Agile Principles Applied in Learning Contexts

Virginia Niculescu
Dan Suciu
Darius Bufnea
virginia.niculescu@ubbcluj.ro
dan.suciu@ubbcluj.ro
darius.bufnea@ubbcluj.ro
Faculty of Mathematics and Computer Science, Babeş-Bolyai University
Cluj-Napoca, Romania

ABSTRACT

Agile methodologies have been recently proposed to be used in education. In this paper, we propose a rephrasing of the 12 Agile principles for learning context, and we provide concrete application-oriented interpretations for them. Additionally, a practical agile learning methodology is proposed to offer a framework where these principles could be applied. The principles together with the proposed methodology were applied to a concrete use case which is described and the resulting impact is analysed.

CCS CONCEPTS

• Applied computing → Collaborative learning; **• Computing methodologies → Parallel computing methodologies**; **• Software and its engineering → Agile software development**.

KEYWORDS

agile development, agile principles, agile methodologies, learning outcomes, teaching methods, learning frameworks, use-cases

1 INTRODUCTION

One of the most difficult goals to achieve in delivering academic courses is to attract and preserve students' attention and commitment. Furthermore, another important challenge is to ensure the students accomplish the needed learning outcomes. In order to overcome these challenges, new teaching methods were developed, and part of them have been inspired by predictive (waterfall) or adaptive project management methodologies. This seems to be an obvious approach since the teaching/learning process has many common elements with the way a project is developed. A project is a temporary, unique and progressive endeavour made to produce some kind of tangible or intangible results [10], and this is also the way we usually learn. Both projects and teaching activities involve multiple parties with different objectives (sometimes conflicting), a very tight schedule to get things done, a fixed deadline, limited resources and a lot of expected/unexpected changes along the way. Moreover, both teaching and project management processes require planning, monitoring and control with continuous verification and validation of deliverables and feedback from all involved individuals. We consider that just translating a particular methodology to an educational approach is not so effective. It is more valuable to understand and embrace the values and principles that stay behind all practices and tools a methodology is based on. Understanding the principles will give us a good foundation to develop specific practices and tools for effective education.

In the last 20 years, there has been a gradual transition to using more flexible and adaptive project management frameworks, called agile methodologies, especially in software development projects. In 2001, a group of 17 software consultants published "Manifesto for Agile Software Development"[1], which explicitly stated the four key values and 12 operating principles that lie beneath the Agile mindset.

These values and principles are more or less followed by all Agile methodologies, as Scrum [19] (which is the most popular Agile methodology, more than 75% of Agile project teams implementing a derivation of Scrum[7]), Extreme Programming[4], Feature Driven Development[17], Agile Unified Process[5], Disciplined Agile[3][2] etc.

Since the Agile values and principles were not changed in the last 20 years, all mentioned methodologies suffered a lot of transformations to better fit the project development teams' needs.

The first objective of this study was to reanalyse all the 12 principles of Agile Manifesto through the education context perspective, and to give application oriented interpretations for effective teaching and learning purposes. The second objective was to apply these agile principles for finding concrete methods to be applied in concrete educational use-cases based on Agile learning.
Corresponding to these objectives the contribution of this study is two-fold:
- an application oriented interpretation of the 12 Agile principles, and a practical Agile learning methodology;
- the description and analysis of a concrete use-case where Agile methodology was applied.

The paper is structured as it follows. The next section presents a brief survey of the previous work related to Agile learning. Section 3 analyses the 12 Agile principles and proposes the methodology, and in section 4 we present a concrete application of Agile learning. The paper ends with the conclusions and the proposed further work.

2 RELATED WORK

Agile learning being derived from the Agile software development as a methodology is relatively new, but recently it received important attention, with different corresponding research reports being published.

A series of articles on this subject were collected in a book: "Agile and Lean Concepts for Teaching and Learning: Bringing

Methodologies from Industry to the Classroom" [18]. These articles explore the application of agile and lean techniques to various aspects of education. Topics such as lean thinking in educational workflows, and using team-based approaches to student-centred activities based on Agile principles and processes are discussed.

One of these papers, "Agile Methodologies in Education: A Review: Bringing Methodologies from Industry to the Classroom" [20], reviews some of the Agile methodologies that have inspired educational approaches and provides a description of the features retained in the educational context, reporting several experiences, too.

A first review of such methodologies was presented before by Stewart et al. in [21]. Their aim was to show how Agile methods could be applied to education. In this paper, we also found the first mapping between the values and principles of the Agile Manifesto to specific educational methods and activities. We were inspired by them, but we tried to give a more concrete interpretation that could be effectively applied in practice.

Another review of the Agile learning methodologies is given in [6]. In general, teaching and learning at university has migrated from traditional learning to an active learning methodology where students are expected to learn by doing rather than listening passively to lectures. The study emphasized the direct link between learning by doing and agile learning, since they are proved to be compatible and supportive towards active academic learning.

Particular Agile methodologies were adapted to be used in the learning processes. An example is 'eduScrum' [22], which enables students to become more independent and self-directed in their learning.

The fact that students today are part of the gamer generation inspired the idea that games could also be used as a support for training and competence development in an Agile learning approach [16]. Agile games were also used for learning fundamental agile and lean concepts [9].

Paper [15] explores the possibility to combine cyclic learning with agile learning methodologies and emphasises similarities as building knowledge using several iterations and adding new functionalities/knowledge to each of those. Cyclic and Agile learning do not exclude each other, rather they are complementary, cycling learning being enhanced through Agile methodologies.

Longmuß et al. performed in [12] a complex research project aimed at bridging the gap between industry and university Agile learning. As a result, they emphasised that it is essential to discuss prerequisites, methods, and resources as well as the intended outcomes in detail, before engaging in the process of an agile learning project.

3 AGILE LEARNING PRINCIPLES AND PRACTICES

We first analyze the 12 Agile principles and give interpretations for their practical application in learning contexts, and then we propose a methodology to be used as a framework for concrete application of Agile learning.

3.1 Agile Principles Applied in Agile learning

The purpose of this section is to reanalyse the 12 principles of Agile Manifesto through the educational context perspective, and to map them to effective teaching and learning practices. In [21], the 12 Agile Manifesto principles were translated to 12 so-called Corollaries to the Pedagogical Environment. The translation was done based on what the authors consider being the best fit for educational purposes and contains a mixture between software development related concepts and teaching and learning activities. We consider that in analyzing the principles, the correspondence between roles, goals, and deliverables in both contexts is essential. Agile Manifesto principles mention different actors like customers, business people, developers, development team, sponsors, users, or just team. In addition, there are specific software development artifacts or deliverables that are used: valuable software, requirements, project, technical excellence, design, and architecture.

Considering these, we reanalyse the 12 principles of Agile Manifesto and give application oriented interpretations for effective teaching and learning purposes:

P1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.

The teacher's highest priority is to help students gradually learn the concepts and techniques necessary to perform in a specific field.

The teacher decomposes the scope of the discipline into milestones, sets priorities and deadlines to allow students to learn continuously, and demonstrates knowledge of those concepts that bring the most value to them. Additionally, offering to students early knowledge, palpable skills, and hands-on activities could maintain and increase students' interest about the class.

P2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Welcome changing topics and teaching practices, even late in the teaching time frame. Agile education harnesses change for increasing the knowledge value.

The learning process would be a continuous negotiation between teacher and students, both trying to find the best way to acquire the necessary knowledge to perform in the field. Change is normal and it could affect the content of the lectures, the assessments, or the evaluation process. In the context of Agile learning, this could be translated to:

- Adopt a more flexible course content; the adaptation could be either for the whole group, small groups, or to individuals based on their in-time proven interest.
- Change the evaluation process and set the students' expectations in this direction (these changes could challenge the students to improve their logical thinking and adaptation to problem-solving tasks).

P3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Deliver learning results frequently, from a couple of days to a couple of weeks, with a preference to the shorter timescale.

This principle is strongly linked with the first one and we can translate it in the teaching/learning process by replacing the one-time evaluation technique at the end of the course, with a continuous evaluation flow that allows students to put immediately in practice what they learn during lectures. We adapted this principle for the academic education context referring to a time period between a couple of days to a couple of weeks to receive working deliverables/learning results from students.

Very good learning results could be obtained if these deliverables are developed during the lectures, with 15-20 minutes dedicated to transfer the knowledge and another 15-30 minutes to apply the learned concepts. This technique is called micro-learning and it is usually seen as a specific way to implement and Agile mindset in education [13].

P4. Business people and developers must work together daily throughout the project.

Teachers and students should work together frequently during the semester.

To support a continuous collaboration between students and teachers, a proper communication setup (contact information, communication tools, frequency etc) should be established at the beginning of the module.

We may consider two perspectives of communication: vertical and horizontal. Vertically, we have the professor as a root and the students as leaves, and the teaching assistants on a potential intermediate level. In an agile approach, a special focus should be given to bottom-up communication. This could be achieved by increasing the teachers' availability to answer to the students' questions, and through quizzes that could emphasize the problems of the students. Still, the top-down communication is also important and so students should actively check if there are any new questions, explanations, or requirements added by the instructors. Horizontally, the students' intercommunication is very much encouraged. This could be achieved by allowing students to answer to other students' questions and by working in teams to solve different tasks.

P5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

Empower students in driving classes content and the results and trust that most students are motivated to learn and contribute.

Teachers should create a proper learning environment during lectures or seminars and mentor students when needed, so that students feel motivated to learn. Teachers should play the role of a 'servant leader'[14], being focused on empowering students. Therefore, teachers are serving instead of commanding and they always look to help students grow in ways that unleash their potential and creativity. Moreover, students could choose the details of the practical work if these details lead to the same learning outcomes.

P6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.

Direct communication is essential, not only for delivering lectures but also to clarify all questions students might have.

Direct communication is usually based on face-to-face communication but could also be done through a proper online platform. In this second scenario, it is important to rely on synchronous interaction (i.e. audio and video calls), and not on an asynchronous one (i.e. text messages, e-mails or any other form of texting).

P7. Working software is the primary measure of progress.

An end-to-end learning result is the primary measure of progress.

Measuring progress is crucial to an effective learning process. In this respect, teachers should be able to capture and analyze all "ordinary student work" produced during a narrow time period, including assignments, homeworks, discussions, and group interactions, which represent a vertical slice of learning. All deliverables have to be carefully defined, paying a special attention to acceptance criteria.

If we consider ACM knowledge levels [11] we have: *Familiarity*, *Usage*, and *Assessment*. An important target would be to reach the level *Assessment* for as many topics as possible.

P8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Agile education promotes sustainable learning. Teachers and students should be able to maintain a constant pace indefinitely.

Progressing through a module with a constant pace is important, but finding that pace that is sustainable is crucial. Teachers should observe students' velocity and adjust the content and requirements based on that.

P9. Continuous attention to technical excellence and good design enhances agility.

Continuous attention to technical excellence and good design enhances agility; agility enhances learning.

Principles 1, 3, and 7 talk about delivering fast and frequent results. Nevertheless, students should focus on following good practices in development even if they are not part of the current discipline concepts. They should remain alert in verifying the completeness and correctness of their work.

P10. Simplicity–the art of maximizing the amount of work not done–is essential.

Understanding the concepts and applying them to simple and clear learning results is essential.

The basic learning objectives should be always achieved for the vast majority of students. To achieve this goal, information should be presented in a way that feels simple and intuitive to students, no matter how complex and challenging the theoretical aspects are.

P11. The best architectures, requirements, and designs emerge from self-organizing teams.

The best learning results emerge from collaborative learning.

In collaborative learning [8] two or more students collaborate as a self-organized team to learn new concepts or solve problems. Each student takes responsibility for his/her team to learn and succeed. This way, students will strengthen their skills by teaching their teammates or by learning new skills from them. Additionally, the best students in each group could be implied as mentors not only for helping others, but also for strengthening their knowledge.

P12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

At regular intervals, the teacher and students reflect and offer feedback on the effectiveness of the teaching/learning process.

The evaluation of the results should be discussed together with the students. It is important for the students to understand their accomplished level of competences. Students' self-evaluation could be also an important indicator, especially if it is explicitly compared with the teacher's evaluation.

3.2 A Practical Framework for an Agile Learning Approach

To come closer to a practical approach, we define a generic methodology that could be used for applying agile principles in the learning process.

Similar to the Agile development case, Agile learning uses iterative and incremental steps. In the development case, new functionalities are added in each iteration, while in the case of Agile learning the functionalities are replaced by new competences that should be attained by the students.

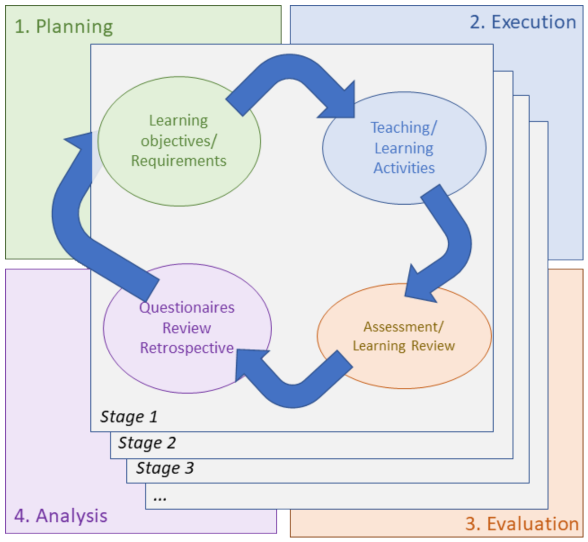
1. Planning
2. Execution
Learning objectives/ Requirements
Teaching/ Learning Activities
Questionaires Review Retrospective
Assessment/ Learning Review
Stage 1
Stage 2
Stage 3
...
4. Analysis
3. Evaluation

Figure 1: An overview of the Agile learning framework.

The proposed methodology is based on applying iterative stages, each one consisting of four activities: planning, execution, evaluation, and analysis. As stated in [12], before defining the stages, an evaluation of the students' prior knowledge could be extremely useful.

(1) *Planning:* In the first stage, an initial plan that specifies the general learning outcomes is developed. This contains the identified sequence of stages based on the covered topics, but without specifying all the details about the teaching and learning objectives involved in each stage. The specific learning objectives of one stage could be seen as associated vertical functional slices. In the next stages, the plan is updated and enriched with details starting from the corresponding learning objectives, but also based on the information accumulated from the previous activities. We define the concrete work that leads to the associated objectives achievement. The associated topics remain, but how thoroughly we present them in the teaching process, and how much knowledge we should expect from students to accumulate will be established in the planning activity at the beginning of each next stage.

(2) *Execution*: In this activity, the execution of what has been planned in the previous Planning phase is done. This comprises mainly the teaching activities: courses, seminars, laboratories presentations and materials, together with the learning activities that could be based on quizzes or practical assignments.

(3) *Evaluation:* Following the principle P3, the evaluation is done at each stage. It could be done by evaluating the practical work or the quizzes results, through presentations, discussions and debates. The evaluation process reviews the level of knowledge that has been attained in the corresponding stage.

(4) *Analysis:* Besides the level of knowledge that has been attained, a deep analysis of the entire learning process should be analysed. Principles P8 and P12 recommend this, and to achieve this, it is necessary to use a process similar to the retrospective from Scrum methodology: the first step is to identify the things that need to be changed and the second one is to propose some new practices which could improve the teaching/learning process. Choosing the most appropriate level of difficulty, time frames, and evaluation methods are on their turn not easy tasks - this is why this analysis is essential before going to the next stage where all these will be considered.

The evaluation is spread throughout the stages, with an optional final evaluation, which could be used to evaluate if the correlations between the competences acquired in each stage were correctly established. Finally, the professor could evaluate the entire process and extract some 'lessons learned' that might be helpful for the future.

4 PDP USE-CASE

To demonstrate the application of the interpreted principles and the proposed framework, we have considered applying agile learning practices in a course that treats parallel and distributed programming. Parallel and Distributed Programming (PDP) is a course offered in the fifth semester of study, and deepens the knowledge related to threads, synchronization through semaphores, conditional variables, monitors and barriers and asynchronous tasks through futures and promises. All these topics are treated both theoretically and practically. Some of these were treated before but only until a certain knowledge level: 'Familiarity' or maximum 'Usage' in ACM classification of the level of knowledge [11]. Tasks partitioning with performance evaluation are theoretically discussed, analysed, and then applied in practical assignments, too. An overview on the main parallel and distributed patterns is presented, and in addition new programming topics such as: Message Passing Interface (MPI), OpenMP and Compute Unified Device Architecture (CUDA) are introduced.

The agile approach was mainly focused on practical work, but since it is in strong connection with theoretical knowledge, the lectures' structure was influenced as well.

From the lecture point of view, choosing the right order in which the knowledge is structured is not influenced only by the content; there are also other (sometimes conflicting) factors which should be considered: going from simple to complex, capturing the interest of the students, increasing the level of competition, etc. Therefore, if we would like to keep a high level of interest among the students, the natural order from the theory point of view might not be the best choice. For our concrete case, this approach was implemented by moving MPI programming forward before going to deepen the multithreading and concurrency concepts (for which some previous knowledge exists), a change that increased the interest of the students for the course. In addition, each lecture was finished with a short quiz and the possibility to add remarks. This facilitates cooperation between the professor and students and allows a minimal evaluation of the level of understanding of the concepts. This was in correspondence with P4 and P3 principles.

Agile strategy was focused especially on the practical work, and following the proposed framework, we started these classes with a quiz with the purpose of evaluating the previous level of knowledge related to the topics that are going to be discussed; this was due to the fact that not all topics were absolutely new.

Two types of works were applied:

- *in-class* exercises – they preceded the practical assignments related to a specific topic,
- assignments – a specific problem is required to be solved and delivered by a certain deadline.

The *in-class* exercises are done in a group of 15 students together with the coordinator professor. The reason of introducing "in-class" exercises is to facilitate:
- starting to work with the programming mechanisms which were theoretically presented;
- free discussion and analysis;
- deepen the collaboration;
- overpass impasse induced by the necessity to start from zero.

Depending on the concepts and competences that are implied, we set the following stages.

Stage1 : Performance oriented multithreading
Objectives:
- multithreading for increasing the computational performance;
- task partitioning; performance evaluation.

Stage2 : MPI and OpenMP
Objectives:
- MPI multiprocessing computing
- OpenMP Multithreading computing.

Stage3 : Conditional synchronization
Objectives:
- conditional synchronization: wait-notify mechanisms;
- producer-consumer problem

Stage4 : Complex Projects
Objectives:
- Client-Server with multithreading;
- CUDA programming.

Each stage contains an *in-class* laboratory and practical assignments.

For the first stage, the practical work was defined as:

C1 $\Rightarrow$ *in-class*: adding two vectors each containing n numbers
L1-L2 $\Rightarrow$ *assignment*: transform an image represented as a matrix of pixels by applying a kernel:
L1 the output matrix is different from the input matrix;
L2 input matrix is transformed and no other additional matrix can be used (decreasing space complexity).

The execution of the assignment was done individually by each student, according to principle P5.

The evaluation was done also during the *in-class* laboratory, but especially through the delivered assignments. The delivery imposes not just to upload the source code but also to present the work in the same group of 15 students. Through these, the execution is verified but also the critical analysis of the solution and of the obtained performance. This way the assignment verification facilitates:

- a good verification of the execution and of the reported performance (i.e that it really corresponds to what is really achieved);
- the level of understanding of the delivered work – if the student completely understood the programming mechanisms that were used, and what impact do they have on the performance;
- help the students to learn from their colleagues' solutions (possible alternatives/improvements/vulnerabilities).

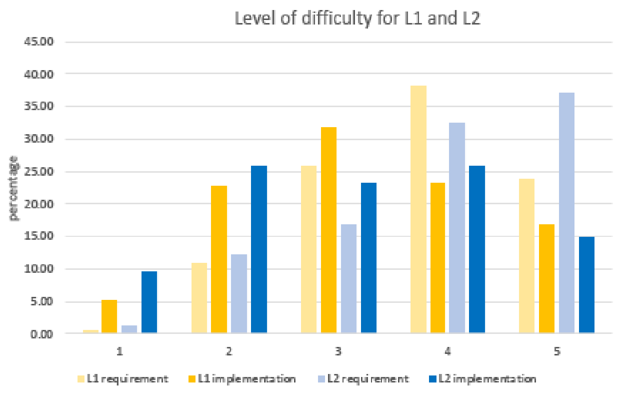

Figure 2: The difficulty level perceived for understanding requirements and for implementation (Laboratory L1 and L2).

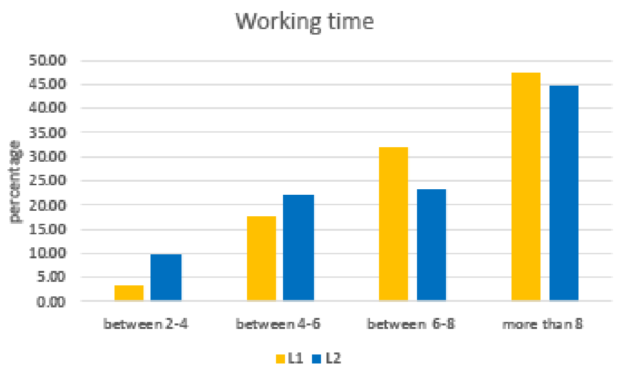

Figure 3: The working time needed for L1 and L2.

The analysis phase was done through retrospectives. The questions of the first retrospective are presented in Table 1.

Table 1: The questions of the first retrospective

1	"How difficult you think it was the understanding of the requirements for the laboratory L1/L2)?" (P10)
	Result: average scores from 5 points: L1 ⇒ 3.24; L2 ⇒ 3.10; Figure 2
2	"How difficult do you consider solving the laboratory L1/L2 was?" (P10)
	Result: average scores from 5 points: L1 ⇒ 3.75; L2 ⇒ 3.92; Figure 2
3	"Did you deliver the laboratory L1/L2 and if yes at which deadline (first or second)?" (P8)
	Result: - Figure 4
4	"How many hours did you need for solving the laboratory L1/L2?" (P8)
	Result: Figure 3
5	"How helpful were the *in-class* exercises for solving the laboratory L1/L2?" (P12)
	Result: average score from 5 points = 3.75
6	"Would you prefer to replace *in-class* exercises by adding more delivery time?" (P12)
	Result: (Yes) ⇒ 34, (No) ⇒ 120
7	"How much helpful was the attendance at the delivery presentation of your colleagues?" (P4, P6, P12)
	Result: average score from 5 points = 2.90
8	"How often did you verify the class materials related to laboratories that were added in the platform?" (P4) (a)several times per week, (b)weekly, (c) each 2 weeks, (d) only when it has been explicitly suggested?"
	Result: (a) ⇒ 47; (b) ⇒ 69; (c) ⇒ 17; (d) ⇒ 21
9	"How often did you verify the class materials related to lectures that were added in the platform?" (P4) (a)several times per week, (b)weekly, (c) each 2 weeks, (d) only when it has been explicitly suggested?"
	Result: (a) ⇒ 25; (b) ⇒ 73; (c) ⇒ 29; (d) ⇒ 27

The working time is important to be evaluated to verify the estimated time for the work associated with the considered subject.

Questions 5 and 6 evaluate if the *in-class* exercises done by a group of students together with the professor are really useful and needed. The structure of the work should be done by allowing the team to set the working procedures. The result shows that we have to keep this way of working.

Question 7 emphasises that the direct discussions are essential to establish good communication and fast feedback about the results.

The last two questions show the need to establish a good communication structure that should be followed by the students, too.

The first retrospective, which was filled in by 154 students, was followed by a deep analysis that reveals several problems, and as a result, some adaptations have been done. These were related to:

- the delivery terms,
- adaptation of the requirements to estimated work time;
- additional explanations of the requirements;
- hints related to the possible solutions.

We started with two possible delivery terms for each assignment (the first for a maximal grade equal to 10, and the second for a maximal grade equal to 8). From the first retrospective analysis, we found out that there are students that started to work on an

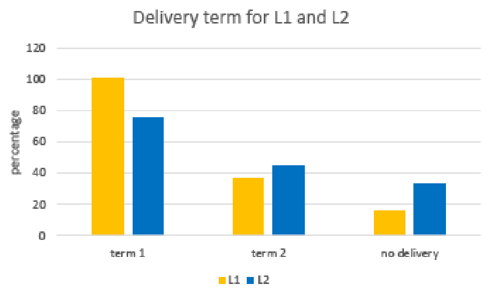

Figure 4: The delivery terms for L1 and L2.

assignment but they could not finish it in due time; In order to encourage them not to completely drop off the realization of some assignments, we added a third delivery term. This was justified by the fact that the main objective is to assure attaining the associated competences by a large number of students (P1 and P10) – if an assignment is finally delivered, this means that the students went through the challenges of that assignment and surmount the difficulties.

Another thing that the first retrospective showed was related to the fact that many students need more explanations related to the requirements and the possible approaches that could be followed to achieve the solution. As a consequence, beside the textual description of the assignment description, we add a visual representation of these. A concrete example is given in the Appendix. Qualitative analysis based on informal feedback shows that these were well appreciated by the students.

The second stage started by applying these adjustments in the planning phase and by defining the details of the following assignments: the problems to be solved (with a specific level of complexity and with work times approximated based on the previous retrospective).

Due to the lack of space, in what follows we will focus more on the following retrospectives that have been done for the next stages, without giving all details about the concrete assignments (problems), their execution and evaluation.

Table 2: The questions of the second retrospective

" How difficult do you consider solving the laboratory L3 was?"
Result: average scores from 5 points: 3.97
"How many hours did you need for solving the laboratory L3?"
Results for L3: between 4-6 ⇒ 19; between 6-8 ⇒ 33; between 8-10 ⇒ 26; more than 10 ⇒ 13;
" Did you deliver the laboratory L3 and if yes at which deadline?"
Result: - Figure 5

The second retrospective was related especially with MPI parallel computing (Assignment L3), since for OpenMP we introduced only the lecture classes and the "in-class" exercises. The decision not to introduce an OpenMP based assignment was taken based on the work-time analysis. The questions of the second retrospective were similar to those of the first retrospective (Table 2).

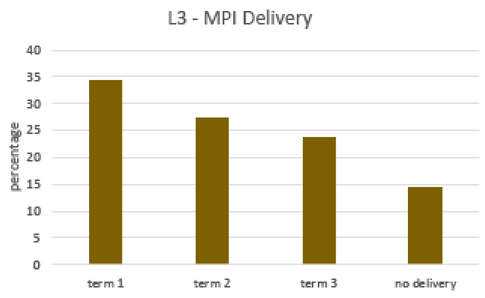

Figure 5: The delivery terms for L3.

The delivery terms analysis for L3 shows that by adding an additional delivery term, more than 20% of the students managed to fulfill the requirements even if with some delay. Since our primary objective is to form competences, this was an important adjustment. The laboratory was considered difficult especially because it imposed a new way of thinking about partitioning between processes and how they could communicate inside MPI. It should be mentioned that at each term the maximal grade decreased with 2 points, and even so the additional delivery term did not decrease the willingness to finish in time.

The third retrospective contained similar questions, but related to assignments L4 and L5 of the third stage.

Table 3: The questions of the third retrospective

"How difficult do you consider solving the laboratory L4/L5 was?"
Result for L4: average score from 5 points: 3.39 Result for L5: average score from 5 points: 3.26
" How many hours did you need for solving the laboratory L4/L5?"

Results for L4: between 4-6 21 between 6-8 ⇒ 38;
between 8-10 ⇒ 20; more than 10 ⇒ 13
Results for L5: between 4-6 ⇒ 41; between 6-8 ⇒ 24;
between 8-10⇒ 13; more than 10 ⇒ 13

" Did you deliver the laboratory L4/L5 and if yes at which deadline?"

Results for L4: first term ⇒ 59; second term ⇒ 22;
third term ⇒ 10; no delivery ⇒ 3
Results for L5: first term ⇒ 62; second term ⇒ 16;
third term ⇒ 8; no delivery ⇒ 8

The 4th stage introduced a different execution organization: the students worked in teams of two students. This brings the advantages of pair programming but also could reduce the working time.

The fourth retrospective was oriented to estimate the difficulty, delivery and working time for the two projects - T1 and T2. In addition, there were other two questions related to working in pairs for the projects (Table 4).

The results for these questions are shown in Figure 6.

For the CUDA project, the problem to solve was the free choice of the students, of course under the approval of the professor. This was inline with principle P5.

Agile Learning Impact

The impact of using agile learning was evaluated through qualitative and quantitative analysis. The qualitative analysis was based on free remarks of the students, which were positively related to the agile learning approach. Some samples are given:

- "The additional explanations of the requirements were very useful. Visual explanations help a lot."
- "I appreciated the fact that it was an adaptation based on our feedback (especially the additional deadline for delivery)";
- "It's good that we have been asked about the problems that we encountered."
- " The problems were difficult and needed a lot of time for solving, but the additional explanations were very useful"
- "I could improve my implementation by seeing others' solution."
- "In the initial starting quiz, I evaluated myself as knowing more than I really did. Finally, I saw that there are topics that I should study more."
- "In-class exercises helped me a lot".
- "The additional deadline gives me a boost to finish the laboratory."

A *quantitative analysis* based on a comparison of the grades from the previous year is shown in the Figure 7. In the previous year (2019-2020), the activity has been done normally, in face-to-face way, and there were additional seminar classes besides the lectures and laboratory classes, while in the current year the classes have been done online due to the Covid pandemic, and there were only lectures and laboratory classes. As it can be seen from the results, the differences are not important, but it is important to note that using an agile approach we managed to cover the online activity disadvantages and the lack of seminar classes.

Overall, the agile methodology proved to be very useful:

• for professors: to better understand the level of the students during the semester and make the needed adjustments,
• for the students: to work with more enthusiasm, and try hard in the conditions when their issues were taken into consideration.

Table 4: Particular questions of the forth retrospective

"How much useful was the fact that you work in pairs for projects T1 and T2?"
"Did you reduced the time of solving because you work in a team of 2 students for project T1, respectively T2?"

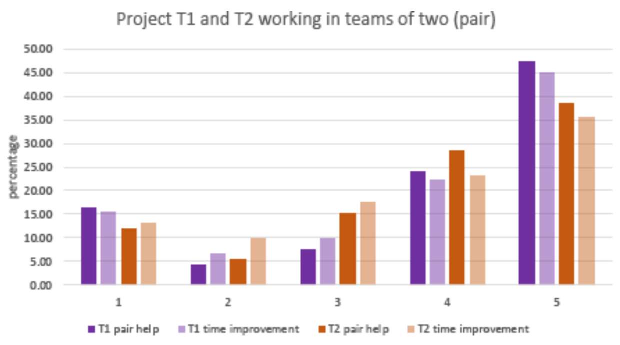

Figure 6: The retrospective for projects T1 and T2.

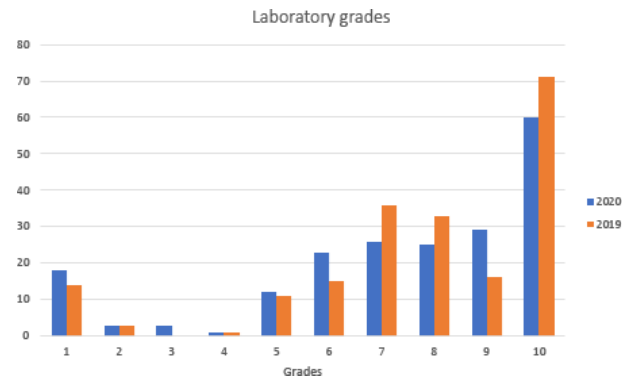

Figure 7: The grades for the practical works in 2019-2020 and 2020-2021.

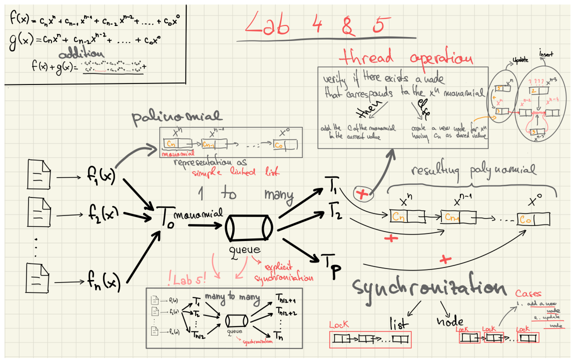

Figure 8: Visual representation of the requirements for Laboratory 4 and Laboratory 5.

5 CONCLUSIONS

In order to go towards a more practical oriented perspective of Agile instructional design, we started by analysing the 12 Agile Principles from Agile Manifesto, and then provide interpretations and a methodology to be useful for concrete applications in practice.

Then, we have concretely applied them inside a course. We considered as a use case a course oriented on parallel and distributed programming, with a special focus on practical abilities and competences. Parallel and distributed programming is, in general, not easy and applying teaching methodologies that facilitate the learning process is needed.

The learning processes were organized in stages/iterations that correspond to the continuous increase in students' abilities and competences. After each iteration, a retrospective quiz was organized, and based on them several dynamic adaptations were set.

The approach was very well received by the students, especially because of their direct involvement in the educational process, but also because a dynamic adaptation was possible inside this setting. The qualitative and quantitative analyses reflect all these advantages.

ACKNOWLEDGEMENT

This work was partially supported by Robert Bosch Corporation.

Appendix

Besides the textual descriptions and verbal explanations, graphical form explanations were provided. For example, for L4 and L5 the problems were: adding large size polynomials represented using linked-list – L4: using different grain size synchronization; L5: using different reading policies: one polynomial by thread or based on producers-consumers pattern (Figure 8).

REFERENCES

[1] Agile Alliance. 2001. Manifesto for Agile Software Development. (2001). https://www.agilealliance.org/agile101/the-agile-manifesto/

[2] S. Ambler and M. Lines. 2017. *An Executive's Guide to Disciplined Agile: Winning the Race to Business Agility.* CreateSpace Independent Publishing Platform.

[3] S. Ambler and M. Lines. 2020. *Introduction to Disciplined Agile Delivery, 2nd edition.* Project Management Institute.

[4] K. Beck and C. Andres. 2004. *Extreme Programming Explained: Embrace Change, 2nd Edition.* Addison-Wesley.

[5] G. Blokdyk. 2019. *Agile Unified Process A Complete Guide.* 5STARCooks.

[6] D. A. Dewi and M. Muniandy. 2014. The agility of agile methodology for teaching and learning activities. *2014 8th. Malaysian Software Engineering Conference (MySEC)*, 255–259.

[7] Digital.ai. 2020. 14th Annual State of Agile Report. (2020). https://explore.digital.ai/state-of-agile/14th-annual-state-of-agile-report

[8] A. Goodfell (Ed.). 1994. *Collaborative Learning: A Sourcebook for Higher Education.* Natl Center on Postsecondary.

[9] R. Hoda. 2019. Using Agile Games to Invigorate Agile and Lean Software Development Learning in Classrooms. In *Agile and Lean Concepts for Teaching and Learning*, Parsons D. and MacCallum K. (Eds.). Springer.

[10] Project Management Institute. 2017. *A Guide to the Project Management Body of Knowledge.* Project Management Institute, USA.

[11] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.* Association for Computing Machinery. 144–154 pages.

[12] J. Longmuß. 2016. Agile learning: Bridging the gap between industry and university A model approach to embedded learning and competence development for the future workforce. In *44th SEFI Conference, 12-15 September 2016,Finland.*

[13] Gona Sirwan Mohammed, Karzan Wakil, Sarkhell, and Sirwan Nawroly. 2002. The Effectiveness of Microlearning to Improve Students' Learning Ability. Vol. 3. 32–38.

[14] Joe D. Nichols. 2010. *Teachers as Servant Leaders.* Rowman & Littlefield Publishers.

[15] V. Niculescu, A. Sterca, and D. Bufnea. 2020. Agile and Cyclic Learning in Teaching Parallel and Distributed Computing. In *Proceedings of the 2nd ACM SIGSOFT International Workshop on Education through Advanced Software Engineering and Artificial Intelligence.* ACM, 27–33.

[16] I.E. Noguera, Guerrero-Roldán, and R. Masó. 2018. Collaborative agile learning in online environments: Strategies for improving team regulation and project management. Vol. 116. 110 – 129.

[17] S. R. Palmer. 2002. A Practical Guide to Feature-Driven Development. Prentice Hall.

[18] D. Parsons and K. MacCallum (Eds.). 2019. *Agile and Lean Concepts for Teaching and Learning: Bringing Methodologies from Industry to the Classroom.* Springer.

[19] K. Rubin. 2012. Essential Scrum: A Practical Guide to the Most Popular Agile Process. Addison-Wesley Professional.

[20] P. Salza, P. Musmarra, and F. Ferrucci. 2019. Agile Methodologies in Education: A Review. In *Agile and Lean Concepts for Teaching and Learning: Bringing Methodologies from Industry to the Classroom.* Springer Singapore, 25–45.

[21] J. C. Stewart, C. DeCusatis, K. G. Kidder, J. R. Massi, and K. Anne. 2009. Evaluating Agile Principles in Active and Cooperative Learning. In *CSIS, Pace University, 2009.*

[22] W. Wijnands and A. Stolze. 2019. Transforming Education with eduScrum. In *Agile and Lean Concepts for Teaching and Learning.* Springer.