ID2207 -Modern Methods in Software Engineering

Final Project

Course Coordinator:
Mihhail Matskin

Course Assistants:
Shatha Jaradat & Cosar Ghandeharioon

2017

Project Description

In this project you are supposed to apply several elements of the **Extreme Programming approach (XP)** to solve the problem of implementing the SEP business described in previous assignments. Only a subset of XP practices/elements will be used, because of the difficulties with complete modeling an XP process in the framework of a small course project. For more details about XP we refer to the lecture notes and to: http://www.extremeprogramming.org/start.html.

We will focus on the following functionalities in **SEP** business:

1. Workflow of event requests.
2. Workflow of tasks distribution to services/production departments.
3. Staff recruitment management (through HR)
4. Financial requests management.

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

Remember that you must solve this problem in groups of **two** students. We will not accept any project done by only one student. XP practices depend on this to be meaningful.

Note: We are not doing Object Oriented Analysis and Design as we did in previous homework in this course. We will follow XP approach.

The main elements of XP in this project will be:

1. Developing user stories.
2. Release planning.
3. Iteration planning.
4. Selecting a system metaphor that could be suitable for the problem solving.
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 Task

1. Develop a set of user stories (remember that they are not use cases and they are much simpler). You should write all user stories for your system (excluding User Interface user stories). The stories should be meaningful and nontrivial, otherwise they will not be counted!
2. Make an estimation of the time it takes to implement each user story.
3. Select a set of user stories for the first release, roughly 60%. A release is a version of a system that is stable enough and has enough new features to be delivered to end users. The chosen user stories for the first release should include main functionality of your system.
4. 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 by starting from simpler user stories. BUT at the end 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).

5. Write a metaphor.
6. Perform the planned iterations.

- It is very important that you actually finish each iteration. When an iteration is finished there must be no “almost done” stories. The stories of the iteration must be written and successfully tested.
- 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. You SHOULD write the test cases by yourself. You are free to choose your programming language. If there would be a piece of code in your deliverables project code which does not have corresponding and appropriate test-case, we conclude that you have not followed test-driven programming approach!
- Do at least 3 refactoring in each iteration.
- Use pair programming.
- Have daily meeting to discuss the project progress

- It roughly takes 2 full days, for two skilled programmers to develop the system

In general the system should handle the following:

1. It is ok if data is stored in a file or memory and disappears when the program is shut down.
2. 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.
3. You don’t have to implement “notifications” in the current project.
4. All the required system functionalities should be automated.
5. In the system, you have to keep record of all customer interactions with the company.
6. There could be number of other things which could be done, but we limit our problem to this description only...’

Deliverables

You should deliver:

1. A set of developed stories with time estimates.
2. A set of selected stories for the first release. Explain your selection by considering importance and risk factors.
3. Iteration plan that is a list of which stories you implement in which iteration.
4. The metaphor.
5. 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. Write the truth! In order to pass you must show that you can draw conclusions of your mistakes, not that you have done everything perfectly.
6. 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.
7. 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.

Report this task by submitting a short description of the acceptance tests (including how they are started). The chosen acceptance test should address the main functionality of your system.

8. Report of daily stand-up meetings! (2 or 3 is okay)
9. A comparison between this approach and the object oriented analysis and design approach that you followed in the previous three assignments (your feedback).

•You need to present your project. Presentation date will be announced later!

References

1. For a list of XP elements see for example Beck, Kent: Extreme Programming Explained: Embrace Change, 2nd Ed (ISBN: 9780321278654) (http://www.mip.sdu.dk/~brianj/Extreme%20Programming%20Explained%20-%20Kent%20Beck;%20Addison-Wesley,%201999.pdf)

2. http://www.extremeprogramming.org/rules.html

Appendix

1-What is a stand up meeting? (http://www.extremeprogramming.org/rules/standupmeeting.html)

You are having 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. The daily standup meeting is not another meeting to waste people's time. It will replace many other meetings giving a net savings several times its own length".

2 -Example User Story: Login

At Startup & When the system starts, the user interface will present a login screen.

The user can now log in using its username and password. After verification the user is presented with the set of actions it can perform.

Time estimate : GUI 0.5 hour, User Account: 0.5 hour

3What is Acceptance Test?

"At the very least, an acceptance test could consist of a script of user interface actions and expected results that a human can run. Acceptance tests can be automated, either using the unit testing framework, or a separate acceptance testing framework." [John Brewer]. For those who are coding in Java, this tool could be helpful: http://exactor.sourceforge.net/