DESIGN PROJECT #RED 7

Renato Chavez s2552558
Shun Nishijima s2977921
Mateusz Bartnicki s3006891
Oliwier Szwalek s2990091

Introduction(a brief report)

The report we made shows structural and behavioural design and what we thought of making a particular design of the software system which is the new TARP learning platform. It includes the interview to the stakeholder in the requirements elicitation phase and also insight from this interview. We also describe the test plan.

Each content has a particular purpose. For constructing requirements, we decided on particular stakeholders and interviewed one of the specific stakeholder as requirement elicitation. From the result of requirement elicitation, we have some idea of user insights to develop software systems which cover all user expectations. Functional requirements and non-functional requirements should cover all expectations.

Use Case Diagrams describe whole TARP system functions that users use, so that it explains the functions which are mentioned at functional requirements. Moreover, there are two extended user stories that describe some specific part of the system and object diagrams are designed as models of these stories.

Whole system is modeled as a class diagram. And also, all structured models and behavioral models show that some specific part of the system as a model. We used Activity Diagram, Sequence Diagram and State Machine Diagram. Each way of modeling has a specific purpose which is explained in the report.

We also mentioned the Testing plan and Metrics of this software system.

In sum, we have made a design report for developing and defining the software system called TARP. The report includes user expectations and several designs which are used to describe specific parts of the system explicitly.

Stakeholders/Actors

- Stakeholders

Users: students, TAs, teachers, module coordinators
Owners: University of Twente
Developers, Builders, Designers, Testers, Maintainers: Mateusz Bartnicki, Renato Chavez, Shun Nishijima, Oliwier Szwalek
Operators: People who work in UT
A relevant stakeholder, with who an interview was conducted: Junseo Kim (Teacher)

- Actors

Students, TAs, Teachers, Module coordinators, Copy machine(possible)

Interview Report

Interviewer:Renato Chavez, Shun Nishijima , Mateusz Bartnicki , Oliwier Szwalek
Interviewee: Junseo Kim(as role of Teacher)
Time: 25th/Nov/2022 13:00-13:30

Place:SP3

Summary

The biggest complaint with the current Learning management system is what the teacher needs to open each student's submission during grading. He prefers bulk download for all submissions for printing them. He grades submissions by pen and paper, so it is not necessary to grade in systems. He said that it is hard to switch from grading by pen and paper because he used to do it for a long time. If we integrate more with the system, it would be a kind of spreadsheet related system. Another complaint is that he just puts grades manually in the system currently. The system could have lists of students and grades. Also it could be that his own list can be submitted to the system and can show grades in the system.

He doesn't have big problems with the interface of the system. He hasn't opinion about avoiding academic fraud. All the information he needs is just enough for managing the courses. He mentioned name, identification number, e-mail address and some sort of contact details are necessary. It is necessary for students to know where they lost the points and what the overall score was. Apart from that, he doesn't have preference.

Effects on our product

- Develop bulk download can be completed by clicking the button once.
- We can't change how to grade from using pen and paper.
- Add list form of grade.(or upload spreadsheet and show contents)
- We can only collect names, identification numbers, e-mail addresses and some sort of contact details.
- We need to make a function of comments to show why the grades were, but we don't need much about.

Prepared question for the interview

- What would make you switch from canvas to Tarp?
- What are your biggest complaints when it comes to Canvas?
- Is there anything you miss on Canvas that you would make a great use of?
- Would you like to assign users to the courses and can remove them?
- What would be the ideal interface for you?
- Are you looking for an anti-fraud/plagiarism system?
- What permissions would like to be given as a teacher?
- What do you want to be able to do with the submission field as well as the submissions themselves?
- What kind of data should your students be able to access?
- What permissions would you like for the students to have?
- Which requirements are most necessary? Bulk downloading.
- Which roles would the teacher role oversee (higher in hierarchy)?
- How should personal data be handled?
- How many people would use the system?
- Who will be using it?

- What kind of account management?
- Which requirements are most necessary?
- What kind of info should teachers see about submissions? (number of attempts, date etc)
- What would be the ideal balance between functionality and ease of use of the software?

User Story

functional

- As a module coordinator I want to be assigned to a course so that I can and remove other users from it.
- As a teacher, I want to avoid students editing their own grade so that I can manage user roles.
- As a member of the exam office, I want a system with m and s numbers to identify students and staff.
- As a staff member, I want a system with role hierarchy to make assigning account permissions easier.
- As a teacher, I want to be able to upload assignments without showing them to the students so that the TA's can look at them beforehand.
- As a teacher, I want to attach files and textual descriptions to the assignment.
- As a teacher, I want to see late submissions flagged as "late".
- As a teacher, I don't want to allow student submitting assignment after closed-for-submission-date
- As a teacher, I want to limit the file types whenever an upload from a student is required.
- As a teacher, I want to allow student to re-submit the assignment
- As a teacher, I want to track a student's upload, so that indication is sent to the student.
- As a teacher I want to be able to bulk download and upload assignments, so that I can use my preferred external auto-grading tools.
- As a teacher, I want to create assignments for my course.
- As a teacher, I want to enable and disable modifications of the assignment for other teachers or TAs.
- As a teacher, I want to attach comments to the assignment.
- As a teacher, I want to publish grades simultaneously.
- As a student, I want to get notifications each time a grade is posted, to stay up to date with my academic progress.
- As a teacher, I want to see a visible interface, so that I can use external auto-grading tools easily.
- As a teacher, I want to bulk download and upload annotated submissions and associated grades.
- As a teacher, I want to see the number of submission attempts per student per assignment so that I can punish them accordingly.
- As a teacher, I want to use a useful interface, so that I can use plagiarism checkers easily.

Non-functional

- As a client, the user wants to follow GDPR not to save unnecessary personal data.
- As an older staff member I want a clear and user friendly GUI, to navigate the system easily.
- As a teacher, I want to have a secure system, to avoid leaking any confidential data to students.
- As a university administrator, I want a product that's compatible with all the most popular platforms, to make sure users with different devices can use it.
- As a university IT employee, I want an easy to update system, so that I can add any features or fixes without needing external help.
- As an everyday user, I want a well performing product, so that I can use it without straining my device.

Use Case Diagram 1(Teacher&TA)

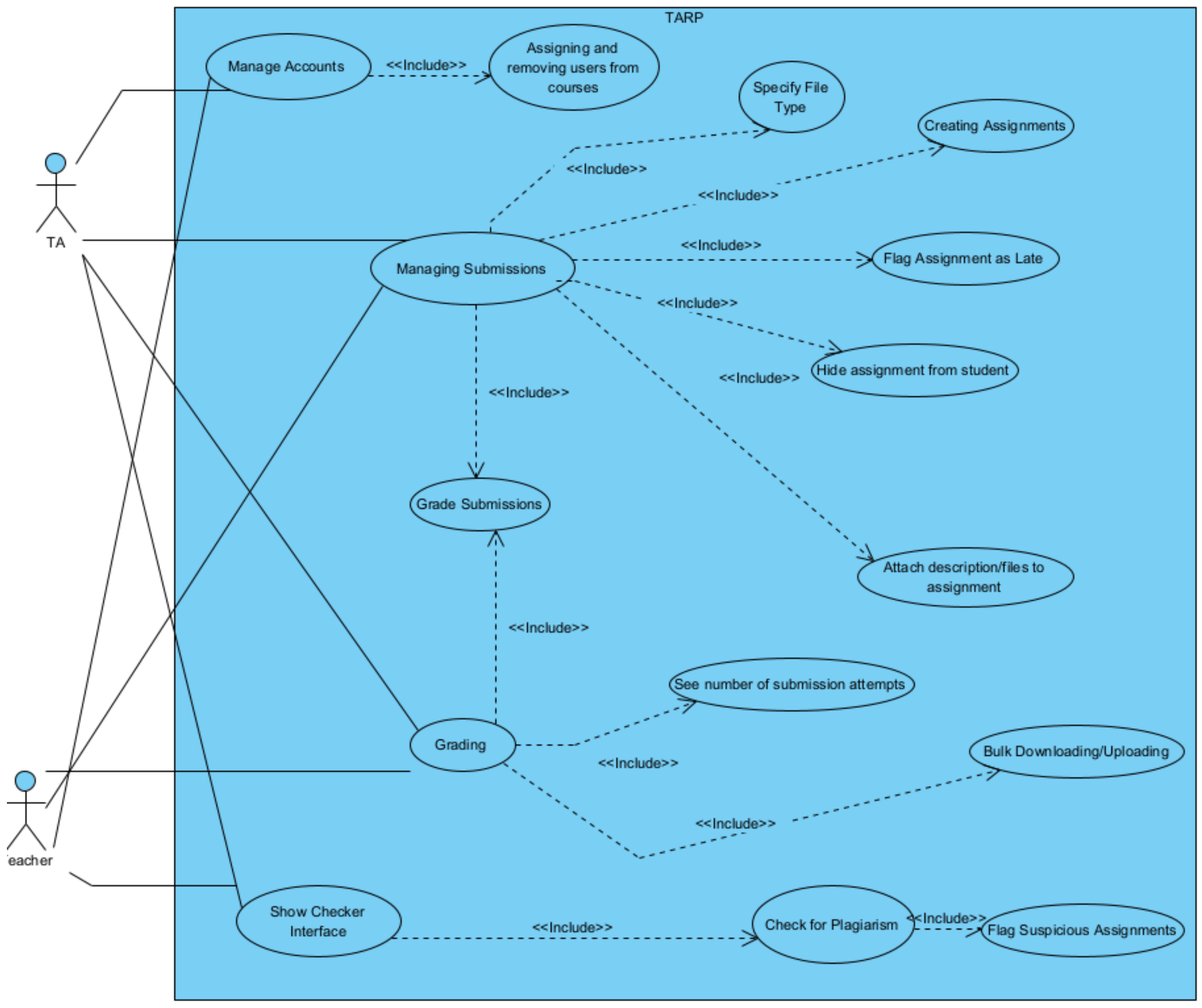


Diagram Description:
All the core functionalities for a teacher and TA are shown. Both can manage their account which mainly includes assigning and removing users from courses. Both can also manage the submissions, meaning they can create assignments, grading them in bulk as well as

modifying the fields in various ways. The system also involves grading functionality, most importantly the bulk download and upload of submissions in order to give the person checking the option of doing it on paper or use their third party auto grading tool of preference. Lastly the service will contain an easy to use plagiarism checker interface for an external tool that then can flag suspicious assignment, which again can be used by both.

Use Case Diagram 2(Student)

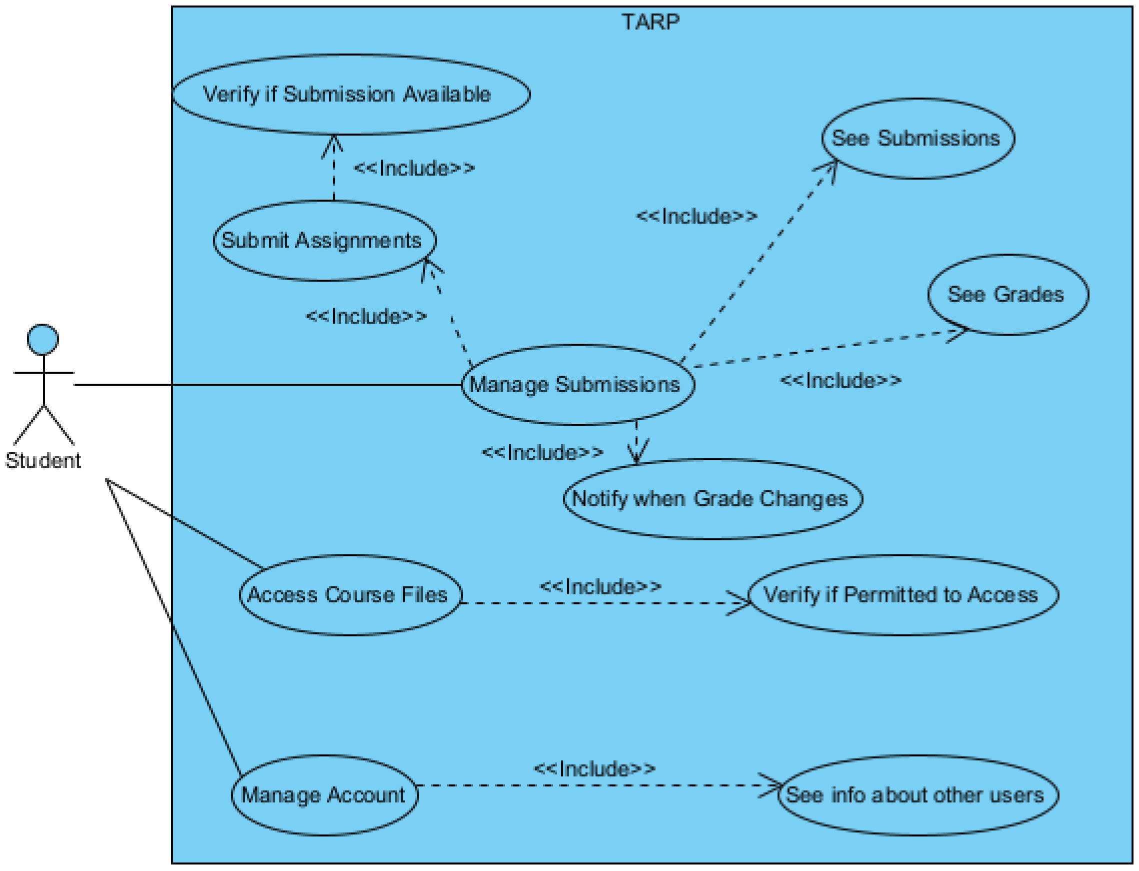


Diagram Description:
The student actor can mainly manage their submissions, including submitting (only if the window is available), viewing the said submission and the grades for them and getting notified when their grade changes. The service also allows the students to access permitted course files and use the account manager, mostly for seeing basic contact information of their peers.

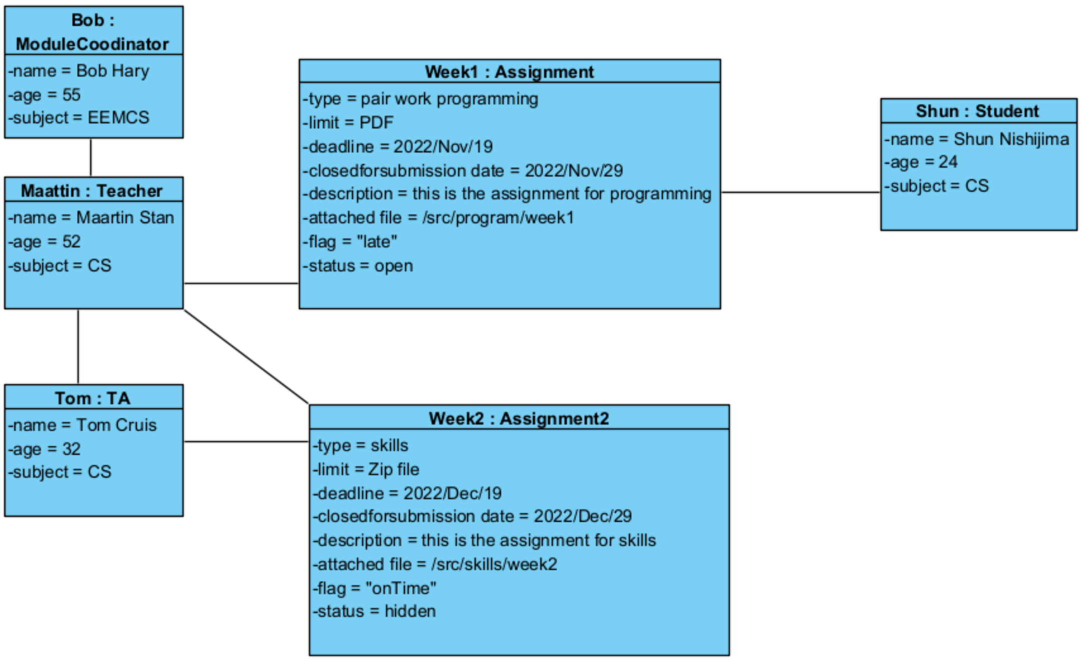
Extended User Story

Case 1: Creating Assignment

As a teacher, I want to create assignments for my course and I hope the assignment contains a textual description of the task and some attached files. Moreover, I want to set a deadline and a closed-for-submission date which means no submission is allowed after the date. I also need to make a submission being flagged as "late" after the deadline.

Furthermore, as a teacher, I sometimes want to hide the contents of an assignment from students until it is published, because TAs can check and modify the assignment.

Object Diagram

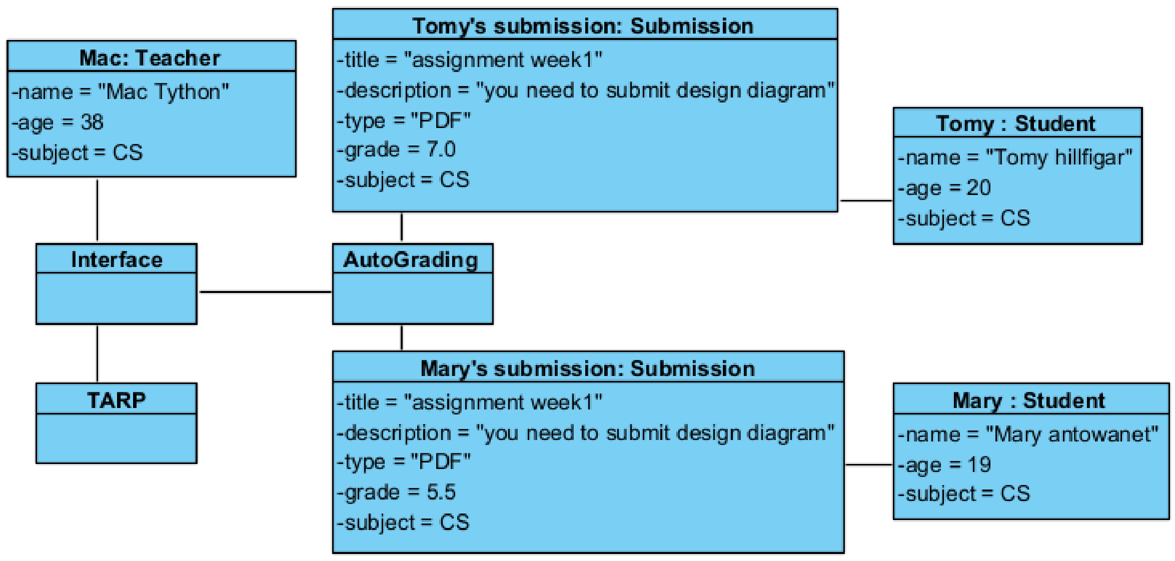


Description:
The diagram shows how the system of creating assignments works for each object. One of the assignments has a “late” flag and status is open in variables. Another one has variables which show that status is hidden. TAs can modify assignments which are hidden from students, so TAs have association with assignment2.

Case 2:Auto Grading

As a teacher, I want to use automatic grading that is an easy interface to use with my own tools for automatic grading. Thus, I want to bulk download all submissions. Moreover, I want to provide test-case reports which are the result from automated tests. Also, I hope it is possible to record how many submissions attempts a student has made using this number for either reducing the grade or limiting the number of allowed submissions attempts.

Object Diagram



Description:
Teachers want to use external autograding systems at the interface in the TARP system. The Autograding system works for all submissions and automatically grades them.

Requirements

Function

The Account Management System should involve functionality to assign users to the courses and remove them. Additionally an admin could change user roles, for example a module coordinator into a teacher. Personal profiles of users should include personal info like m/s numbers, names and contact information.

The Creating Assignments and Uploading Files functionalities should allow users to create assignments and hide them from students if they wish to. With the created assignment the user should have the option to attach text description and files with specifiable types. If the submission date has passed, the system should flag it as "late" and if the teacher chooses to, they can block any late submissions. On the other hand they can also choose to open the assignment for a re-submission.

Auto-grading should have an interface of all the tools available for it. Most notably it includes the option of bulk download of assignments for pen and paper grading. The number of submission attempts should be visible to the teachers.

The system should have an interface for a plagiarism checker, using which the teacher can export the assignments to a third party checking tool. The interface should be able to automatically flag assignments as suspicious if the external tool decides they are.

Non-Function

Performance :

The system should run smoothly on most devices, meaning it does not "freeze up" and does not overwork the CPU.

Security :

The database of the system should store only the name, email and student/employee numbers of users.

Compatibility :

The system should work for most of the mainstream software platforms.

Usability :

The GUI should be clear and simple for non-technical people.

Reliability :

The system should very rarely crash.

Maintainability :

The source code of the system should be well documented to help the university IT employees in updating and fixing the system.

Test Strategy

a) Scope of testing:
    - Easy to use account management system with a hierarchical role structure in which each "level" has different permissions over others. This includes adding and removing users from courses, creating the courses etc.
    - Simple submission management system. It includes functionality that creates assignments packaged together with any necessary files and comments. The system would also allow certain roles to mark some as late as well as hide them from a specific role while remaining visible to others. While creating the submission field, the creator should be able to specify the file type and the submission date. After the indicated window has passed, the field shouldn't accept any more submissions.
    - Grading functionalities, most important of them being a bulk download/upload option of submissions from a selected group of students. The number of submission attempts should also be visible.
    - Some sort of automatic plagiarism checker that would compare the submitted work with the most popular internet searches, similarly to one of the checkers available on the web.

b) Risks
   - One of the biggest risks would be a breach of the account management system that could leak all of the personal data of the users that were stored in the system.
   - Another big risk is if a bug caused unauthorized people to see the “hidden” assignments that were posted.
   - If the system failed to close the submission field after the deadline had passed, students could turn in late work without any consequences.
   - If data were to get corrupted during the bulk download/upload of submissions, it could negatively and unfairly affect the grades of many students.
   - A malfunctioning plagiarism checker could either make students lose points unfairly, or help the fraudulent ones avoid consequences if the communication with external tools got disrupted.

c) Test levels
   - Singular user: the perspective of the app/website visible to the singular user opening it, including the GUI, responsiveness of the website, what can be seen and what is hidden, depending on the role permissions of the user.
   - Module: all the teachers, TA’s and students in a single module, the assignments, grades and announcements that take place within it.
   - Course: the organization of the single course, all the modules included.
   - University: highest level of the system, encompasses everything and everyone in it.

d) Test types
   - Functional: tests that make sure everything works in the technical sense, meaning there are no bugs, glitches or unintended side effects that would affect the service negatively
   - Security: test if all the exchanges are properly encrypted and if confidential data cannot be reached from the outside by unauthorized parties.
   - Performance: see if the service runs smoothly, without too many stutters or lag while also putting as little strain on the users machine as possible.
   - User Interaction: test with users, see if the service feels intuitive to navigate, if the learning curve of using it isn’t too hard and the process isn’t annoying.
   - Surveys: send out surveys for the targeted demographic, ask what they would like to see in the system and then check if their expectations meet with what the current system offers.

Test Plan

a) Schedule :
   - The setup and preparation of all the resources required for testing should be completed within a month of development.
   - Testing of any feature should be completed within 2 days of its first implementation or any future adjustments of it.
   - The whole testing process and any final checks should be completed within 2 of development completion.

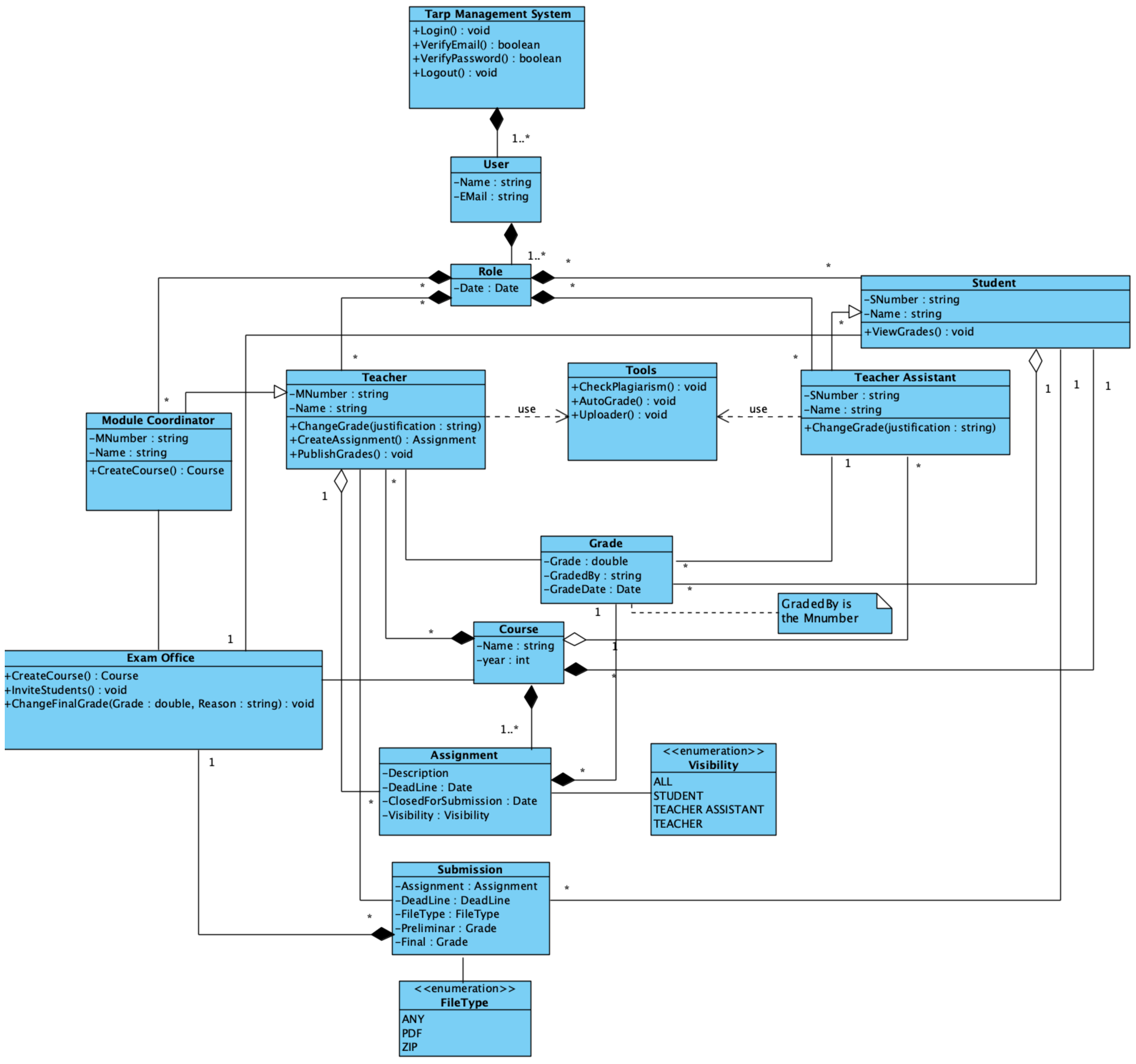
b) Roles & Responsibilities:
    - Test manager - oversees the whole testing phase
    - Testers - people who perform the manual tests and set up any automatic ones
    - Quality assurance - people who make sure that the testing is being done with accordance to the specified requirements

c) Deliverables:
    - Before testing: test schedule and plan document
    - During testing: data required for testing, testing tools
    - After testing: test results, logs, bug reports, summary of the process

d) Objectives:
    - The main objective of the testing is to ensure a service that is functional and secure, meaning there are slight to none chances of bugs and crashes as well as no possibility of confidential data leaking at any point.
    - The other objective is a product that is easy and satisfying to use, which doesn’t strain the user to find any function/data they might want to access and makes the navigation process intuitive and as a result satisfying as a whole.

Structured Model

Class Diagram

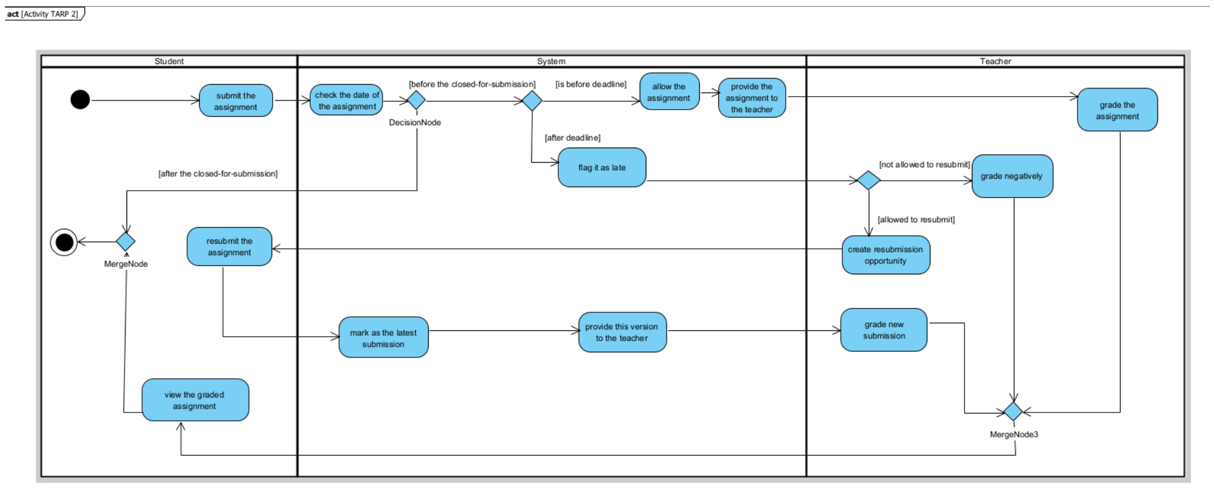


Description:
The class diagram follows the GDPR, general data protection regulations, which means only the necessary information will be collected from its users. Other than that, the class diagram includes just the most important methods so that the basic functionality of the system can be implemented for this. This should be a class diagram so the developing team can coordinate on the same idea for the learning management system with other entities like designers which should have no idea in programming. The diagram begins with a main learning management system diagram to implement the basic functionality of the site, for example login in with an account, verifying email, pass and logging out. After that the initial class for just implementing basic methods communicates with the role class for assigning the most basic role possible of a user, lower than a student, since users only contain a name and an email address, in contrast to students which also have a student number. Teachers and TA's make use of the external tools provided in TARP by the time grading is needed. A grade

depends on an m-number since only teaching people can grade the grades. Additionally Date should also be a parameter for Grade since it is needed the date when some grade is being changed or set so TARP can determine whether the assignment is already due, late, etc. Dependency arrows are mentioned in order to represent the hierarchical division between Module Coordinator > Teacher and Teaching Assistant > Student. We strived to do the class diagram as minimal as possible, since the first sketch included nasty class assignments return List<Course> to refer to Courses and List<pair<Student, Assignment>> among others, but this will end in a very occluded diagram, more difficult to present to the design team since it requires some technical knowledge. We strongly believe a good development team can make use of this diagram to implement tarp from its core functionalities.

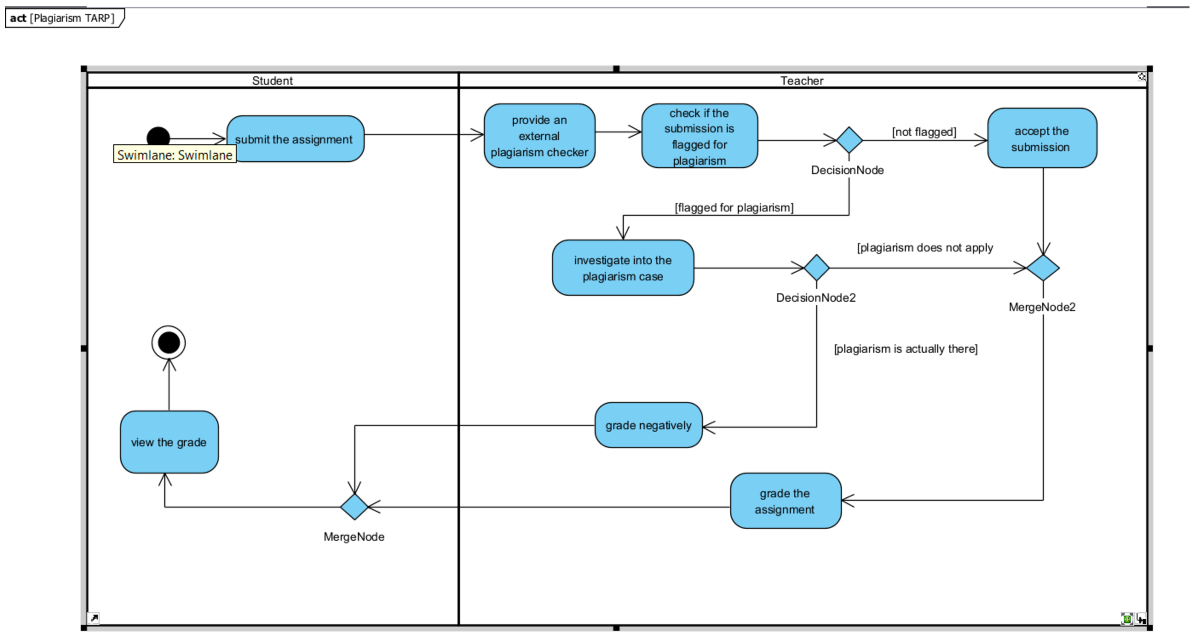
Behavioural Model

Activity Diagram 1: Assignment submission


act [Activity TARP 2]
Student
System
Teacher
submit the assignment
check the date of the assignment
[before the closed-for-submission]
DecisionNode
[is before deadline]
allow the assignment
provide the assignment to the teacher
grade the assignment
[after deadline]
flag it as late
[after the closed-for-submission]
[not allowed to resubmit]
grade negatively
[allowed to resubmit]
create resubmission opportunity
MergeNode
resubmit the assignment
mark as the latest submission
provide this version to the teacher
grade new submission
view the graded assignment
MergeNode3

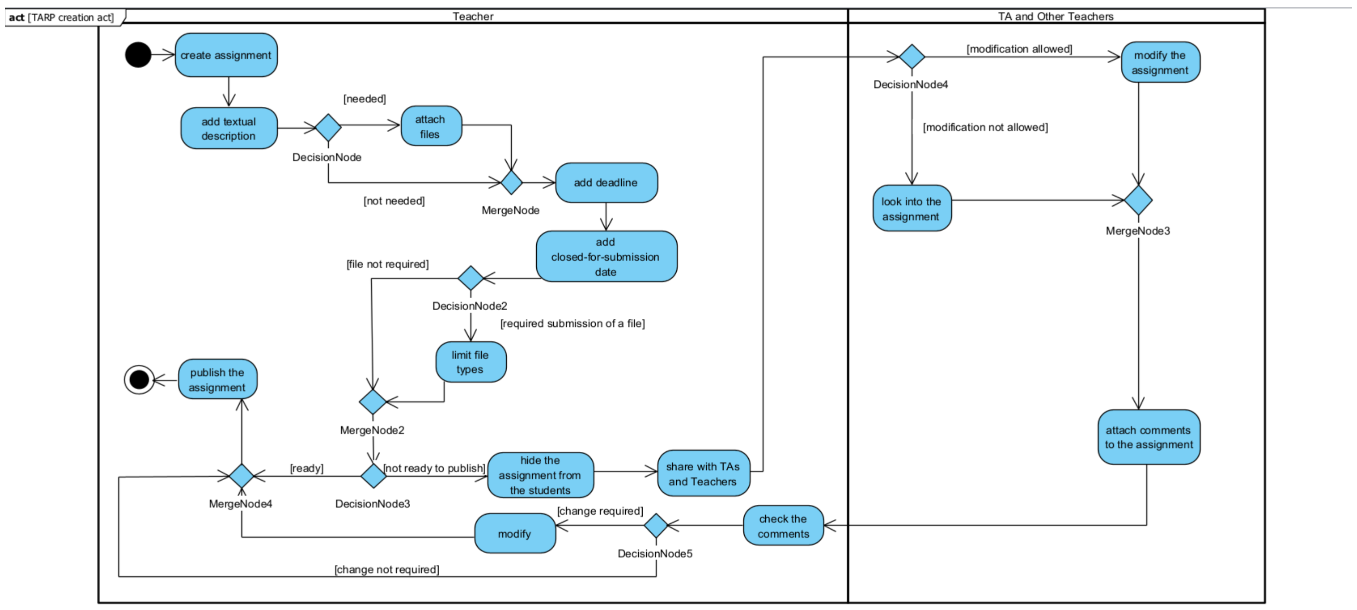
Description:
The activity diagram “Assignment submission” represents a series of actions that are made by a student, teacher, the TARP system and the interactions between them. “Assignment submission” explicitly gets into details of submitting the assignment by a student, which then it’s being processed and has the dates checked by the system, which is eventually provided to the teacher, so that he can either grade the submission, or if the deadline was not met - create another submission opportunity.
This particular enterprise of the process of a submission in the TARP system was chosen to be presented with an activity diagram, since to our personal belief it’s the best way to reflect how one action influences another activity and how interactions look globally.

Activity Diagram 2: Checking for plagiarism



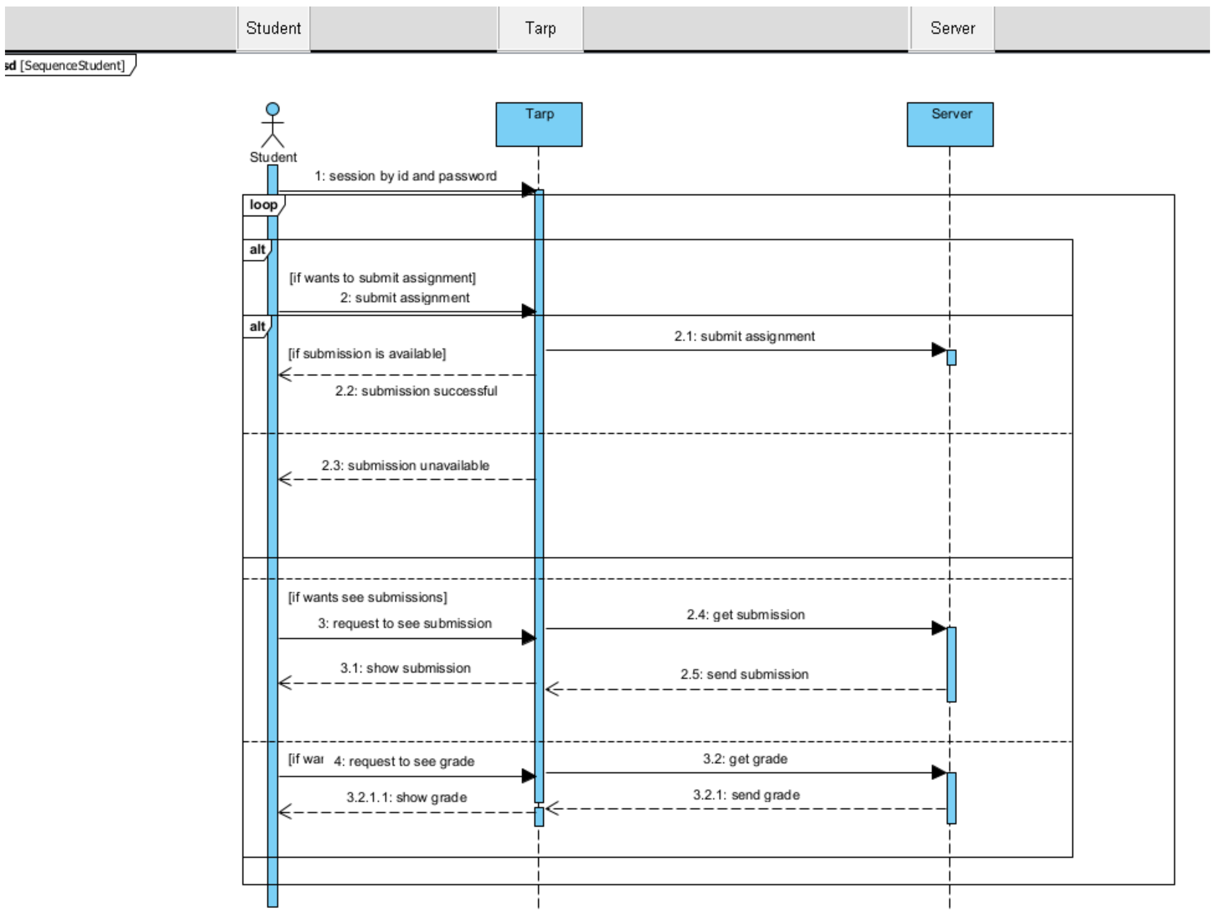
Description:
The activity diagram “Checking for plagiarism” represents a series of actions that are made by a student, teacher and how actions influence another. “Checking for plagiarism” diagram explicitly provides the view on such activities like attaching an external tool to the system, verifying submitted assignments and grading it. The diagram also represents how a student's action of submitting the assignment influences appropriate steps that will be taken by the teacher.
The activity of “Checking for plagiarism” was chosen to be presented by the activity diagram since it is the best and most understandable way to show how one action causes a reaction. The teacher’s choices, how to act under particular conditions are also explicit.

Activity Diagram 3: Creating the assignment



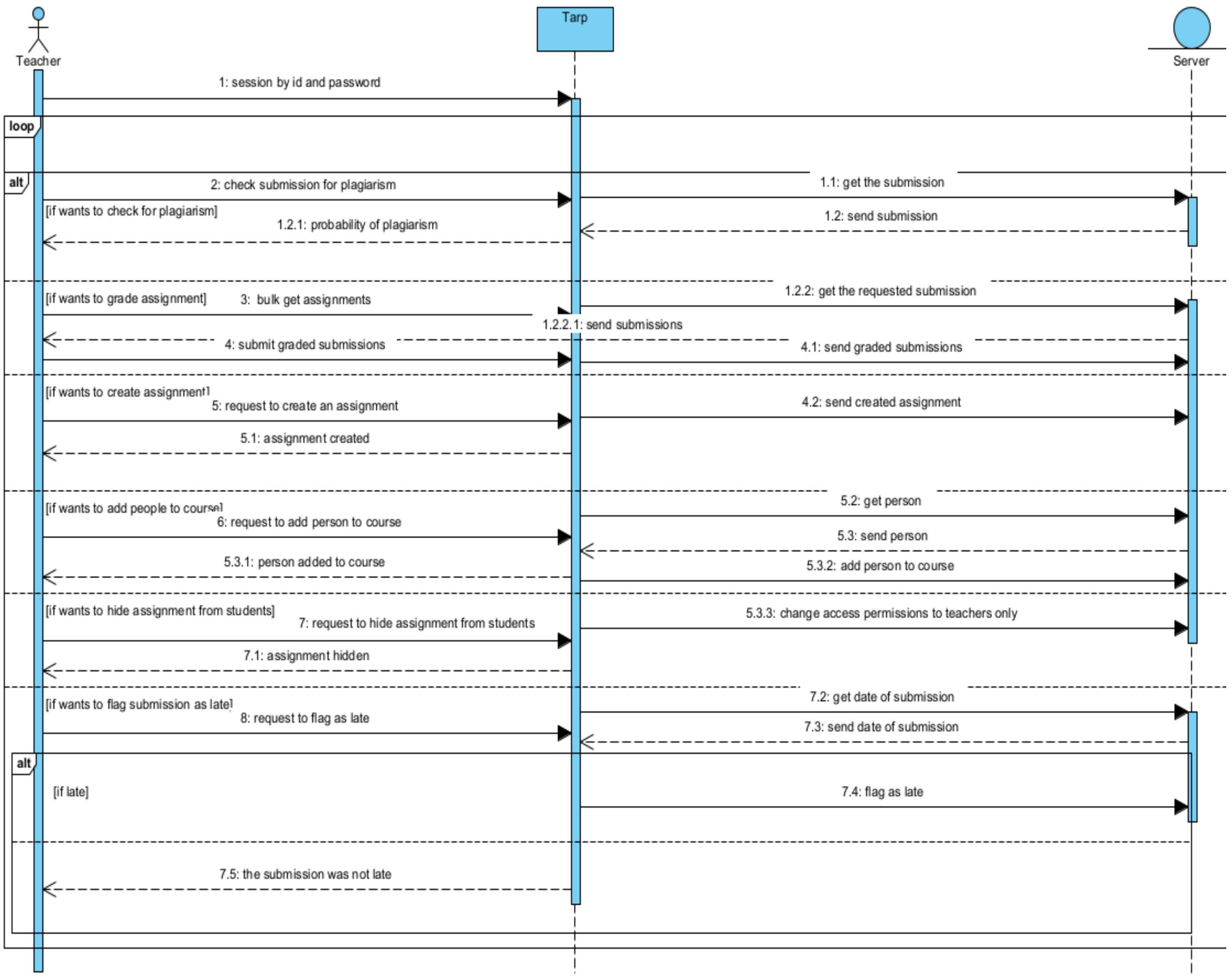
Description:
The activity diagram “Creating the assignment” represents a series of actions that are made by a teacher, other teachers and TA and how each action influences another. “Creating the assignment” particularly states what steps have to be taken in order to create an assignment for students. It covers possibilities such like adding textual description of the assignment, attaching additional, necessary files to it, setting a deadline and closed-for-submission date, limit the types of submitted files, hiding the assignment from students and sharing it with other teacher and TAs, who are able to check it, modify it and comment on it. Conditions on which certain enterprises can be made and how to act when the condition is not met are also explicitly stated.
The activity of “Creating the assignment” was chosen to be presented by activity diagram since it is the clearest way to show the amount of actions that can be taken and how to act under certain conditions.

Sequence Diagram 1(Student)



Description:
The student sequence diagram shows that the system in terms of the user has three lifelines: student, tarp and server life-lines. First the only actor, regarding the Student role should be able to login to Tarp learning management system using their credentials. The whole sequence diagram loops with the designated loop so that the management system is still showing to the student. After that several alt blocks are attached to represent the several options the Student have in the system, as well as their relevance actions if the command is selected (such as getting the submission from the server after the student requested access to see the submission, then sending the submission from the server to tarp and then tarp shows the submission to the student for example).

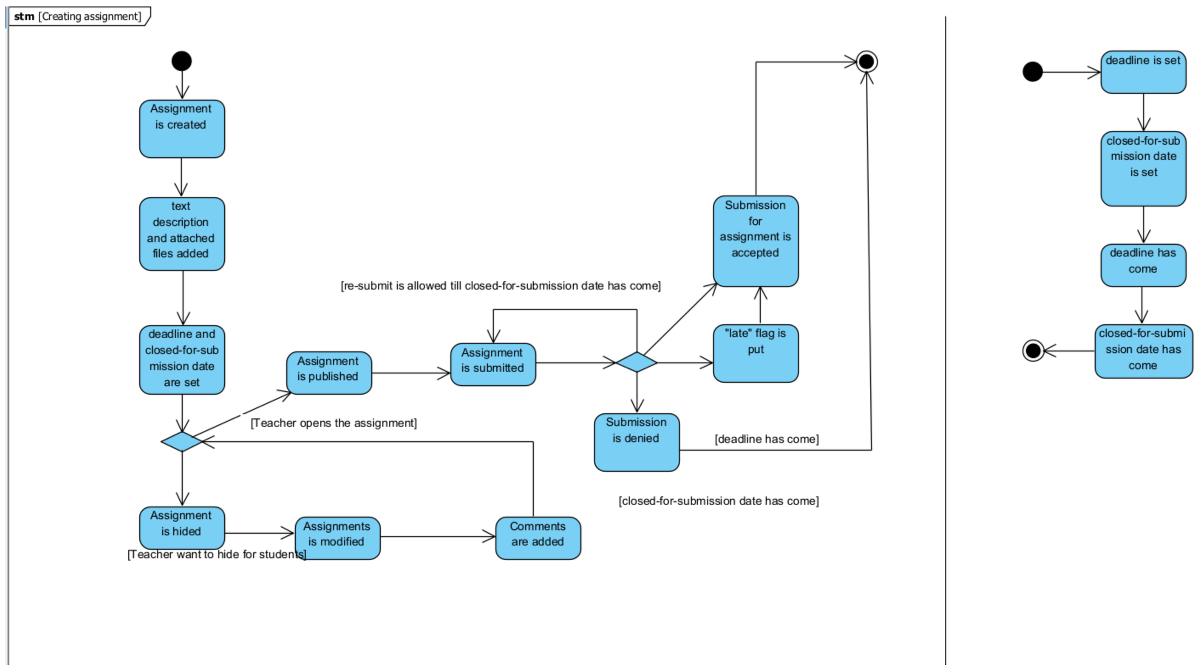
Sequence Diagram 2(Teacher)


Teacher
Tarp
Server
1: session by id and password
loop
alt
2: check submission for plagiarism
[if wants to check for plagiarism]
1.1: get the submission
1.2: send submission
1.2.1: probability of plagiarism
[if wants to grade assignment]
3: bulk get assignments
1.2.2: get the requested submission
1.2.2.1: send submissions
4: submit graded submissions
4.1: send graded submissions
[if wants to create assignment]
5: request to create an assignment
4.2: send created assignment
5.1: assignment created
[if wants to add people to course]
6: request to add person to course
5.2: get person
5.3: send person
5.3.1: person added to course
5.3.2: add person to course
[if wants to hide assignment from students]
7: request to hide assignment from students
5.3.3: change access permissions to teachers only
7.1: assignment hidden
[if wants to flag submission as late]
8: request to flag as late
7.2: get date of submission
7.3: send date of submission
alt
[if late]
7.4: flag as late
7.5: the submission was not late

Description:

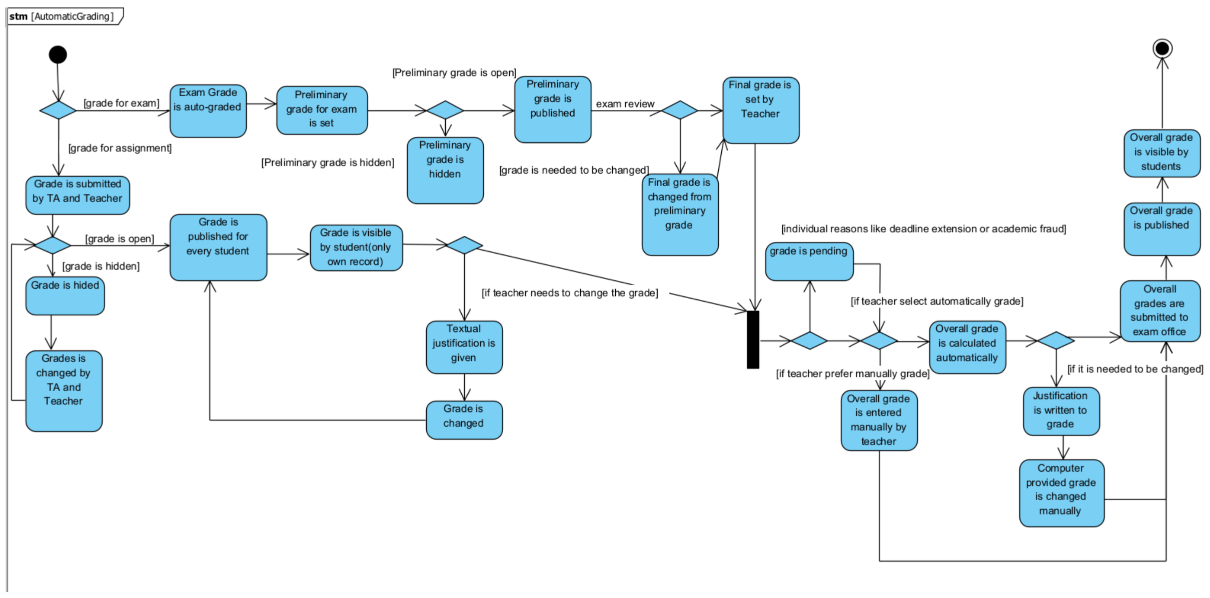
The Teacher sequence diagram shows that the system in terms of the user has three lifelines: teacher, tarp and server life-lines. First the only actor, regarding the Teacher role should be able to login to Tarp learning management system using their credentials. The whole sequence diagram loops with the designated loop so that the management system is all the time being shown to the student. After that several alt blocks are attached to represent the several options the Teacher have in the system, as well as their relevance actions if the command is selected (such as request to flag some assignment as late is made from teacher to tarp, then tarp gets the submission date and if the assignment has a late date then tarp will be able to mark is as late, otherwise the teacher will receive a message explaining that flagging the submission as late is impossible if the submission deadline does not consider it as late, thus, fair grading and fair late flagging.

State Machine Diagram 1(Creating Assignment)


stm [Creating assignment]
Assignment is created
text description and attached files added
deadline and closed-for-submission date are set
Assignment is published
[Teacher opens the assignment]
Assignment is submitted
[re-submit is allowed till closed-for-submission date has come]
Submission for assignment is accepted
"late" flag is put
Submission is denied
[deadline has come]
[closed-for-submission date has come]
Assignment is hided
[Teacher want to hide for students]
Assignments is modified
Comments are added
deadline is set
closed-for-submission date is set
deadline has come
closed-for-submission date has come

Description:
The state machine diagram describes the system of “Creating assignments”. The left side of the diagram shows the flow of which the assignment is created and submission for the assignment is accepted. The right side of the diagram shows the flow of the time which means “deadline” and “closed-for-submission date” for the assignment has come.
The main features of the system is that a teacher can hide the assignment till they want to reveal and the function which can recognize assignment is late for deadline and closed-for-submission date.
The reason why we chose the design which is composed of two different diagrams is the way to show time consuming is easily understandable for the client.

State Machine Diagram 2(Auto-Grading)


stm [AutomaticGrading]
[grade for exam]
Exam Grade is auto-graded
Preliminary grade for exam is set
[Preliminary grade is open]
Preliminary grade is published
[Preliminary grade is hidden]
Preliminary grade is hidden
exam review
Final grade is set by Teacher
[grade is needed to be changed]
Final grade is changed from preliminary grade
[grade for assignment]
Grade is submitted by TA and Teacher
[grade is open]
[grade is hidden]
Grade is hided
Grades is changed by TA and Teacher
Grade is published for every student
Grade is visible by student(only own record)
[if teacher needs to change the grade]
Textual justification is given
Grade is changed
[individual reasons like deadline extension or academic fraud]
grade is pending
[if teacher select automatically grade]
[if teacher prefer manually grade]
Overall grade is calculated automatically
Overall grade is entered manually by teacher
[if it is needed to be changed]
Justification is written to grade
Computer provided grade is changed manually
Overall grades are submitted to exam office
Overall grade is published
Overall grade is visible by students

Description:
The state machine diagram above explains how the overall grade is published and delivered to students. There are several decisions which means teachers can edit the grade before revealing it to students and also by accepting some possible complaints from students except overall grade is sent to the exam office.
We use the “Join” node, because all assignments grade and final grade of exam should be gathered before calculating the overall grade.

Testing & Metrics

Testing

From the functional and non-functional requirements that were specified in ___ section, the most crucial items from the requirements can be interpreted as: Assign users to the courses and possibly remove them too, Change user roles, creating assignments and possibility for making them non-available to students, Flag for suspicious assignments.
This set of tests was meticulously selected so a clear overview of what the system should return as a result can be formed after the test execution.

Overview:

Test Type	Test Description	Test Step (cyclomatic complexity)	Expected Results	Verdicts
Assignment Creation Test	Tests whether the assignment was successfully created	3	A new assignment created in Tarp	The assignment was successfully created
Assign users to course Test	Tests whether the respective users (TA’s and Teachers) have the possibility to assign users to a course	4	A new User had been added to the respective course	The user had been successfully assigned to the selected course
Remove users to course Test	Tests whether the respective users (TA’s and Teachers) have the possibility to remove users from a course	1	A new User had been removed from the respective course	The user has has been successfully removed from the course

Change user roles Test	Tests whether the respective users (TA's and Teachers) have the possibility to change the roles of a user	5	The regarded user have different roles now	The selected user roles had been updated successfully
Change user visibility of an assignment Test	Tests whether the user which wants to change the permission has higher significance in the hierarchy, and if so that the other user has a different visibility of the assignment.	3	The regarded user has now either gained access to an assignment. Or the regarded user has now lost access to an assignment.	Assignment visibility for the selected users had successfully updated
Suspicious Assignment Test	Tests if the regarded assignment is correctly flagged as Suspicious by the Plagiarism Checker. This is particularly useful for Teachers and TA's when grading.	4	Either the test was fraudulent and was flagged accordingly, or it wasn't fraudulent and was flagged accordingly.	Returns the plagiarism check of the selected assignment (either returning that is a clear submission or a fraudulent one)

Metrics

In relation to the learning management system, the two more relevant metrics to evaluate the effectiveness of the software as a correct solution to the original problem can be answered by the use of different metrics, among them: Defect Density and Cohesion.

Defect Density

Refers to the software testing technique used to find the total number of defects present in a system before its release. The relevant stakeholders or People that will use TARP in the future should act as feedback so the number of defects can be correctly counted. Afterwards, the number of defects are used to calculate the defect density in the system, by dividing the defects with the number of lines the system has.

Cohesion

Refers to up to what amount two modules belong together, meaning that the connection between two modules should be as minimal as possible, according to the design pattern.
The system should aim for high levels of cohesion and loose coupling levels.
Coupling regarding the level of interdependence between modules, this is particularly useful since a good design pattern is to keep the relation or connection between two modules as minimal as possible, and this is possible with the low levels of interdependence between modules, thus low coupling. Regarding high cohesion levels, this is only possible when the different parts within a module are highly connected between themselves, thus usually high cohesion leads to low coupling.

Conventions

During the development of the system, the following conventions will be followed.

Code Conventions

Conventions that will be used during the development of the system.

Idioms

During the development phase of the system, pythonic expressions will be used when possible. We believe this will improve readability of the code to other developers that may want to read it.

Formatting conventions

During the development phase of the system, braces for every block of code will follow the way java developers do it. We believe this will improve the readability of the code by a java experienced audience.

Code Snippets

During the development phase of the system, code snippets will be used throughout all development made by developers. We believe this will improve the effectiveness of our team, this will also prevent delaying the end of the development phase.

Naming conventions

During the development phase of the system,camelcase format will be used by the developers. This will follow java standard naming conventions.
For e.g. camelcase will result in CamelCase with the format.

Other Conventions

Conventions that will be used after the development of the system.

Naming conventions

The upload of a lecture video made by a Teacher or TA, will follow the same naming convention used in the canvas learning management system. Based on the acronym.
For e.g. Lecture 6 Topic 4 will result in L6T4 with the format.

Appendix

everymember's contribution

Mateusz Bartnicki

- Use Case Diagrams
- Sequence Diagrams
- Test Plan
- Test Policy
- Conducted the interview
- A list of functional and non-functional requirements

Oliwier Szwalek

- Prepared the questions for the interview
- Prepared the initial user stories.
- Activity Diagrams
- Description of activities diagrams
- Initial construction of the report.
- Conducted the interview.
- Cared for the integrity of divided tasks.

Shun Nishijima

- Introduction of report (Summary)
- Interview report(Summarize interview and get insight which can change our development)
- Whole construction of report
- Completed both state machine diagrams
- Both extended user stories and object diagrams
- user stories

Renato Chavez

- Description of both sequence diagrams
- A complete class diagram of the system
- Description of class diagram
- Testing
- Metrics
- Code conventions
- Helped during use case diagram creation

- Helped during sequence diagram creation