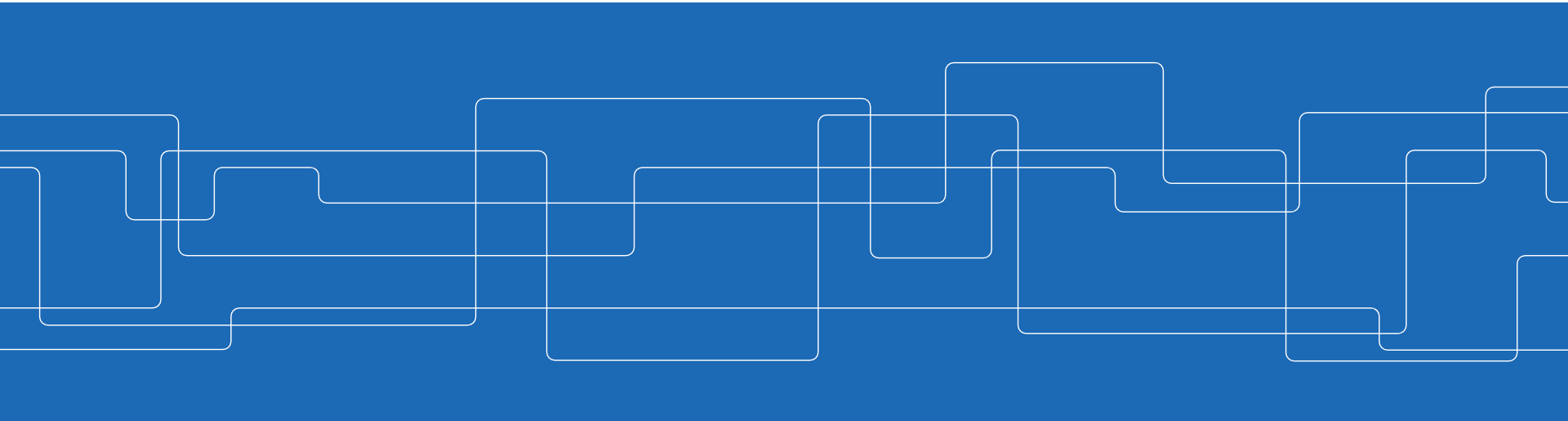
ID2207 - Modern Methods in Software Engineering

Final Project



Project Description

- Applying a subset of **Extreme programming approach (XP)** practices:

1. Developing user stories.
2. Release planning.
3. Iteration planning.
4. Selecting a system metaphor that could be suitable for the problem.
5. Developing a system in a test-driven fashion for selected user stories.
6. Refactoring the programs.
7. Pair programming.
8. Daily Stand-up meeting.

Project Description

- We will use **SEP business**, but we will follow the XP approach, not "Object Oriented analysis and Design"

- Group of 2 students
- We will focus on the following functionalities in SEP:

- ❑ Workflow of event requests (event initiation and application)
- ❑ Client data management (client registration and events history)
- ❑ Workflow of tasks distribution to services/production departments. (choose one sub-team from each department)
- ❑ Staff recruitment management.
- ❑ Financial requests management.

The employees' records, scheduling issues, salaries and other parts of the problem are not required for the project.

User Stories

- Simpler than use cases
- Make an estimation of the time it takes to implement each user story.

1. Login story

Login
"Any employee in the insurance company can access the system through the login screen where she/he enters her/his user name and password. After verification and based on the logged in user's authorization level, she/he will be able to access different functionalities".
Time Estimate: GUI: 0.5 hour, Logic: 2 hours

User Stories

Example of a user story that include many details:

1. Manage customer record story

…..

❑ View Customer personal record details

Split from: “Manage customer record story”

❑ View customer events history

Split from: “Manage customer record story”

……

User Stories

Another example:

3. Search criteria story
   Split from “Search for claimant” story

Search criteria
“A CD department employee can search for a claimant by name or email. If no claimant found, she/he should see a meaningful message”.
Time estimate: GUI (already in base story), logic: (included in base story, since this is a details split story)

Release Planning

- A release is a version of a system that is stable enough and has enough new features to be delivered to end users.
- Select a set of user stories for the first release, roughly 60%.
- The chosen user stories for the first release should include the required SEP functionalities that were highlighted in a previous slide.

Release Planning

- Classification of user stories by value and risk

Categories defined for value (Wikipedia):

a. High Value -> system cannot function without this story.

b. Medium Value -> non critical but has a significant business value

c. Low Value -> Nice to have

And categories defined for risk (Wikipedia):

a.High Risk -> either complex or no enough details or high chance of changes

b. Medium Risk -> standard complexity, medium change of frequent changes, maybe incomplete details

c. Low Risk -> simple, not likely to change, full details available

Release Planning

User Story Name	Value	Risk
…….	High	Low
……..	High	Medium
…….	…….	……

Release Planning

Number of stories for each combination:

	High Value	Medium Value	Low Value
High Risk	11	5	2
Medium Risk	5	6	3
Low Risk	1	1	3

You can Select according to the sorting combinations of value and risk. For example, you can focus on the combinations of (high, high), (high, medium), (medium, high) and (medium, medium).

Iteration Planning

- Divide the development of the first release into at least 3 iterations.
- Select stories to be implemented in each of these iterations.
- Adjust the iteration duration to your current time constraints.
- Whatever you have developed should be seen as integrated set of user stories which demonstrates full/partial functionality of your system (not islands of unrelated/far-away user-stories).
- It is important to finish all the user stories in each iteration

Metaphor

Example for a metaphor:
Insurance company

A court where the judge considers accused cases and takes a decision based on prove innocence documents provided by defense representatives.

Metaphor

1. Mapping between metaphor elements and actual system elements

Metaphor	System
Court	The insurance company
Court Judge	Employee Rank B, who will either approve or reject the insurance claim.
Accused	Claimant who is a customer in the insurance company.
Case	The innocence request from the crime.
Crime	The accident that triggered the case.
Court Decision	The decision whether to continue in the payment order and fix the car or to reject the claim. If it is approved, then we will have the payment order which can be mapped to “Innocence Consequence”
Prove Innocence documents Provided by the defense representative	The insurance team and garage feedback. The defense representative is the employee who contacts the garage and the insurance team in order to add details to the claim that might help in the approval.

Perform the planned iterations

- You are free to choose your programming language.
- Use pair programming.
- Do at least 3 refactoring in each iteration: generalized classes, modified attributes, fixing data structures, missing relationships….
- Have a daily meeting to discuss the project progress

stand up meeting

Daily meeting (usually in the morning as the beginning of the working day) with all group members discussing the project goals, issues and progress." During a stand up meeting developers report at least three things; what was accomplished yesterday, what will be attempted today, and what problems are causing delays.

Stand up meeting report

Example of a meeting report – You are free to choose another template

Meeting Date	
Participants	
Meeting Notes	1. **Summary of our activities in the previous day:** a. Sorting and selection of user stories for each iteration was successfully done. 2. **Today expected actions:** a. Writing the Metaphor! b. Thinking of the metaphor effect on the classes' names. 3. **Problems:** b. Till the moment, we don't have major issues.
Comments	Expected outcome for today: Filling the metaphor section and initial idea about classes naming *Written by*:

Implementation Details

- It is ok if data is stored in a file or memory and disappears when the program is shut down.
- You should have a user interface to be used by different employees of SEP company. This means that you need to have a kind of authentication and authorization mechanism to allow and control user access to different parts of the system.
- You don't have to implement "notifications" in the current project.
- All the required system functionalities should be automated.
- In the system, you have to keep record of all customer interactions with the company.

Test Cases

- Use test-driven programming. First write a test, then the implementation necessary to compile and run the test.
- You only have to write tests for the model, not for the user interface.
- You are not allowed to use any tools to write the tests and perform unit testing on behalf of you.

Acceptance Test

- an acceptance test could consist of a script of user interface actions and expected results that a human can run.

1. Successful Login

Test Case Name	Login
Expected Actions	1. Navigate to the login page of the system 2. Enter user name: “employeeA” 3. Enter password: “p@ssw0rd” 4. Click on “Login” button
Expected Results	The system should redirect the user to the tasks list page, given that the provided credentials are valid.
Test Result	Successful

Expected Deliverables

- A set of developed user stories with time estimates
- A set of selected stories for first release. Explain your selection by considering importance and risk factors.
- Iteration plan that is a list of which stories you implement in which iteration.
- The metaphor.
- Description of your test-driven pair programming process and applied refactoring. Also describe how well you managed to estimate what should be done in each iteration.

Expected Deliverables

- The source code (including source code for Test Cases), and a readme.txt file that explains what is needed to compile and run the program.
- Write acceptance tests for two of the user stories you implement. You must choose stories that take input and give output through the user interface. The acceptance tests shall be fully automatic, that is they shall give input to the user interface and evaluate the results displayed by the user interface.
- Report of daily stand-up meetings!
- Comparison between this approach and the object oriented analysis and design approach – your feedback

You need to present your project !