of 2 weeks.

Sprint tries to teach students how to create a new project, activities, work with the *Android Studio* interfaces and create custom app layouts. The student is required to work with the *Google Developer Console* interface that is required for mapping activity. The main requirement for this sprint is to create custom application design, create layout files with required identifiers (needed for automated testing) and translations of multiple-language text. In the requirements, students also have to provide screenshots of the required emulator activities to practice with *adb*. Own application icon is also included in the design requirements of the sprint.

Objectives: new project, activities, UI components, *Google Developer Console*, app design, work with emulator, translations.

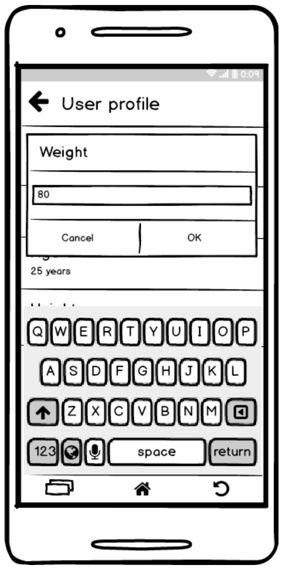

Figure 1. Example of the wireframe application design.

### *C. Sprint 2: Monitoring of sport activity*

Sprint has a duration of 3 weeks and focuses on the main functionality of the application. Students in the first steps create help classes from a pre-prepared class skeleton. Pre-prepared class files contain the constants needed to implement the required static methods. Classes are `final` inspired by the `java.lang.Math` class, and will be used multiple times in the application from different places.

Another part of the sprint is to create the main monitoring service and implement regular data distribution for *broadcast* listeners. Students then have to create a *broadcast* listener in the main activity that will receive the data and update the UI. The last requirement is the navigation between activities with extra parameters sending. Already in this sprint application stability and user experience (UX) is tested.

Objectives: calories count, help classes. services, activities implementation, broadcast receivers, intents, sharing with 3rd party apps, *Google Maps API.*

### *D. Sprint 3: Data persistence*

This lab radically changes the appearance and usability of the application. As it is usual in practice *product owner* has changed his requirements, so it will introduce the major structural changes that students will need to adapt. They will have to refactor already created source code. Product owner will introduce a new live wireframe prototype. This change should alert students whether they can maintain their solution even in case of unexpected changes. Sprint takes 2 weeks.

Sprint requirements include adding application menus and lead the developer to fragments use. Students will also create their own list type adapter and will use it in the application. The most important focus is to store data in a local database and synchronize the local database with the backend server and vice versa. Other requirements include the implementation of an application login form using the *Google* account and implementation of user and application settings.

Objectives: fragments, menus, own list type, ORM[6] and database, authentication, settings, database sync with 3rd party service.

### E. Sprint 4: Advanced features

The latest sprint is focused on student creativity and lasts for 2 weeks. *Product owner* finely circumscribes the requirements to the student but much less compared to previous sprints. The focus is on personalizing the design for different device resolutions, adding a notification management service, using *text2speach*, implementing a training plan, and signing in to an app through external services. All mentioned features are required. The last task is to create an original application functionality. This sprint is tested only manually by teacher because each student has the task of creating a unique feature of the application with the third-party libraries

Objectives: third-party libraries, design responsiveness, notifications, training plan, *text2speach*, multi-authentication, own original functionality, added app value.

### F. Usage of agile artifacts

In the first sprint the student will be familiar with the app *personas*. This will help determine the users for which the application will be created, and the student's task will be to satisfy all targeted users. Thus, in a particular sprint, the student will not have a detailed description of features he does not have to think about as in a classical course curriculum, but he will get a list of personas, and for every user type he will have to adapt each application functionality to satisfy him.

Requirements for students will be written in the form of user stories. They will ensure that students have to think about solving the problem based on a specific use case. Based on these, students will need to divide the requirements form user stories into epics and themes which are part of agile development.

Given the proposed SCRUM method, it was necessary to design a test platform so that the tasks of each sprint could be tested separately. From all mentioned sprints from previous subsection, we needed to create 3 test suites (except 0th and 4th sprint). The tests of 4th sprint is identical to previous one to ensure that the functionality of previous submitted solution is maintained after the implementation of extended functionality.

## VII. Results

First we looked at how much success we can expect from the *Architecture First* method used in the course, especially with a focus on understanding the basic concepts of Android (Sprint 0). We included 2 groups of respondents in the test sample of 159 respondentes:

- Android OS users without programming experience, not attending the course periodically (6 respondentes).
- Developers without any experience with mobile application development, course attenders(153 respondentes).

[6]Object-relational mapping.

Table II
Understanding of basic concepts of Android apps in Sprint 0.

| **Respondent type/ number of students** | **Awarded points** | | | | |
|---|---|---|---|---|---|
| | **1** | **2** | **3** | **4** | **5** |
| Non-programmer, Android user, non-course attender | 1 | - | 1 | 3 | 1 |
| Course attender | 5 | 9 | 33 | 63 | 43 |

In the experiment, their task was to fulfill a complete guide from *Sprint 0*, where students worked with app *TryMe* through the *adb shell*. Then they should evaluate how they understood the basic building blocks of Android apps. The results of their answers are shown in the Table II (5 points = absolutely understand, 1 point = not understand at all). Respondents were between the ages of 18 and 42.

The results in Table II show that most of the students comprehend more than good (4/5 points) which means that their feeling of comprehension is above average. The average score of the questionnaire was 3.83 points which is generally a very good result that confirms the suitability of using *Architecture First* in the course despite the modest adjustment of this method due to the nature of the course. We can assert that 94% of the students have learned matter at 3 points and more which is an excellent result.

As mentioned before the results from the questionnaire show only the subjective feeling of the student. Therefore, the course teachers compared the level of understanding of the basic concepts of the Android platform during personal assessment of assignment to the teacher. Teachers compared the level of student comprehension in 2016 and 2017 run of the course in the discussion. In 2016, the level of comprehension the basic Android concepts was roughly 2 points from a teacher's point of view (poor comprehension). In 2017, after teaching using the *TryMe* application the level of understanding was 4 points which confirms the results obtained in the questionnaire.

We also compared student results before SCRUM use in the course. In 2015, the worst student had 71% assessment (see Figure 2), indicating that students are supernumerary above average or the course assessment is too simple. At the same time, before using SCRUM, students had only one submission so it was relatively easy to get the required number of points at once. It is unrealistic to have so many over-average students' assessments because the average should cover the majority, the above average should be something special.

If we compare the results from year 2015 with 2017 the students' assessment distribution has been normalized (see Figure 2). Most of the students are in the middle of the assessment scale what is confidential information about students' results. At the same time, the rating is more trustworthy because result is created by several submissions provided by SCRUM methodology practices. The automated tests which ran at the background of the course also helped to assess students by fair-play rules.

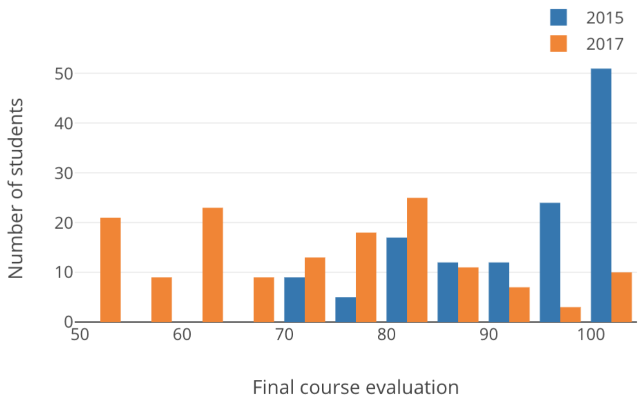

Figure 2. Final course assessment distribution in years 2015 and 2017.

## VIII. Future work

Improving the test process is an endless loop that can not be stopped. Based on the 2017 experience in the next semester of the course we will try to increase the teaching/guided part of the course. Besides iterative development, we want to use other labs to preview Android platform from different viewpoints, such as design options, working with sensor, or work with other device types like *Android Things*. In addition, we still consult our course suggestions with *Wirecard* company whose proposals are very valuable, especially for our future graduates.

## IX. Conclusion

This experimental investigation has suggested how to create a curriculum for a programming course, specifically for the development of the Android application course. The results we obtained with the questionnaire indicated that using *Architecture first* method would be appropriate for beginners. Since the responses were based on the subjective feeling of the student immediately after learning, we observed and evaluated the understanding of the basic principles of the Android platform at the end of the semester during manual submission of assignments. In this observation, we came to the conclusion that the understanding of these principles was approximately the same as in the completed questionnaire. We therefore recommend this method for use in the beginners programming courses.

At the same time, we evaluate the multi-annual learning experience in an agile course creation and suggest recommendation for individual programming of student if the agile method is not the main subject of the lesson. This will ensure students' focus on the main subject of the course. The iterative development with SCRUM coupled with an automated testing environment of assignments for each sprint separately motivates students to work continuously. At the same time, we were able to make the course assessment more effective which is visible on the student results where the most of the students are in the middle of the rating scale. In the previous course runs we had supernumerary above average students and this data were far away from reality. Paper brings many suggestions directly from practice thanks to the *Wirecard* company.

We also looked at the most common mistakes that students have made in their solutions in previous years of the *Application Development for Smart Devices* course. Based on these mistakes, it is possible to focus the test cases more accurately and appropriately adapt the course curriculum.

## Acknowledgment

The authors would like to thank all the participants in the research, especially students and teachers of *Application Development for Smart Devices* course. At the same time we thank *Wirecard* company for sharing their research results, experiencies with moblie testing and any other methodical help. This work was supported by the KEGA grant no. 047TUKE-4/2016: Integrating software processes into the teaching of programming

## References

[1] V. Mahnic, “Scrum in software engineering courses: An outline of the literature,” vol. 17, pp. 77–83, 01 2015.
[2] S. Denning, “Succeeding in an increasingly agile world,” *Strategy & Leadership*, vol. 46, no. 3, pp. 3–9, 2018. [Online]. Available: https://doi.org/10.1108/SL-03-2018-0021
[3] Z. Masood, R. Hoda, and K. Blincoe, “Adapting agile practices in university contexts,” *Journal of Systems and Software*, vol. 144, pp. 501–510, Oct 2018. [Online]. Available: http://dx.doi.org/10.1016/j.jss.2018.07.011
[4] R. Bojorque and F. Pesántez, “Curriculum design based on agile methodologies,” *Advances in Intelligent Systems and Computing*, vol. 785, pp. 84–94, 2019, cited By 0.
[5] M. Madeja, “Innovative approaches in introductory programming courses,” Master's thesis, Technical university of Košice, 05 2015.
[6] R. Hoda, N. Salleh, and J. Grundy, “The rise and evolution of agile software development,” *IEEE Software*, vol. 35, no. 5, pp. 58–63, September 2018.
[7] M. Meeker, “Internet trends 2017,” CODE CONFERENCE, Tech. Rep., 05 2017.
[8] StatCounter, “Operating system market share worldwide,” 10 2018. [Online]. Available: http://gs.statcounter.com/os-market-share/#monthly-201101-201809
[9] M. Cook, “The 50 most popular moocs of all time,” 04 2015. [Online]. Available: http://www.onlinecoursereport.com/the-50-most-popular-moocs-of-all-time/
[10] Coursera Inc., “Courses and specializations,” 2017. [Online]. Available: https://www.coursera.org/courses?_facet_changed_=true&domains=computer-science&languages=en&query=android
[11] edX Inc., “Android courses search,” 2017. [Online]. Available: https://www.edx.org/course?search_query=android
[12] J. Sharp and G. Lang, “Agile in teaching and learning: Conceptual framework and research agenda,” vol. 29, pp. 45–52, 03 2018.
[13] T. Linden, “Scrum-based learning environment: Fostering self-regulated learning,” *Journal of Information Systems Education*, vol. 29, pp. 65 – 74, 06 2018.
[14] V. Mahnic, “Teaching scrum through team-project work: Students' perceptions and teacher's observations,” vol. 26, pp. 96–110, 01 2010.
[15] T. Reichlmayr, “Working towards the student scrum - developing agile android applications,” 2011.
[16] D. Rover, C. Ullerich, R. Scheel, J. Wegter, and C. Whipple, “Advantages of agile methodologies for software and product development in a capstone design project,” in *2014 IEEE Frontiers in Education Conference (FIE) Proceedings*, Oct 2014, pp. 1–9.
[17] R. Pecinovsky, *OOP - Learn Object Oriented Thinking & Programming*, ser. Academic series. Tomas Bruckner, 2013. [Online]. Available: https://books.google.sk/books?id=xb-sAQAAQBAJ
[18] R. Pecinovský, “Methodology architecture first,” vol. 5, pp. 107–114, 04 2013.

[19] M. Madeja and J. Porubän, “Automated testing environment and assessment of assignments for android mooc,” vol. 8, pp. 80–92, 07 2018.

[20] R. Pecinovský, “Zadávání a vyhodnocování úkolů při výuce oop,” in *Počítač ve škole 2007*, Amaio Technologies, Inc. Nové Město na Moravě, Czech rep.: Počítač ve škole, 2007, pp. 1–4.

[21] V. Silverthorne. (2015, 08) How popular is the scrum process? [Online]. Available: http:// searchsoftwarequality.techtarget.com/photostory/4500251249/Vers ionOne-surveys-Agile-development-for-2015/5/How-popular-is-the-Scrum-process

[22] D. Radigan. (2016) Epics, stories, versions, and sprints. [Online]. Available: https://www.atlassian.com/agile/delivery-vehicles

[23] S. W. Ambler, “Going beyond scrum,” in *Disciplined Agile Delivery*. Disciplined Agile Consor, 10 2013.

[24] guru99.com. (2017, 08) Complete guide to android testing & automation. [Online]. Available: https://www.guru99.com/why-android-testing.html

[25] M. Warcholinski, “What is the difference between wireframe, mockup and prototype?” 04 2016. [Online]. Available: https://brainhub.eu/blog/difference-between-wireframe-mockup-prototype/

[26] M. Madeja, “Testing of applications for os android,” Master’s thesis, Technical university of Košice, 04 2017.