Part V

Student Engagement

Chapter 11

A Workshop for Agile Modelling

Students have various difficulties with software modelling, the software development process and with positioning modelling as a means to support their software development. The Agile methodology Scrum has gained popularity in industry and also amongst students. Unfortunately agile projects often lack adequate documentation. Modelling and the agile process could complement each other. The combination of modelling and agile development is not often used in education. Based on our positive experience with the interactive LEGO4SCRUM workshop we use in our programs, we propose an approach based on this workshop that integrates UML modelling into the Scrum process. The workshop lets students experience a whole development cycle from a modelling perspective. Besides this new approach we also categorised comments students wrote down based on their discussions with their peers. We evaluated the workshop with a questionnaire. The students react positive on the approach and indicate they have gained new insights. This chapter explains the workshop set-up, presents its evaluation and discusses the results.

This chapter is based on the following publication: Dave R. Stikkolorum and Michel R.V. Chaudron. A workshop for integrating UML modelling and agile development in the classroom. In *Proceedings of the Computer Science Education Research Conference 2016*, pages 4–11. ACM, 2016

11.1 Introduction

As educators we often see students have various difficulties with software modelling and organising a software development process. Besides learning a syntax of a language such as UML, students have to deal with translating problems into models (analysis) and suggest a solution (design). The abstraction skills that have to be trained for this kind of task often are isolated from another student challenge: the software development process cycle.

Although methodologies such as RUP (Rational Unified Process) and MDD (Model Driven Development) are intertwined with modelling activities, students have difficulties in giving the models a role in the process and treat them isolated, as necessary evil, not as useful tool that supports their development process. Mussbacher et al. [94] show in their work that modelling is not that widely spread (yet) as we (educators and scientists) had hoped for. In the past decade agile methodologies, in particular Scrum, have won popularity in industry and in the classroom. Scrum is designed to do 'just enough' and not to overdo on documentation and does not emphasise modelling. Students seem to appreciate Scrum but, as also found by Stettina et al. [129] in an industry survey, the documentation they produce is inadequate and often lacks models of the design.

Only providing students a methodology that they like cannot do 'the trick'. Students still have to cope with the difficulties of the analysis and design skills and they need to be trained. Educators have to better understand what kind of difficulties students have and how they can guide them to improve their modelling skills.

In our own software engineering programs we use the UML standard for modelling domain and solution with a focus on class diagrams and sequence diagrams. Although the syntax is relatively easy to learn, we notice students seem to have difficulties in placing the use of these diagrams in the process of software development: they do not see the relationship between the diagrams and cannot see the supporting role it has during development.

Consider the following example: in an initial design some related classes play a role in a certain scenario. A very useful technique is to use sequence diagrams to update the design through simulation of certain scenarios of use cases. Our experience is that students often only see the separate steps and do not update their models.

Another problem we noticed students struggle with is how to move from analysis to design. They hesitate what should be the first step.

In industry most software development is done in project settings and students have to learn how to efficiently and effectively work together. After having experience using RUP in our programs we now use Scrum as a software development methodology. We believe the best way for learning Scrum is to let students apply it in a project course. Because we want to avoid to let them just 'use Scrum', we introduce them to the subject of Scrum in a series of lectures and then let them apply this in a project. Often this did not turn out to work that great. We wanted the students to have more experience in Scrum before they start, therefore we adopted a well known workshop that introduces Scrum in a few interactive hours to the students: 'LEGO4SCRUM' [76](mentioned in Section 11.2). The advantage in using the workshop is that students see a whole development cycle in a short time frame. We believe, because of the short time frame, they better relate the development steps.

In previous work [137, 136] we tried to find what typical difficulties students have during a software design task in order to collect information and what kind of feedback we could provide students to improve their modelling skills. We found out that students have difficulties in summarising and explaining their problems clearly.

Based on our positive experience with the LEGO4SCRUM workshop, we propose in this chapter an interactive 2 lecture-hours workshop. In the workshop students are activated to use various types of modelling and have to come, with the help of the discussion in their group, to a detailed design for sprint 1 of the Scrum process (explained in Section 11.3).

In this chapter we answer the following research questions:

- RQ1: Can we integrate (UML) modelling and agile development into an educational workshop?
  and the following sub-questions of RQ1:
  - RQ1.1: Do students learn about the role the (UML) diagrams have in a development process?
  - RQ1.2: Does the workshop help students to improve their modelling skills?
- RQ2: Which insights for modelling education do we gain from the students' group discussions during the workshop?

With this research we aim to contribute: i) a workshop approach for combining modelling and agile development in the classroom and ii) collecting insights into the problems students face during software modelling activities in order to develop future educational programs.

When we talk about modelling in this chapter we mean expressing the problem domain or designing the software solution with the help of UML diagrams.

This chapter is organised as follows: in Section 11.2 we discuss related work. In Section 11.3 we explore our approach and show the set-up of the workshop in more detail. We show the results in Section 11.4 and discuss them in Section 11.5. Section 11.6 discusses threats to validity. The last Section (11.7) is used for conclusions and future work.

11.2 Related Work

To put our work into context, we discuss the related work in this section. First we address work that discusses the problem of understanding the different modelling steps and models. Secondly we address research that combines (elements of) agile with model based methodologies. Lastly, we show interactive approaches of software education workshops and games.

Kaindl [62] discusses the difficulties in transitioning from analysis to design by arguing that the analysis model represents other things than the design model. He explains that analysis objects reflect the problem domain and therefore cannot be the same thing as a design object that reflects a part of the solution. Shin explains that this transition difficulty is caused by semantic distance [122]. He argues that design objects are closer to code than analysis objects are to the problem domain. This creates a semantic distance between the analysis and design.

Since more than a decade researchers suggest to combine modelling and agile from different point of views. Scott Ambler discusses the use of models in an agile way [5] in order to avoid 'over documenting' and proposes to do 'good enough'. Rumpe [116] suggests to use an executable version of UML and compares the use of that with the agile focus on creating rapidly executable models. He applies agile modelling with UML in his textbook [117].

Zhang et al. [162] describe a combination of MDD and agile. They use pair modelling, continuous modelling and UML code generation.

Schroeder et al. [119] use Scrum and a network based card game case for their educational activities. They encourage students to use UML to support their steps and documentation(not per se model based). They mention LEGO4SCRUM [76] as a good team building exercise.

Krivitsky [71] describes the LEGO4SCRUM workshop, a workshop in which students experience a Scrum development cycle including multiple sprints within a couple of hours. The workshop is divided into parts that represent typical Scrum phases. A timer constrains the students' actitivities. Paasivaara et al. present a simulation game for teaching Scrum to students with LEGO blocks [98]. The game was first developed as a training tool for a security company that adopted agile development in their processes.

Other game based approaches are described by Fernandes et al. They use a card game [36] for university level students to learn Scrum. Wangenheim et al. show a paper and pencil game to teach their students the application of Scrum [158].

Our workshop combines the idea of using models within an agile process with the time-driven structure of the LEGO4SCRUM workshop. We cover a Scrum development cycle and let students experience to move from analysis to design. We do not use the LEGO blocks.

To our knowledge, in practice agile and modelling is not combined often in education.

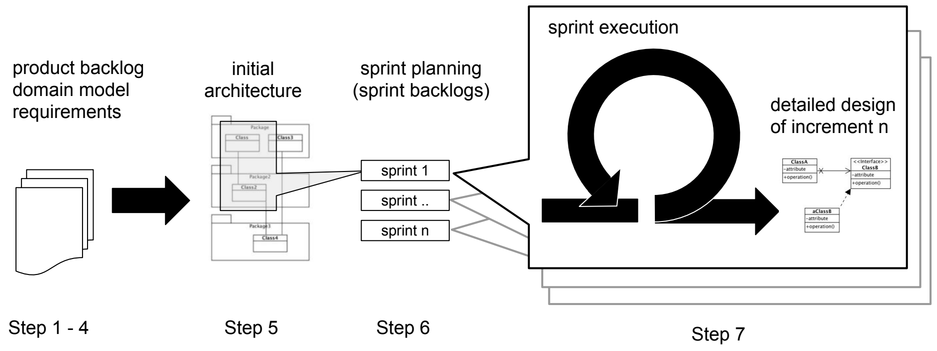

Figure 11.1: *workshop setup*

11.3 Approach

In this section we describe the approach of the workshop. We describe the participants, the case that was used during the workshop, show the tools that were used and explore the workshop structure. We conclude with the questionnaire that was conducted at the end of the workshop.

11.3.1 Participants

The participants of the workshop were first year Bsc students that followed the second semester of a software engineering program. In advance the students followed a Java programming course and followed a project course in which they used Scrum. The students had a basic understanding of class diagrams and sequence diagrams from the

UML. The workshop took place during a regular lecture of the 'analysis and design' course. Of the 88 registered students 50-60 participated in the lecture. In parallel students worked on a project. We used the project groups in the workshop. Students of 13 groups participated. The groups in the workshop consisted of 3-5 students.

11.3.2 Case

As a runner I want to measure my time and distance so that I can improve my running next time I run

As a runner I want to measure my heart rate so that I can monitor my physical condition

As a runner I want to measure my blood pressure so that I can monitor my physical condition

As a runner I want to send my statistics to the doctor so that we can discuss them

As a doctor I want to read clients' statistics from the clients database so that I can use this information in my advice

Figure 11.2: *Initial User Stories*

The case that was provided was about a 'running app' for a mobile phone that is capable of handling different sensors, such as a watch (measures heart rate) and a blood pressure belt.

11.3.3 Materials and Tools

The students needed a pen and paper to write down proposals they had to hand-in during the workshop. As a modelling tool most of the students used Visual Paradigm [157]. Pen and paper was also allowed. The lecturer needed a beamer and laptop to present slides and to present a timer that was visible during the workshop.

As a runner I want that the music adjusts to the heart rate, so that I can enjoy my running even more

As a runner I want a map that shows my position so that I know where I am

Figure 11.3: *Adding User Stories*

11.3.4 Workshop Procedure

The main idea of the workshop, like the LEGO4SCRUM version, is to simulate a whole Scrum development life-cycle with the emphasis on modelling. The pushing factor is time. The time constraints per phase force the students to discuss, make decisions and model. The product a sprint delivers in our workshop is not a LEGO building or for example (compiled) code. The students produce partial designs that represent an increment of the software solution. The diagrams students use from the UML are use case diagrams, sequence diagrams and class diagrams. The workshop procedure described below follows a typical Scrum work flow. We integrated the modelling activities with the 'standard' Scrum phases. We see this as an extension, because Scrum does not prescribe modelling activities. We believe the iterative character of Scrum, especially having multiple sprints, enables developers to elaborate on their (initial) design.

Figure 11.1 shows the workshop process. The workshop procedure was as follows:

1. **Initial set-up**: The workshop starts by presenting a first domain model that was based on initial user stories (see Figure 11.2). Also the user stories are shown on the screen.

2. **Adding new user stories** (5 minutes): The first activity is to come up with suitable use stories to add. The students have to think as the client and deliver two properly described user stories in time. After the deadline the lecturer discusses the proposed user stories and explains why some of them are not properly defined. Two additional user stories are chosen. This is done by acclamation (for the extra user stories see Figure 11.3).

3. **Update the domain model** (5 + 5 minutes): the students have to update the domain model based on the new user stories. The domain models are peer reviewed. When the deadline has passed the teams have to send two students to criticize the models of two other teams. They grade the models on a 1-5 Likert scale and hand-in the grades.

4. **Adding new requirements** (5 minutes): the next step is to add two new requirements. Five requirements are already shown on the screen (see Figure 11.4 for the initial requirements). The students may use internet and other resources to come up with suitable requirements for the 'running app' case. They hand-in their proposals on paper. Two additional requirements are chosen. This is done by acclamation again.

Requirements:

- The measurement is done by bluetooth devices
- The application should run on Mac OSX+, Windows 8+ and Ubuntu 14+
- There should be a graphical interface and sound
- The distance is displayed in km or miles
  - there is one digit used after the '.' e.g. 10.3 km

Figure 11.4: *Initial requirements*

Extra Requirements:

- The application should adjust the type of music to the pace of the runner.
- The application should use the Google Maps API.

Figure 11.5: *Extra requirements*

5. **Propose the initial architecture** (10 minutes): to be able to plan your sprints most Scrum implementations mention an initial architecture or initial high-level design. This design shows developers an overview of the work that has to be done and therefore supports planning. In our case the students are given a 4 layered architecture framework (UI layer, Application Layer, Domain Layer, Library Layer). The domain model together with requirements that were introduced in the previous steps push the design of the students in a certain direction. The students have to use a UML class diagram to model the architecture. The initial designs are peer reviewed again by different students from other groups.

6. **Sprint planning** (5 minutes): the students choose what particular part of their initial design can fulfil one or more user stories and defines that as a sprint. Sprints take 15 minutes. Students have to choose the part from which they estimate it is feasible to design (refine) it in 15 minutes.

7. **Sprint 1 .. n** (15 minutes per sprint): An agile project consists of multiple sprints that deliver increments of the product. In the workshop the increment of the product is represented as a detailed design part. The iterative character of Scrum, having sprints, is the typical link with what we expect from using modelling languages, elaborating and refining the (design) models. In the sprint the students will use two different diagram techniques from the UML: class diagrams and sequence diagrams. The students will update (refine) the design based on the sequence they explore with the sequence diagram, or test their updated classes and relations by simulating possible scenarios that link to user stories. Again the results are being reviewed by peer reviewers.

8. **Workshop feedback** (5 minutes, not shown in Figure 11.1): the lecturer mentions what stood out while he/she was observing and ends the workshop.

11.3.5 Questionnaire

After the workshop the students were asked to fill out an online questionnaire about their experience with the workshop approach. The questionnaire consisted of 8 questions that were answered on a likert scale (-2,1,0,1,2) and 4 free text questions. The questions targeted the students' experience with this different way of learning and how well they think they have learned from the workshop.

11.4 Results

In this section we discuss the questionnaire that was used to collect the data for evaluating the workshop. First we show the results of the 8 Likert scale questions. Subsequently we show the results of the 4 free text questions. 30 students filled in the questionnaire.

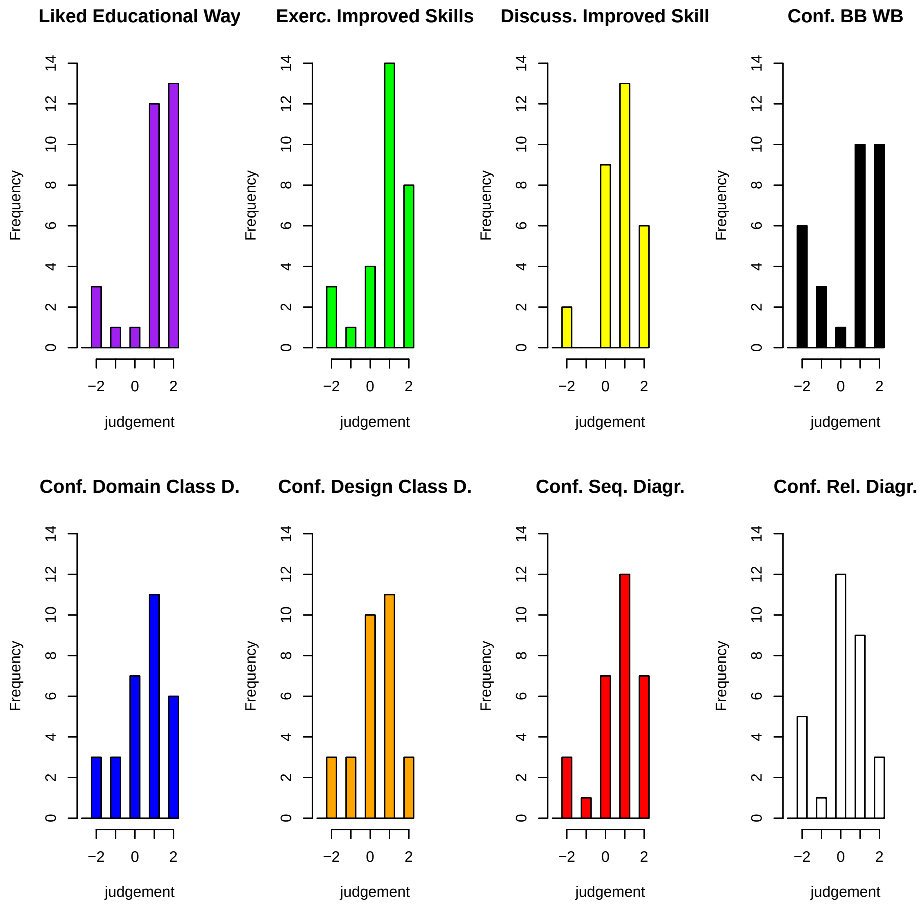

Figure 11.6: *Student Survey Responses (N=30)*

11.4.1 Scaled Questions

Figure 11.6 shows the histograms of the scaled questions of the questionnaire. To explain to which question an abbreviation in the histogram relates we list the questions with the abbreviation between the parentheses. We added statistical information: the mean and standard deviation. For every question we show a percentage. It is the percentage of the total respondents that judge neutral (0) or positive (+1,+2) which is notated as follows: j⩾0: 50%

- Did you like the way we used the exercise as a way of learning? (Liked Educational Way: M=1.03, SD=1.25) j⩾0: 87%
- Did the exercise help you to improve your modelling skills? (Exerc. Improved Skills: M=0.77, SD=1.19) j⩾0: 87%
- Did the discussion in the group help you to improve your modelling skills? (Discuss. Improved Skills: M=0.70, SD=1.02) j⩾0: 93%
- How confident are you that you understand the difference between black box perspective and white box perspective? (Conf. BB WB: M=0.5, SD=1.55) j⩾0: 70%
- How confident are you that you understand what a domain class diagram is? (Conf. Domain Class D.: M=0.47, SD=1.22) j⩾0: 80%
- How confident are you that you understand what a design class diagram is?(Conf Design Class D.: M=0.27, SD=1.11) j⩾0: 80%
- How confident are you that you understand what a sequence diagram is? (Conf. Seq. Diagr.: M=0.63, SD=1.19) j⩾0: 87%
- How confident are you that you understand the relation between the different diagrams? (Conf. Rel. Diagr. M=0.13, SD=1.20) j⩾0: 80%

The results show a positive judgement on all the scale questions. On all questions 70% - 93% of the subjects answer neutral or positive (+1 or +2). If one only considers the very positive judgements (+2) there is more variation.

Considering the mean (M=1.03) and the high amount of positive judgements (25 out of 30) of ‘Liked Educational Way’, students judge most positive about the educational approach, followed by the positive judgements about the improvement of their modelling skills.

What typical difficulties/challenges were discussed in your group while your were doing the modelling tasks? *"How detailed or simplified data modelling should be under different context"* *"Which class belongs in which package. What functions and attributes to give the classes."*
At the moment what is the most difficult topic in modelling for you? *" ... but I will continue my exercise to understand more about the relationships between the different diagrams"* *"Constraining and limiting the modelling. You can include everything if you want. The most difficult is knowing where to draw the line. What's included in this system? What's included in this label etc."*
What did you learn from the exercise that you did not learn from the lectures? *"I learned more about refining the domain model as well as grasping a better understanding of the initial design concept."* *"gained a better understanding of the whole modelling process since we went from the start to almost end."*
Remarks about the lecture *"Really good exercise lecture and I wish we could have more lectures like that. I believe the presence of the TAs could have helped a bit more as they could be going around and help with questions."* *"Interesting. We should have more exercises like this."* *"A lot of fun. I love the competition part of it!!! But I think that whole groups are a bit too big for such a short time exercise. Its hard for everyone to contribute equally."*

Table 11.1: *General comments on the exercise*

11.4.2 Free Text Questions

Table 11.1 shows some of the comments the students had on the exercise. Here we list the questions we asked:

- Please write down (short sentences) what typical difficulties/challenges were discussed in your group while your were doing the modelling tasks.
- At the moment what is the most difficult topic in modelling for you?
- What did you learn from the exercise that you did not learn from the lectures?
- Any remarks about the lecture? Please write them down.

The answers on the free text questions were analysed and categorised.

Group discussion and difficulties The students mentioned the following matters as part of the group discussion or as a subject that they find difficult:

Translation of a case - students commented on the difficulties they have 'translating' a case-text into a conceptual model. The main step here is an analysis step: the discovering of key concepts in the domain. Students struggle when they have to decide whether something is a concept or not. They explain they have difficulties in linking new concepts to the concepts they identified earlier in the analysis of the case: do responsibilities belong together in one concept or do they need to be split across separate ones? Is a responsibility big enough to be its own class or can it be an operation of some class?

Organising - another matter that was pointed out is having difficulties in organising a design into layers. Students indicate they discussed about how to assign attributes and operations to the classes and how classes should be assigned to packages.

Detail - a point of discussion that emerges is how much detail should be used in a diagram? One student commented about where to draw the line: 'You can include everything if you want'. The following questions were asked: should attributes and operations be added? If so, how many? Should parameters be included in operations? Do we have to put multiplicities and labels on associations?

Layout - students have the need to discuss the layout of the diagram. They mention they sometimes discussed more about the aesthetics than the logic.

Different diagram types - students discuss about when to use which diagram type. Also, they are puzzled about the relations between the diagrams. We believe this relates to the consistency rules that apply to the diagrams, such as: if a method appears in a sequence diagram, what does this mean for the class diagram of the design of the system?

Learned from the exercise Students indicate they have learned several matters from the exercise. They relate to the topics found in Section *Group discussion and difficulties*. Students gained a better understanding in how to build diagrams (detail, organising, translation of a case). They comment they learned about refining the domain model by analysing the case, identifying key concepts and adding detail (detail, translation of a case). After attending the workshop they better understand the initial design concept (diagram types). Students mention 'how to easily update from the diagram' (detail) and learned about layering (layout, organising).

Related to the development process students comment they have learned about the different steps and roles of the diagrams in software modelling (diagram types) by seeing the whole development process from the beginning until the end.

Other remarks One student wrote that he/she learned 'about the importance in consensus for naming'. Another mentioned reflected 'our group thought about more domain aspects than other groups'. Some students suggested to work in smaller groups in order to let everybody participate equally.

11.5 Discussion

In this section we discuss the research questions that were presented in Section 11.1. Based on the results presented in Section 11.4 we aim to answer them.

11.5.1 RQ1: Can we integrate (UML)modelling and agile development into an educational workshop?

The workshop based approach enabled us to combine UML modelling and agile development. In a short time frame students were able to experience UML modelling as support during a Scrum based development process. We believe, by compressing real-world time into minutes, students benefit of seeing the whole development cycle in one lecture. They are able to quickly experience difficulties that arise because of decisions made just a couple of minutes earlier. Besides the usefulness students also mention it was 'fun' to do. Also the statistics (fig 11.6: Liked Educational Way) show this.

RQ1.1: Do students learn about the role the (UML) diagrams have in a development process? In the free text questions the students comment, because of the exercise they were able to see the 'whole picture' of the development process. They explained that they were able to see the relations between the different steps and the different models that they can use. The statistics show 40% of the students is neutral and another 40% is positive about seeing the relationship between the different UML diagrams. Only 20% has a negative feeling about this. Although, considering the mean of 'Conf. Rel. Diagr.' (0.13), students seem somewhat cautious to judge their understanding very positive, we believe the workshop helped for the larger part of the students.

RQ1.2: Does the workshop help students to improve their modelling skills? Students believe they improved their skills by doing the exercise. 73% has a positive judgement about this. 63% thinks the discussion in their groups helped improving their modelling skills. We expect that the improvement students judge about are: understanding the relationship between the different diagrams and development phases and refinement steps (mentioned in Table 11.1). The workshop actively emphasises these topics through practice. Of course additional long term practice is required for significant improvement in these skills and the variety of additional modelling skills.

11.5.2 RQ2: Which insights for modelling education do we gain from the students' group discussions during the workshop?

The statistics suggest us to use this kind of exercise more often. The 'Exerc. Improved skills' and 'Discuss Improved Skills' data show a positive judgement from the students. Also free text comments saying 'Really good exercise' or 'it was very interesting' indicate a, for students, motivating educational approach.

The workshop enables the students to discuss the difficulties they face during the modelling tasks. We believe that modelling in groups enabled the discussion between the students and causes them to argue about the choices they make. This is comparable with the pair-programming technique from agile development, where coders review each other during coding. The students were able to show us there is a variety of challenges they have to face. From the response on the free text questions, we were able to identify 5 topics students find difficult as shown in Section *Group discussion and difficulties*.

Maybe dividing the group into pairs, as suggested by some of the students could achieve even better results.

11.6 Limitations and Threats to Validity

At the moment the workshop does not reflect on the quality of the student software designs from the lecturer's perspective. It is limited by the judgements of the students' peers. This was done because of the focus on the role of the diagrams in the agile process rather than to have an implementable design.

Out of the 50-60 students, 30 participated in the questionnaire. We are aware that this could introduce a bias in our results.

We are also aware that we have analysed one particular case and have to be careful to generalise our findings.

11.7 Conclusion and Future Work

In this chapter we proposed a workshop that integrates UML modelling into agile development in order to: i) have an integrated software development approach that helps student to understand the role of (UML) modelling in a software development cycle and ii) collect information about students' difficulties that arise during discussing the modelling task in a group. The results indicate that the students in general have a positive judgement about the workshop. The statistics also show that most students judge they have improved their modelling skills and that the workshop gave insight into the relationship between modelling and the software development process. We were able to categorise their comments. In future work we will add the results of the comments to our already collected student feedback on similar analysis and design assignments. We will use the data in order to expand our own educational modelling tools and educational approaches with proper feedback to guide students during their modelling tasks. As future work we like to explore a variation of this workshop in which we will include actual coding and exercise maintenance of the code.