Training Scrum with Gamification

Lessons learned after two teaching periods

Ulrich Schäfer
East Bavarian Technical University of Applied Sciences (OTH-AW)
Media Informatics and Mobile Computing
Faculty of Electrical Engineering, Media and Computer Science
Kaiser-Wilhelm-Ring 23, D-92224 Amberg, Germany
Email: u.schaefer@oth-aw.de

***Abstract*—Scrum is an agile project management method for modern software, or more generally, product development. Scrum principles are easy to understand, but its different roles, artifacts and meetings need to be trained in practice to ensure participants act appropriately in the course of a larger project. We report on a gamification (or game-based learning) approach to train Scrum to electrical engineering and computer science students using the Minecraft game. The main motivation of the idea is to concentrate on the agile framework itself instead of on software development or engineering issues. In this paper, we describe two teaching periods with approximately 110 students in total at a university of applied sciences. We modified the setup in the second run. We compare the two versions and present a critical discussion including findings and lessons learned from the teaching experiments.**

***Keywords*—gamification; game-based learning; agile software development methods; Scrum; software development; Minecraft; project management**

I. Introduction and Motivation

Scrum is an agile project management method for (software) product development. It is named ”after the scrum in rugby – a group responsible for picking up the ball and moving it forward” [1]. Central to the Scrum methodology is the team, consisting of 5–7 experts who take responsibility for iteratively pushing a defined target forward. Scrum has been developed by Ken Schwaber and Jeff Sutherland between 1995 and 2001 [1], [2], [3]. Scrum is currently probably the predominant agile software development method [4] for small teams. Scrum can be scaled to considerably more participants by *Scrum of Scrums* with representatives from each team.

The idea of the teaching project was to train Scrum in practice to ensure that students know the different roles and meetings and learn to act appropriately in the course of a larger software or product development project. Compared to a previous publication [5], we give a more elaborate description of the project and a critical discussion including findings and lessons learned from the first and second teaching period.

The audience, young, second-year bachelor students of applied computer science and electrical engineering, had different backgrounds because of their main study subjects (electrical engineering students without software engineering background). This is why we used a *gamification* [6] approach in this course. It in addition helps to abstract from software engineering and programming issues and to concentrate on the framework itself. A discussion of *gamification* vs. *game-based learning* is presented in section X.

Minecraft is a computer game originally created by Swedish programmer Markus Persson. It has been further developed and distributed by Mojang AB, a Swedish company bought by Microsoft in 2014 for 2.5 billion US$. Minecraft ”enables players to build constructions out of textured cubes in a 3D procedurally generated world” [7]. All participating students knew Minecraft before, some experts among them were able to teach the less experienced in their team. All teams (except a pioneer team in the second teaching period) shared the same Minecraft world from the beginning, and the server was configured to run in *creative mode* – as opposed to *survival mode* where participants play against other players and monsters and can “die” and lose their resources.

This paper is structured as follows. We start with an overview of the illustrations depicted in this paper in section II. In section III, we describe the course context in which the Scrum project is embedded. In section IV, we briefly introduce Scrum, and Minecraft in section V. Preparation of the Scrum project is summarized in section VI, and the main project phase, Scrum with Minecraft, in section VII. We present results of the course evaluation in section VIII and address related work in section IX. The term gamification vs. game-based learning is discussed and revised in section X. In section XI, we discuss lessons learned from the two teaching phases and compare them. Finally, we summarize in section XII and discuss pros and cons, concluding with some ideas on future directions.

II. Illustrations

To illustrate the described teaching endeavors and project results, we inserted photos, plans and screenshots into this paper. Figures 1, 2, 3, 4 are photos from the teams working on the project. Figures 5, 6, 7 and 14 depict examples of plans and maps used. Figures 8, 9, 10 are screenshots of the

978-1-5090-5467-1/17/$31.00 ©2017 IEEE 25-28 April 2017, Athens, Greece
2017 IEEE Global Engineering Education Conference (EDUCON)

Minecraft world built in the first teaching phase. Figures 11, 12, 13, 15, 16 and 17 are results of the second teaching phase.

III. Overall Course Plan

The computer science students take mandatory software engineering courses in their third and sixth semester (of seven) which the electrical engineering students do not have. However, both groups had programming experience. Most of the students had no previous experience in industrial project management and software engineering.

The Scrum training project was part of a one-semester, three hours per week course in the fourth semester called 'project organization'. The Scrum project phase did not last a full semester but rather a half. It was preceded by

- a general introduction to (classical) project management and organization,
- introduction to the precedence diagram method (PDM) and critical paths for project planning, including a blended learning phase (video) to deepen the understanding, and a practical exercise with a standard project management software application: build a project plan of your individual study (3.5 semesters done, 3.5 to come)
- three external presentations from industrial project managers (two in the second run)
- a self-study phase followed by 50–60 brief, one-slide student presentations on topics, phenomena and obstacles occurring in real project life in companies, greatly described in over 80 chapters of a translation of [8].
- an overview of approaches to agile project management in general
- a presentation on Scrum based on [9], available under a liberal creative commons license, the freely available Scrum Guide [2] for self-study and reference, and an online video showing Scrum stand up meetings and the life of a Scrum master in a real company.

The second half of the course was completely dedicated to the Scrum project. In the second run, the Scrum project started 1–2 weeks earlier, as had been suggested by the students from the first run. The main reason was to reserve more time for the Scrum project phase.

The project started with an initial poll; only two of 56 participants in the first teaching period preferred a true software development project to the gamification/game-based learning approach with Minecraft.

The *product* had been defined by the teaching professor in his role as the *product owner*: a true to scale virtual 3D model in Minecraft [7] of the (real) university campus for 1700 students with two large and four smaller buildings, including rooms, laboratories and other facilities. In the second run, the university campus model was complemented by the model of a part of the adjacent historic city center.

IV. Scrum

Scrum is an agile (software) product development and project management method. It is characterized by *roles* such as *product owner*, *Scrum master* and *team member*. A *team* consists of 5–9 members. They convene in meetings such as *estimation* and *planning meetings*, *daily Scrums*, *review meetings* and *retrospectives*. Meetings are *time-boxed*, with well-defined start and end.

Fig. 1. Standup meetings with whiteboards.

Product development and meetings occur in cycles called *sprints* until a *shippable product* is ready. The main goal of each sprint is to produce a *minimal viable product* (MVP) that is presented to the product owner or, in some selected sprints, to the client. A Scrum project typically lasts several weeks or months.

Artifacts are project documents such as *product backlogs*, *sprint backlogs*, *burndown charts* (Fig. 2) or *impediment lists* (the latter managed by the Scrum master).

Typically, and following agile principles, these artifacts fit on a whiteboard or poster. It is also possible to manage them electronically, e.g. for multi-site development. *Planning Poker* can be used to build accurate consensus estimates for sprint planning. Backlog items for a sprint can be prioritized using the *MuSCoW method*: Must have, Should have, Could have, Wont have.

A typical schedule for a Scrum project can be characterized as follows.

1. Definition of product backlog, user stories, priorities (product owner, with input from customer)
2. Scrum sprint loops:
   a. *Weekly estimation meetings*. The product owner presents product backlog items to be estimated; team members estimate items for the current sprint using

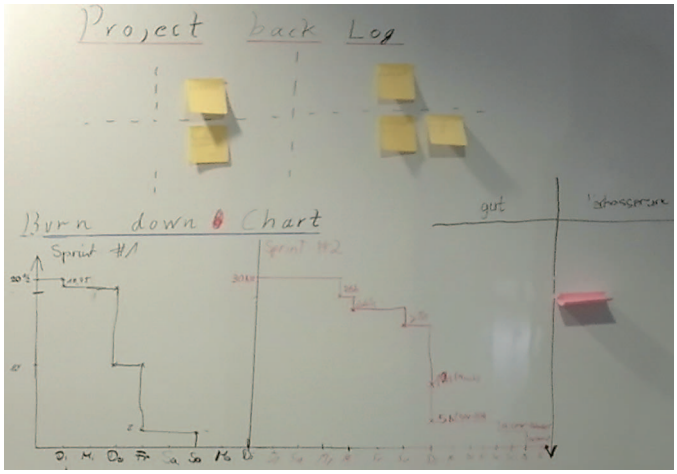

Fig. 2. Burndown chart of a team.

planning poker; large backlog items are divided, partly postponed to later sprints; the product owner clarifies unclear items until the next estimation meeting. Result is an estimated product backlog with smaller backlog items, stories to be checked, clarified.

b. *Sprint planning meetings*. Functional analysis; the team decides what to implement (promise) during the current sprint. Results are definitions: selected product backlog, requirements for each backlog item, user acceptance tests, level of done per item. The team commits itself to the defined tasks. A second planning meeting is held for design or implementation decisions on the same day after the first planning meeting.

c. *Daily Scrum*. Each team member briefly reports on what they have done, are working on, need to do and update the team backlog (task board) accordingly (Fig. 1); possible obstacles are described and discussed and put onto the impediment backlog, the latter being managed and tracked by the Scrum master.

d. *Sprint Review*. At the end of each sprint, there is a review meeting to collect feedback from end users and the product owner. New user stories may arise, impediment backlog is discussed and new items may be put on the impediment and team backlog.

e. *Sprint retrospective*. The team discusses (on a meta-level) successes and problems that occurred during development and how work could be organized in a better and more efficient way in the next sprint.

V. Minecraft

Minecraft is a 3D blocks open world game where players can build landscapes and buildings. Material such as stone, wood, ore, glass, etc. is available in cubic form and can be obtained unlimited. Material can be combined to more useful things such as tools or weapons (crafting). Blocks can be placed mostly freely in a virtual grid. The world is also built initially from this material algorithmically by the system based on random values.

Besides the crafting goal (”craft”), there is also a survival mode in which players can collect resources (”mine”), transform material blocks into tools, and fight against monsters – the survival mode was not used in our setup. The game can be played standalone on a PC or mobile device, or via network or Internet connected to a private or public server.

Fig. 3. Teams working with Minecraft.

More than 100 million Minecraft licenses have been sold since its launch in 2009 which makes it one of the best-sold video games worldwide. In our course, we used Minecraft-Edu [10], a special version of Minecraft for teaching. The pricing model is similar to the standard version with a free server that accepts up to the licensed number of clients. In contrast to the standard Minecraft clients, no individual registration is necessary on the client side. This made it easy for the students to build not only in the computer rooms at university (Fig. 3), but also remotely from home on their own machines via VPN (virtual private network).

We did not make use of the education-specific features of the MinecraftEdu server. We ran it in its bare configuration in creative mode, with animals and the 20 minutes daylight/night feature enabled. Some students complained about these settings, most were happy and built fences or rooms for the animals and used torchlights and glow stone for illumination of the buildings at night.

VI. Preparation

After a general introduction to Scrum, we provided the students with material for the Scrum meetings: colored stickers, pens, and posters as basis for backlog lists and charts. Instead of Scrum boards or whiteboards, we used blank posters lying on tables in the first teaching period which was not optimal for meetings but worked (Fig. 4). In the second run, we used eight

Fig. 4. Teaching phase 1: team meetings without whiteboards.

whiteboards (one for each team) that facilitated true stand-up meetings (Fig. 1).

Copies of the Scrum Guide [2] were given to the students so that they could consult the instructions on how to hold meetings, structure backlogs and burn-down charts. Furthermore, we suggested a freely available *Planning Poker app* to the teams as a useful tool for estimation meetings.

The *Moodle* Learning Management System [11], [12] was set up with chat, forum and repository for each team. These tools were used by the groups to synchronize efforts when working from home, to document their progress with uploaded photos of Scrum boards (posters in the first run) and burndown charts (Fig. 2).

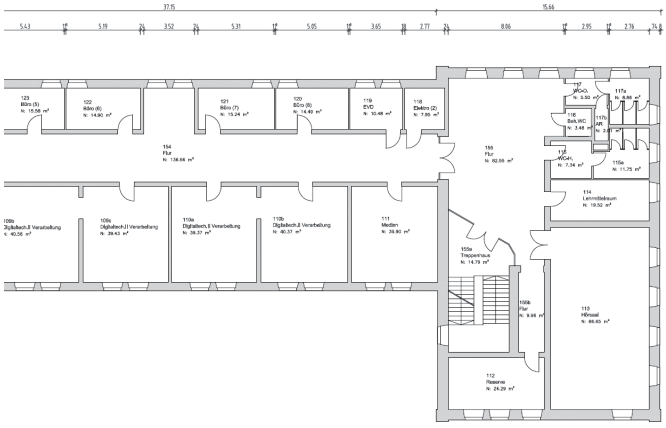

Fig. 5. Example of a floor plan (part of input specification for the teams in the first run).

When many students played at the same time, the additional per team chat through Moodle was considered an advantage over the global Minecraft chat. Finally, photos of the university campus and buildings, floor plans and side views (Figures 5, 6) were provided by the university's administration and added to the Moodle repository.

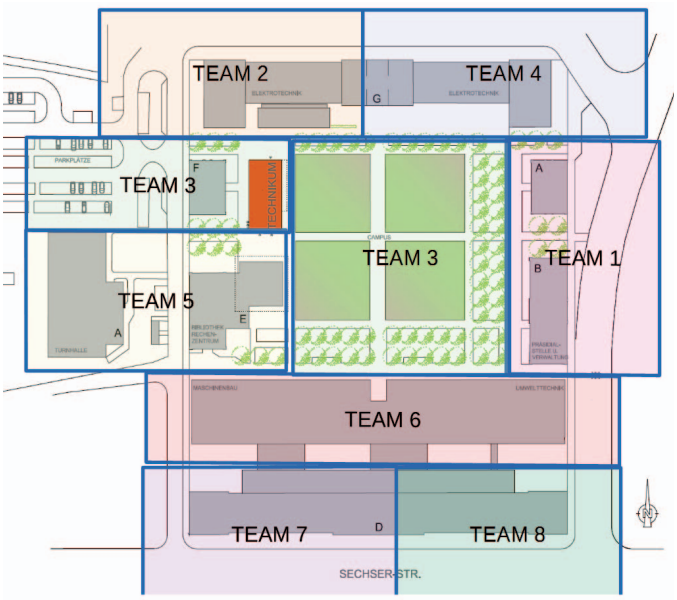

Fig. 7. Campus: team areas defined beforehand (part of input specification for the teams).

VII. Main Project Phase: Scrum with Minecraft

Initially, the teacher in his Scrum role as product owner defined the result product and divided the campus evenly into eight areas, with each of eight teams responsible for the buildings and facilities in their area (Fig. 7; in the first run). Results can be inspected in Figures 8 through 11.

Fig. 8. Mensa building interior.

For the second run (extension of the Minecraft world to the historic city center), we obtained detailed city and street maps from the municipality (Fig. 14) and divided them into per-team areas, too. Results can be inspected in Figures 15, 16, and 17.

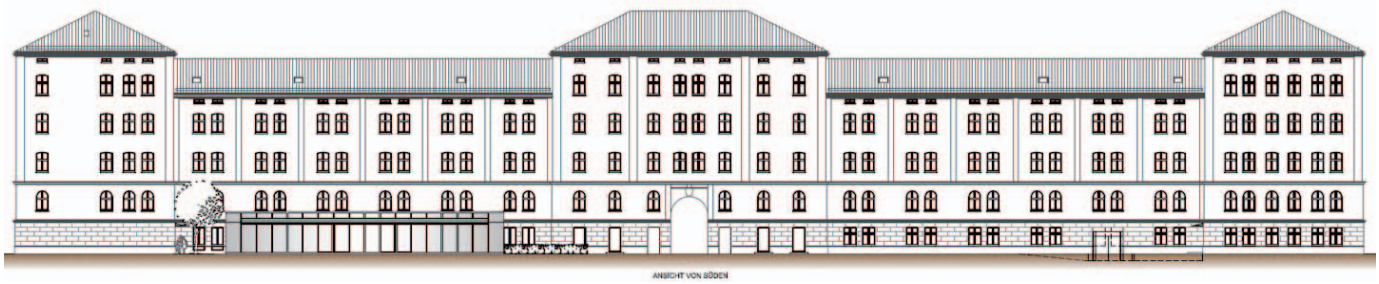

Fig. 6. Side view of a campus building, our main faculty building (part of input specification for the teams).

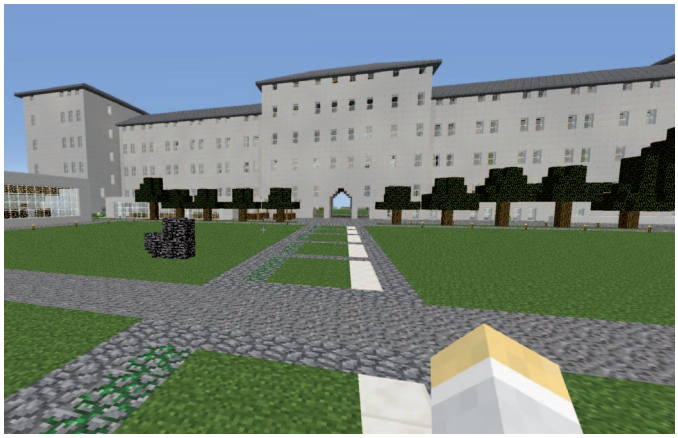

Fig. 9. Campus building (cf. Fig 6).

Fig. 10. Campus in Minecraft seen from above.

Fig. 11. Transition from campus to historic city center.

Then, each team elected their Scrum master. Planning poker was performed using a free app for smart phones in an initial estimation meeting. Negotiations among teams took place in Scrums of Scrums. Two Scrums of Scrums were held, the first one to agree on material, colors, scale and thickness of walls, roof pitch, etc. The second Scrum of Scrums was held after the first sprint to adjust landscape and building borders to exact positions in order to avoid unwanted gaps or steps at area edges. The main building phase consisted of four sprints on average, each taking one to two weeks.

Each sprint comprised planning meetings, daily Scrum, sprint review meeting and sprint retrospective. Due to the course schedule, only one real 'daily' Scrum standup meeting took place per week; some teams also met virtually using Moodle's chat facility. This was the only major deviation from the true Scrum meeting schedule.

Documentation of the sprints with posters containing product backlogs, impediment lists, burndown charts, etc. were produced by the teams during meetings. Photos of backlog lists and burndown charts (posters in the first teaching period, back side of the Scrum boards in the second teaching period) were taken at the end of each meeting using smartphones and stored in the Moodle learning management system.

A *potentially shippable product increment* had been available after each sprint: the current Minecraft world snapshot, shared by all teams. In both teaching periods, the project was finished on time, with input from all teams as planned.

VIII. Evaluation

A post-course evaluation covering the whole course, not only the Scrum project (36 written answers and orally in the final retrospective meeting), did not lead to a clear picture. Three students in the written evaluation complained about too much workload, while many others explicitly stated that

the project was well feasible and enjoyable. Two would have preferred a true software development project with Scrum over the gamification (or game-based learning) approach.

In a final *retrospective meeting* after the first run, the students explicitly mentioned that distribution of work and communication worked very well. All participants were impressed by the overall result, a realistic, complex virtual world, the on-time fulfilment of the complex project goal, and how much can be achieved with 50–60 persons, agile project management and division of labor.

IX. Related Work

Scrum education is an important and steadily growing area of software engineering research, university education and (mainly) business. Ideas and implementation range from single person training to large-scale courses. Among them are other game-based approaches such as building Lego worlds (similar to approaches of building in Minecraft), or card games. Both have the disadvantage that people who do not like the format are less willing to join. The same may hold for Minecraft, but at least the younger generation who grew up with Minecraft, seem to like it from our experience.

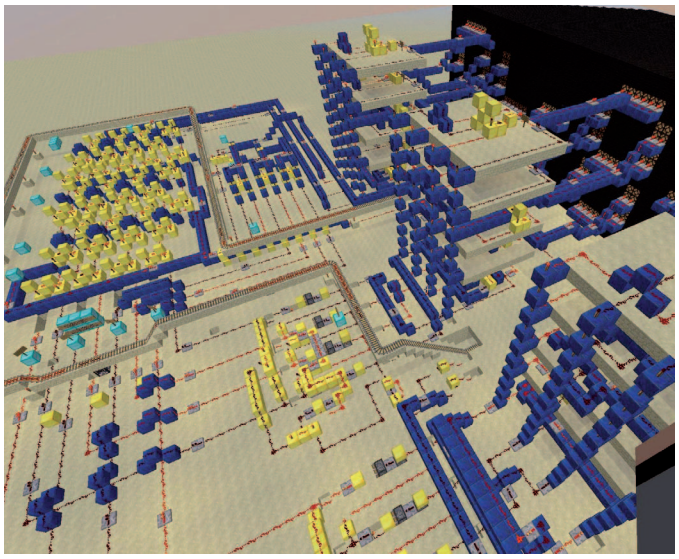

Fig. 12. A calculator built with Redstone by the Redstone pioneer team.

Only few approaches to Scrum training with Minecraft have been described so far. [13] train an unknown number of upper secondary students aged 16–17 basic Scrum knowledge, building a street with two houses. [14] presents a test course with 15 students who created a virtual company building within two days and five Scrum sprints. To our best knowledge, our approach is the first one realized as a two months long part of a full-semester university course, with a larger project and a significantly higher number of participants in eight Scrum teams working in parallel.

X. Gamification vs. Game-based learning

This paper uses the term gamification in its title. Literally, gamification means making something, in our case development with Scrum, a game. [6], [15] define gamification as "the use of game design elements in non-game contexts". One could argue that in our approach we do not just use game elements, but we build a learning process upon a complete (computer) game. Therefore, a term such as game-based learning would be more appropriate.

On the other hand, we focus on the building part (creative mode) in Minecraft only, which is not a classical game with competition and scores or bonuses as is often described as a central element of gamification. Competition may take place between the different teams if there is more than one team (as in our case eight teams), but again, here we focus on a collaborative effort of teams to fulfill a common goal, not on competition.

To complete our little tour of terminology, our specific approach of building a virtual model of a campus or city that is usable in further contexts shares some common elements with games with a purpose (GWAP; [16]) or crowdsourcing. This, however, is more a side effect and not the primary goal that is learning Scrum.

XI. Lessons Learned From the Two Teaching Phases

To support realistic stand-up meetings, we used whiteboards instead of posters lying on tables in the second phase. This helped to induce an authentic Scrum stand-up meeting feeling. The usage and benefit of virtual Scrum Poker with apps was not well understood by the students, and even less accepted in the second run. From this experience, we can conclude that is important to provide the teams with true Scrum poker cards in paper for future courses.

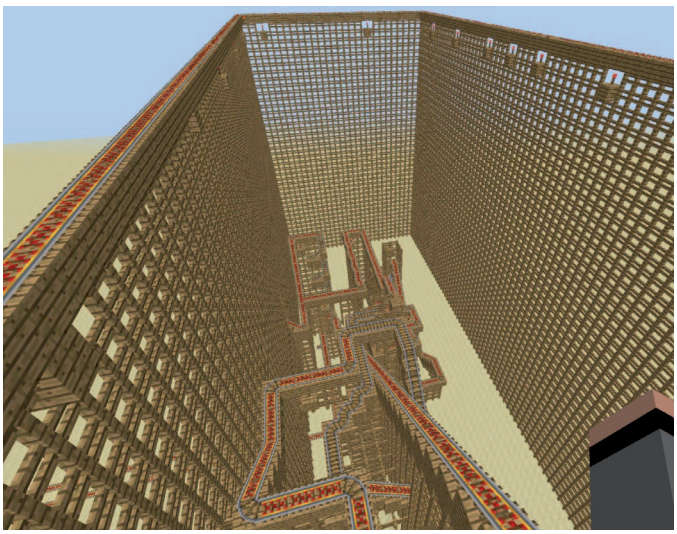

Fig. 13. Roller Coaster built with Redstone by the Redstone pioneer team.

Minecraft provides a special, fictional material called *Redstone* that can be used to build electronic circuits, circuitry gates, pistons, dispensers, and similar machines. One of the teams in the second teaching period pioneered in an endeavor that investigated an alternative approach to collaboratively constructing virtual buildings. It is especially suitable for the electrical engineering and computer science student audience, namely building a working model of a calculator (Fig. 12) as well as a roller coaster (Fig. 13), a railway with stations and some more.

We refrained from defining a global project goal based on Redstone for all course participants from the beginning because performance and stability issues could arise. However, although we ran the Redstone team on the same server as the other teams, no problems occurred. It is unclear whether the Redstone approach would scale to 50 or more participants on a standard server PC, and has to be left to future experiments.

Fig. 14. Example of a street map (part of input specification for the teams in the second run).

XII. Summary and Outlook

We summarize our findings from the two Scrum courses held up to now. From our experience, the described approach has both advantages and disadvantages. Gamification (or game-based learning) is motivating and helps bringing participants with different backgrounds together in project teams. The latter is what is needed in real development projects.

The Minecraft approach helps focusing on the project management part of the endeavor and learning the – mostly new – Scrum methodology. Third, the omission of software development and related problems allows to manage a large number of participants by a single teacher (in case this is a goal). In fact, both teaching projects with altogether 110 participants were (except for software installation and setup

Fig. 15. Historic city center: church interior.

and procurement of meeting material for the teams etc.) managed by a single professor.

There are also drawbacks. A Scrum learning project should have a real external stakeholder or customer defining a project goal externally, e.g. a representative of the local tourist office, a company, or an administration. This was impossible to integrate with the short preparation phase in our setup and it is also questionable whether this can become realistic in conjunction with a gamification (or game-based learning) approach at all.

Furthermore, for time reasons we did not submit change requests to the teams. This would be an important obstacle for the teams to deal with in a more realisitic training project.

From the viewpoint of software engineering, the gamification idea as presented almost completely lacks an aspect that is very important in combination with agile software development: test-driven development. We currently see no realistic setup that combines test-driven development with Minecraft except some artificial definition of testing embedded in the game.

An alternative, 'hybrid' approach could consist in a combination of Minecraft building tasks complemented with software-based building targets based on a Minecraft API (application programming interface): there exist extensions ('mods') for Minecraft that provide a crafting API for programming languages such as Python, the most prominent being based on a light version of Minecraft for Raspberry Pi [17]. The latter, however, is very limited compared to the standard or educational version of Minecraft and does not support networked, server-based scenarios. There is a backport for the standard Minecraft version called RaspberryJuice [18].

While the main goal of the project was to train Scrum in practice, the resulting Minecraft model of the university campus from the first run can also be re-used in further teaching

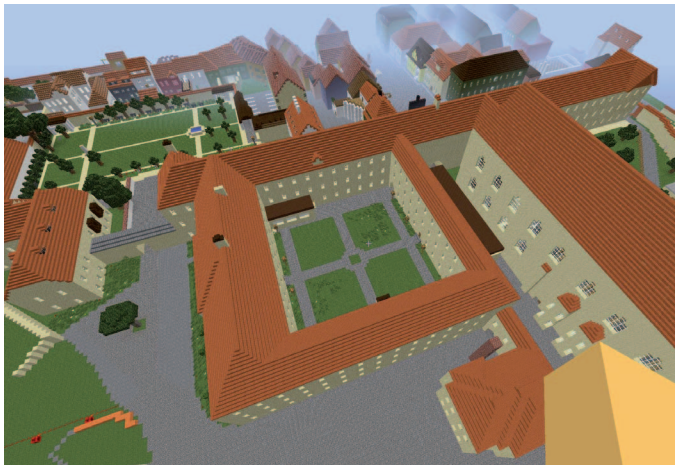

Fig. 16. Historic buildings next to the campus, seen from above.

Fig. 17. Historic city center seen from above.

projects in the future, e.g. for providing virtual classrooms and in combination with other applications, software projects and computer games.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their helpful comments, Harald Hofberger, the dean of our faculty, for financial support, Margit Jäger for contributing floor plans and building views and photos, my collegues Dominikus Heckmann for providing teaching material on classical project management, and Martin Frey for moral support and our lab engineers Heiko Specht, Florian Haupt and Benjamin Michallok for help with the Minecraft server and client installations. Last but not least I thank my university for awarding me the teaching prize 2015 for "Scrum with Minecraft".

REFERENCES

[1] K. Schwaber, "Scrum development process, position paper," in *Business object design and implementation: OOPSLA '95 Workshop Proceedings*, Univ. of Michigan, 1995.

[2] K. Schwaber and J. Sutherland, "The Scrum guide," 2001, http://scrumguides.org.

[3] L. Rising and N. S. Janoff, "The Scrum software development process for small teams," *IEEE Software*, vol. 17, no. 4, pp. 26–32, July 2000.

[4] K. Beck, "Principles behind the agile manifesto," 2010, http://www.agilemanifesto.org/principles.html.

[5] U. Schäfer, "Teaching Scrum with Minecraft," in *Proceedings of the European Conference on Software Engineering Education (ECSEE) 2016*, Seeon, Germany, June 2016, pp. 191–195.

[6] S. Deterding, R. Khaled, L. Nacke, and D. Dixon, "Gamification: toward a definition," in *CHI 2011*, Vancouver, BC, Canada, May 2011.

[7] "Minecraft homepage," February 2017, http://minecraft.net.

[8] T. DeMarco, P. Hruschka, T. Lister, S. McMenamin, J. Robertson, and S. Robertson, *Project behaviors: from adrenalin junkies to template zombies*. Dorset House Publishing, 2007.

[9] M. Cohn, "An introduction to Scrum," 2014, English redistributable presentation slides, http://www.mountaingoatsoftware.com.

[10] "MinecraftEdu homepage," February 2017, https://minecraftedu.com.

[11] "Moodle learning platform homepage," February 2017, http://www.moodle.org.

[12] M. Dougiamas and P. C. Taylor, "Improving the effectiveness of tools for internet-based education," in *Teaching and Learning Forum*, 2000.

[13] T. Härnvi, "Cooperation in minecraft," in *GAMEPADDLE – Video Games. Education. Empowerment.*, M. Anderle and S. Ring, Eds. Milan, Italy: Ledizioni Srl, 2012, pp. 82–91, http://www.jff.de/gamepaddle/wordpress/publication/.

[14] B. Stieler, "Scrum in zwei Tagen praktisch vermitteln – geht das überhaupt?" 2013, https://denkanstoss.netpioneer.de/projekt-management/si2d/.

[15] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining gamification," in *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*. ACM, 2011, pp. 9–15.

[16] L. von Ahn and L. Dabbish, "Designing games with a purpose," vol. 51, no. 8, pp. 58–67, August 2008.

[17] "Minecraft Pi," February 2017, https://www.raspberrypi.org/learning/getting-started-with-minecraft-pi/.

[18] "RaspberryJuice homepage," February 2017, https://dev.bukkit.org/projects/raspberryjuice.