C31310 - Agile Methodologies
Worksheet: Project Process

Diana Silvia Teodorescu, dst1@aber.ac.uk

1. Outline of the project

My final year project will be an analysis of the data collection provided from the historical newspapers found at the National Library. The newspapers are digitised and have a basic API that can be used to search, request and modify the data. The point of the project is to discover interesting themes and concepts within the articles and using data mining processes and statistics to present them in an interesting way to the user.

I am considering a few options regarding the data processing and it can be made available either through a mobile application or a web application. In both cases I will look into connecting the data resulted through analysis in the newspapers with external data to attract interest into the past and the changes that occurred over the years.

2. Process description and methodology choice

For my project methodology I was thinking of using an agile methodology due to the flexibility and freedom that it offers. Feature driven development can provide the amount of freedom needed when the entire requirements are unknown from the beginning while still offering a structure to the development. I am considering using evolutionary design and some of the XP practices such as constant refactoring and simple design to make sure that my code is easy to understand in case changes need to be done further ahead. With YAGNI it's also important to remember to refractor after each change which will also help with the verification part of the FDD steps.

Taking into account the FDD best practices there will be an initial object model developed that will be assessed with my supervisor. This will need to take into consideration the risks involved and make sure the expectations are accord with the time and resources available. After an overall model has been decided I will be able to split the tasks into features and arrange them in priority order. Having specific deadlines depending on iteration size and also asking for small releases would make my work easier and would encourage me.
After the feature list is planned due to a lack of team members with different roles a careful inspection of all the activities and the way they are broken down would be very important.

The feature development steps will have to be moulded on what resources and roles are available in this case. Following a plan would help me keep focus and I believe that keeping track of my progress through coloured charts is useful. This will be good motivation especially when the green(features completed) and yellow (features in progress) would start to take over the other colours.
Also providing regular builds will help verifying the system and also spot any integration errors early giving a better chance to fix them.

I have picked FDD over XP due to the fact that a customer is not as important in the planning and development stage as it is in Extreme programming. In my case the customer can either be considered the National library or the supervisor. The National Library holds the entire data and can provide information regarding the API and the other resources available. They might be interested in the end product but they won't be involved the project progress and don't have specific requirements regarding it execution except for their need to represent the data in an exciting way that would attract people to use it.

After the code has been developed, the class owner creates unit tests for the class they own and do a code inspection throughout all of the code in the feature. The order of unit testing and code inspections does not matter, but both need to be done before the code is added to the project code.
The problem with choosing feature driven development as with other agile methods is the need of a team. The fact that it provides individual code ownership is a plus over XP in this case as I will be the only person that owns the code. Regarding progress reporting a way to go around this is to constantly test my code and refractor it and request assessments on the weekly meetings with my supervisor.

I do like the XP refactoring principle and I agree that the code structure need to be checked and modified regularly and that this will remove the issue that comes up with ad-hoc design decisions. The inspection of the code will be done by me and it should be an important part. Using rubber duck debugging and unit tests could also prove valuable.

3. Choice justification

My idea is to start with basic requirements for a simple product but build more features into it along the way to make it nicer and offer more functionality to the user. As much as I appreciate the idea of no design planning and more customer input that XP offers I believe in my case some structure would be welcomed.

Because of the nature of my project and the uncertainty of the data analysis results I am certain that being restricted to a specific design that is set in stone from the beginning would be a wrong choice and that changes should be welcomed.

I think that regular tests, feature planning and simple design are the points that could definitely improve my work. Also providing a method of observing how my work is going and what is left to be done can support me and encourage me to be methodical until the end.