Masters of Computer and Information Sciences

Software Development Methods 409232
Semester 2, 2015

ASSIGNMENTS 2
Development Process Project
Contribution to final marks: 60%
DRAFT 1

Based on the discussions in class, the following decisions about the assignment are documented

Due dates:
Final product available via the cloud to Client before Wed October 21st 11:59pm.
Team Evidence Portfolio wiki and other evidence closed off on Wed Oct 21st 11:59pm
Team sprint reviews every two weeks. Times and dates to be negotiated in weeks 8, 10 and 12.
Team presentation in the lecture in week 12.

Team Effort
- This is a team assignment to be managed and submitted as a Team of 4-6 people.
- All Team members will receive the same mark for this assessment unless it is indicated in the Team Participation Form that contribution or participation was not equal.
- The lecturer may also negotiate an unequal assignment of marks under certain circumstances.
- Please upload to the team wiki a completed AUT team participation assignment cover sheet. THIS SHOULD BE SIGNED BY ALL TEAM MEMBERS. (100% is the benchmark. Lower than 100% means this team member did not participate equally in the group)

Submission Requirements:
The following items will contribute to your final grade for this assessment.

- Electronic Evidence Portfolio. The methods used need to be evidenced for evaluation and this contributes to your final grade. All the evidence files (videos, audio, pictures, documents) should be uploaded to the team's wiki on a regular (daily) basis. The evidence as media files should be uploaded to the team wiki in a week-by-week structure and labelled according to the evidence names in Table 1 The wikis will be locked off before the lecture on Week 12
- Product (SERLER). A running and quality assured product should be available in the cloud by the week 12 deadline via a Heroku link. Any instructions for running the application should be available on the wiki (e.g logins, data etc).
- Use Trello (trello.com) to manage the process of collaboratively managing, developing and implementing your application. Make sure you invite your lecturer to be part of your Trello board using the SPECIAL EMAIL**: autsdm2015@gmail.com.** Trello activity log will be used to check usage.
- Use GitHub (Github.com) for code sharing, version control and code integration Make sure you invite your lecturer to be part of your Github repository using the SPECIAL EMAIL: autsdm2015@gmail.com. The GitHub activity log will be used to check usage and evaluate code.
- In week 12 lecture, a 10 minute team presentation reflecting on your experiences with the organisational and technical SE methods used in this assignment compared and contrasted with academic literature on SE practice. An informative handout for the audience should be included.
- Sprint reviews for process and progression (as well as product). This will involve your lecturer reviewing your methods using a checklist of the evidence items expected, usage of Trello and usage of GitHub. The PO will also review the features committed to for each sprint.
- Use APA 6th reference formatting for citing others' published materials. (APA_6_AUT in Endnote)

Purpose of Assignment
This assignment relates directly to the following course learning outcomes:

3. Critically assess, compare and contrast the distinguishing features of a variety of software development approaches and methods
4. Recommend and justify the selection of development approaches, methods and practices across the full range of development activities for different development contexts
5. Use a selection of industry standard models, tools and techniques that support development methods
6. Critique research literature in the area of software development methods
8. Apply some software development methods and critically reflect on the experience.

Aims

The aim of this assignment is to develop a software application as a vehicle to practice and reflect on methods and techniques related to team-based software development across the full development lifecycle. This problem-based approach motivates some of the learning expected in the course as well as applying some learned methods and tools. This includes project management and organisational methods as well as technical methods.

In summary the aims are:

- To gain experience in the full development lifecycle of software development from idea to implementation using Agile practices.
- To gain an overview of the common practices and tools used in commercial Agile software development.
- To understand some of the benefits and challenges of commercial Agile software development
- To contribute to a research project for the Software Engineering Research Lab (SERL) at AUT

Note that you are assessed more on evidence of progress and learning related to the (eventual) well-executed application of methods rather than the final software product.

You should communicate frequently with your lecturer (assessor of the process and evidence) to understand the requirements of the evidence portfolio and process, as well as the Product Owner (assessor of your delivered product) to understand the requirements of the product.

Some constraints on the process and tools are imposed to simplify the experience and assessment complexity. The learning should focus on the principles associated with these methods and tools that are transferrable. For example, incremental, iterative development is well-evidenced good practice and can be implemented in a number of ways. We are using methods from Scrum as a framework to manage this.

Expected Development Methods and Tools
All development should involve the use of the following methods and supporting tools. Evidence of use of these methods and tools should be included in the Evidence Portfolio as described later.

TECHNICAL AGILE METHODS

- Ruby on Rails as a programming language. No IDE specified. (Text editor and console is fine). Other libraries such as Bootstrap and Font Awesome can be used. Note that the Rottenpotatoes exercises use a slightly older version of Rails and other libraries. You can the newer version if you want.
- Any operating system for the development/test/production environments is fine (Windows, OSX, Linux).
- Version control, code sharing and code integration using GitHub (cloud-based)
- Test driven Design using RSpec
- Behaviour Driven Development using Cucumber and Capybara
- Continuous Integration using Travis CI or Heroku's CI service (cloud-based)
- Bug management – not specified. You have a choice, but it must be cloud-based and available to your lecturer for evaluation.
- Deployment to the cloud using Heroku (cloud-based)

ORGANISATIONAL AGILE METHODS

- Project management and coordination for the development using Trello (cloud-based). Trello should include the following information:
    - project shared information (e.g. definition of Done, contact details)
    - A (changing) product backlog
    - A sprint backlog (which will change sprint number at the start of each sprint)
    - workflow columns including *at least* "Develop, Test, Done"
    - User stories should have a different colour for each sprint so they can be distinguished in the Done column.
    - **User stories** should include information: the story title, the story, a priority number, an effort estimate in story points, acceptance tests (including pass/fail & dates), explanation notes. This information should align with information used by Cucumber.
    - **I recommend having an issues/risks column**
    - **Bug cards may be incorporated into the workflow as tasks– they should be distinguishable from user stories. Should they be included in the sprint velocity?**
- I recommend that Trello is also used to manage the tasks FOR THE PORTFOLIO and OTHER EVIDENCE related to this assignment. This may be a separate team Trello board
- Progress monitoring of development using a Burndown chart - ONE PER SPRINT - should be available on the cloud and updated daily as necessary. This may be a file (e.g. spreadsheet or photo) uploaded to the Wiki, a plugin for Trello, a file uploaded to Trello, or some other tool. It may be photos of a hand-drawn Burndown chart or a spreadsheet or some other tool. THIS SHOULD BE UPDATED CLOSE TO EVERY DAY BY INFORMATION (HOW MANY STORY POINT LEFT TO DO) FROM EACH PERSON WORKING ON A USER STORY
- You will use (and experiment with) practices from Scrum and XP for this development project. FOCUS ON TRYING TO DO THESE WELL/CORRECTLY EVEN IF IT TAKES A BIT LONGER.
- This includes:
    - Face-to-face stand-up scrum meetings (*at least* one per week)
        - DAILY Electronic scrum meetings when not face-to-face (still answer 3 questions)
    - Face-to-face Sprint planning meetings (at least one per sprint)
    - Face-to-face estimation meetings using planning poker and user story points (at least one per sprint – may be during sprint planning meeting)
    - Face-to-face Sprint review (show case) meetings (1 per sprint with PO)
    - Face-to-face sprint retrospective meetings (1 for sprint 1 and 3, with a list of actionable items as an output). FOR SPRINT 2, use Padlet.com to experiment with an electronic retrospective and compare the results with the face-to-face meetings.
    - Experiment with Pair programming (at least once per sprint). Compare this to individual.

This is summarised in Table 1. **Collecting and uploading evidence of using of these methods and tools should be planned and managed carefully. Trello could be used to manage this also. Make sure it is simple for your lecturer to find these evidences week-by-week and sprint-by-sprint.**

Organisational Practice	**Evidence required**	**Minimum frequency**
Product Backlog management	Trello Activity log Lecturer Reviews	Continuous/daily
Sprint Backlog Planning	Video/audio	1 per sprint
SCRUM standup meetings (answer 3 questions only)	Video/audio – challenges board	2 per sprint (1 per week)
Sprint Reviews	Video/audio - minutes	1 per sprint
Spring Retrospective meetings	Video/audio – list of actions	1 per sprint
Pair programming	Video/audio	
Burndown charts	Document	Continuous (daily)
Planning poker for estimation of story points	Video	1 per sprint
Technical Practice	**Evidence required**	**Minimum frequency**
Mock-up / lo-fi protoype	Document/mockup	One per sprint
Version control, code sharing and code integration using GitHub	Lecturer access to GitHub repository for every team Demonstration in sprint 2 **Screen shots**	Continuous (Examples 2 per week)
Test driven Design using RSpec	Lecturer access to code Demonstration/explanation sprint 2 **Screen shots**	Continuous (Examples 2 per week)
Behaviour Driven Development using Cucumber and Capybara	Scenarios in github Demo sprint 2 **Screen shots**	Continuous (Examples 2 per week)
Continuous Integration (using Travis CI or Heroku's CI service	Demo sprint 2 **Screen shots**	Continuous (Examples 2 per week)
Deployment on the cloud using Heroku	Demonstration at reviews	Sprint reviews (1 per sprint)
Bug management (tool?)	Screen shots	Continuous
Communications with each other	Video/audio/screen shots	Continuous (selection of important examples)
Communications with product owner	Video/audio/screen shots	Continuous (selection of important examples)

Table 1 Evidence of Process and Methods expected in portfolio

System Design Constraints

Ruby on Rails imposes a Model-View-Controller design pattern for the architectural structure at the code level. This is good practice (although not the only approach) to separate these concerns for future code maintainability and extension.

The system should be delivered over the cloud, using a client-server architecture based on http protocols over TCP/IP. We are using Heroku's servers for the cloud service and a local browser as the client.

A relational database should be used for storing and manipulating persistent data. PostgresSQL is recommended but Ruby on Rails and Heroku can support others.

Portfolio of Evidence
The evidence as media files should be uploaded to the team wiki using a sprint-by-sprint, week-by-week structure and labelled according to the evidence names in Table 1. Example structure:

Sprint 1
Week 1
Relevant Evidence items in Table 1
Week 2
Relevant Evidence items in Table 1

Sprint 2
Week 1
Relevant Evidence items in Table 1
Week 2
Relevant Evidence items in Table 1

Sprint3
Week 1
Relevant Evidence items in Table 1
Week 2
Relevant Evidence items in Table 1

Media files should be of reasonable quality and of a common format. For example:

Video - .mp4
Audio - .mp3
Picture - .jpeg
Document - .pdf

The evidence should be uploaded to the team wiki on a regular (daily?) basis.

If media files are too large to upload to the wiki, you can create a shareable link to the file on Google Drive or OneDrive (both have 15GB free storage) and upload this link to the wiki.

Sprint Reviews
You can expect to have both the lecturer and Product Owner present at a review at the end of each sprint (dates and times to be negotiated by each team).

The lecturer will want to see evidence of process, methods and tools as well as software progress (e.g. explanations/evidence of use of Trello, Github, Continuous Integration, TDD, BDD, retrospective outcomes).

The PO will want to see the features for the product that the team committed to for that sprint.

The project

The project is the Software Engineering Research Lab Evidence Repository (SERLER), as described by the Product Owner.

While it is important to deliver working software at the end of each sprint, the focus of this course is process – the application of organisational and technical methods for developing software. Therefore, the main emphasis on your final grade will be the evidence of process and progress (70%) rather than the product (30%).

The Product Owner will be available to discuss the product requirements from time to time and is available online at autsdm2015@gmail.com.

Three requirements documents have been prepared by the PO so far (available on Blackboard):

SERLER DRAFT Workflow Overview
Information on the SERLER repository
SERLER DRAFT Data Structure

The teams
The class is divided into two streams that will be producing their own version of the product.

Each stream will have three development teams that are responsible for different modules (feature sets) of the product.

Each team may have members who are accountable in development, analysis or testing. All members should be learning others' skills to some extent though. These roles can be rotated as needed. One or more person may take on the scrum master role.

This scrum master is a service leadership role that should ensure that the team understands the process agreed on and it adheres to Agile principles; ensures blockages to progress are identified and the team addresses the blocks; may facilitate meetings. Also protects the team from demands of management and over-zealous POs!

Feature Allocation

The following tables show which teams are allocated to which development effort. More detail is given in the user stories provided by the PO.

STREAM 1

Team	**Module**
Alpha	Search and display
Beta	Information Entry
Gamma	Admin and workflow

STREAM 2

Team	**Module**
Delta	Search and display
Echo	Admin and workflow
Foxtrot	Information Entry

DRAFT USER STORIES BY THE PO. NEEDS FURTHER REFINEMENT

Search and Display Module
As a registered user of SERLER I want to be able to search for empirical evidence on different methods based on different constraints so that I can make informed decisions. NOTE: Things like the SE method, the research method, a date range, an author, the credibility rating, the benefit being tested. Any combination of these should be possible for a search as inclusion an/or exclusion criteria.

As a registered user I want to be able to view the results of a search in a tabular format so that I can view a lot of information easily.

As a registered user I want to be able to easily sort the displayed results of a search by any field I want so that I can experiment with what is useful

As a registered user I want to be able to select what columns are visible in a displayed search so that I can experiment with what is visible

As a registered user I want to be able to save any searches I make so that I can easily run them again later to save time

As a registered user I want to be able to browse the repository by information like date or SE method so that I can familiarize myself with the data

As a registered user I want to be able to share the results of my search by email or facebook so that I can inform collaborators

As a registered user I want to be able to save the results of my search so that I can manipulate them or view them later.

As a registered user I want to be able to print the results of my search so that I have a record of them

As a moderator or administrator I want to be able to see a list of all articles still to be moderated so that I know how much work is to be done

As a moderator or administrator I want to see a list of all articles that were previously rejected so I can check submitted articles were not previously rejected

As an analyst or administrator I want to be able to see a list of all articles that are ready for analysis so I know how much work is to be done.

As an analyst or administrator I want to list all partially analysed articles so I know which ones need to be finished

As an administrator I want to be able to list all the users with different roles so that I can check who is doing what roles

Information Entry Module
As a registered user I want to be able to submit an article to the repository so that the repository is kept up-to-date. (NOTE need some flexibility in this – as well as filling a form, could allow submission of a reference in a specific format such as APA, or a pdf, or a bibtex file. I am worried about copyright if a link or a pdf).

As a returning registered user I want to be able to logon to the system so that I have access to the features appropriate to my role.

As an analyst I want to be able to enter repository information quickly and easily, minimizing errors and effort so that I can add to the repository information quickly and correctly. (NOTE: some information should have pre-set values to choose from, for example)

As an analyst I want to be able to save a partially finished evidence item and easily return to it later, so that if

As an administrator I want be able to enter news and recent interesting/significant articles for users to be informed and add interest when they are on the SERLER website

As an analyst I want to be able to change the information on an evidence item before it is available to the general user.

As an administrator I want to be able to change, delete or add an evidence item no matter what its status (NOTE: need to keep an audit trail kept of these changes)

Admin and Workflow Module

As a new user I want to be able to register so that I don't have to enter my details again and the system knows what role I have. NOTE: I want to be able to register using my existing Facebook or Google accounts.

As a paper submitter I want to be informed if my submission was accepted or rejected and why, so that I learn what is suitable

As a user I want to be informed of new articles and interesting news when I logon so that I am kept up-to-date with new interesting material

As a moderator I want to be informed whenever a new paper is submitted for moderation so that I know to schedule it

As an analyst I want to be informed whenever a new paper is ready for analysis so that I know to schedule it

As an administrator I want new user registrations to be verified by email as a level of security (NOTE FROM PO – this isn't quite a user story but I can't think how to say it)

As an administrator I want to be able to restore information from an automatic backup so that the repository is safe

As an administrator I want to be able to change, delete or add users (NOTE: needs an audit trail kept of these)

As an administrator I want to be able to change the role registered users have so that they can get access to the features they need to to their role.

As an administrator I want to be able to change pre-set values available in the database so that the values are useful

As an administrator I want to be able to add, modify or delete information fields in SERLER

As a moderator I want to be able to easily accept or reject papers so that SERLER has only high quality articles. (NOTE: the criteria for acceptance should include – (1) has been peer reviewed (2) is not already in SERLER (3) is relevant to SERLER (ie empirical evidence about some aspect of a SE method)).

As a registered user I want to be able to give feedback on SERLER that includes bugs and desired new features and features I like so that I can increase the quality of future releases of SERLER

As an analyst I want to be able to find the pdf of an article that I have legal access to in an online library and download it and tag parts of the article so that I can do my or analysis. (NOTE it would be great if analyst could select a piece of text in a pdf, and select a tag (e.g. SE method) which triggers a copy and paste of this selected information into the right field (the SE methodology) in the database).

As a user I want to be able to logout of the system so that others can't use SERLER with my logon

WORKFLOW AND ADMIN TEAM IS ALSO RESPONSIBLE FOR

- Entering some actual information in SERLER. Find at least 5 relevant high quality articles, analyse them, and enter the data into SERLER.
- Creating at least one dummy user in each role (subscribed user, analyst, moderator, administrator)
- Creating the database for other teams to use
- Overall consistent User Interface and navigation, including a logo for SERLER

MARKING CRITERIA for Evidence Portfolio

There is clear evidence of:

- An honest effort to use Agile organizational and technical methods to develop the software as a team
- Reflection and learning from sprint to sprint from experimentation resulting in changes
- Regular communication for coordination of effort and collaboration
- Quality assurance through testing and reviews and fast feedback loops
- Planning and monitoring

The evidence portfolio includes media files for each week/sprint

Organisational Practice	**Evidence required**	**Signs of Health (A-grade)**
Product Backlog management	Trello Activity log Lecturer Reviews	Regular usage All team involved in using it Complete and accurate Used correctly Product Backlog changes as expected
Sprint Backlog Planning	Video/audio	Time boxed User stories have expected detail, including estimates Expected velocity is discussed Result is a set of features that the team is COMMITTED TO DELIVERING No stories are added during a sprint
SCRUM standup meetings	Video/audio – challenges board	Timeboxed Only three questions are answered
Sprint Reviews	Video/audio - minutes	To be discussed
Spring Retrospective meetings	Video/audio – list of actions	To be discussed
Pair programming	Video/audio	To be discussed
Burndown charts	Document	To be discussed
Planning poker for estimation of story points	Video	To be discussed
Technical Practice	**Evidence required**	
Mock-up / lo-fi protoype	Document/mockup	To be discussed
Version control, code sharing and code integration using GitHub	Lecturer access to GitHub repository for every team Demonstration in sprint 2 **Screen shots**	To be discussed
Test driven Design using RSpec	Lecturer access to code Demonstration/explanation sprint 2 **Screen shots**	To be discussed
Behaviour Driven Development using Cucumber and Capybara	Scenarios in github Demo sprint 2 **Screen shots**	To be discussed
Continuous Integration (using Travis CI or Heroku's CI service	Demo sprint 2 **Screen shots**	To be discussed
Deployment on the cloud using Heroku	Demonstration at reviews	To be discussed
Bug management (tool?)	Screen shots	To be discussed
Communications with each other	Video/audio/screen shots	To be discussed
Communications with PO	Video/audio/screen shots	To be discussed

MARKING CRITERIA for product

- There is incremental development
- It is fit for purpose, with expected features delivered (may be different to the initial expectations)
- It is robust (ie I can't break it in less than 1 minute)
- The user interface is cool and pretty and useful

MARKING CRITERIA for Team Presentation

- I really like it
- Others really like it
- You look like you enjoyed it
- It covers the material we agreed on
- It shows some maturity and depth of thinking