Context Project Guidelines 2013/14

1. Introduction

The goal of the **Context Project (CP)** is to develop, implement, validate, present, and demonstrate a software product that satisfies the needs of an external party (*users*) in a given non-ICT (societal, scientific, or business) context. To achieve this goal, students will work in teams and with an external party to understand the problem and the context in which the problem is defined, the needs of the external party regarding how to solve the problem, and the possibilities to realize a software product as an effective and innovative solution to this problem. Furthermore, students will practice their knowledge and skills in software engineering (including requirements engineering, architecture and software design) in a concrete practical case to realize the envisioned software product effectively and efficiently. This product will be implemented following an agile software engineering process. The project will be completed by a presentation and demonstration of the developed product.

2. Learning objectives

We distinguish among three categories of learning objectives: (1) Process, (2) Product, and (3) Presentation.

1. Process:
After completing the CP the student will be able to:

- Work in a team following an iterative and incremental software development process
- Work with users and stakeholders from a non-ICT context
- Analyze and evaluate a problem in a non-ICT context
- Acquire the information and knowledge on the given non-ICT context from literature and experts, in order to design and develop an effective and innovative solution
- Evaluate the requirements/user stories of different stakeholders
- Develop the system following an agile software development process
- Reflect on his/her own contribution to the project and the final product
- Form a vision of the role of IT in the given context and for solving the given problem

2. Product:
After completing the CP the student will be able to:

- Analyze and document the system requirements in a requirements document
- Design and document the architecture for the given problem in a non-IT context
- Specify an iterative and incremental test and implementation plan
- Develop a prototype and/or final product according to the requirements, design, implementation, and test plan
- Continuously test and evaluate the solution

3. Presentation:
After completing the CP the student will be able to:

- Report in oral and written form to his/her peers, supervisors, users, and customers
- Demonstrate the developed software product

3. Projects and coordinators

The overall coordinators of the CP are

- Alan Hanjalic (A.Hanjalic@tudelft.nl)
- Alberto Bacchelli (A.Bacchelli@tudelft.nl).

The CP consists of five concrete projects defined in different practical contexts. Each project has its own objectives, coordinators, supervision teams, and external parties. The following projects are available:

1. **Health Informatics**
   Coordinator: Willem-Paul Brinkman
2. **Coordinated Teams: A Search and Rescue Mission**
   Coordinators: Catholijn Jonker
3. **My Cultural Heritage**
   Coordinator: Martha Larson
4. **Programming Life: Synthetic Biology**
   Coordinator: Marcel Reinders
5. **Computer Games**
   Coordinator: Rafael Bidarra

4. Context Project Organization

The CP will consist of two parts: *course* and *project realization*. In the course part, students will acquire the knowledge and skills that will enable them to successfully pursue the project and realize the envisioned software product. The course part has four components:

1. **Context seminar** – Learn about the context domain and needs of users, acquire knowledge about the technology to be incorporated in the software product.
   Lecturers: To be determined by individual projects
2. **Project skills course** – Learn how to work in a team, plan and realize objectives, and reflect on the achievements.
   Lecturer: Dr. Laurens Rook (L.Rook@tudelft.nl)
3. **Information skills course** – Learn how to search and effectively deploy information in reports.
   Lecturer: Nicole Will (N.U.Will@tudelft.nl)
4. **Interaction design course** – Learn how to develop effective human-computer interaction (HCI) solutions.
   Lecturer: Willem-Paul Brinkman (W.P.Brinkman@tudelft.nl)

Each of the five projects listed in Section 3 is performed in team of not more than 5 students. We refer to these groups as *project teams*. Groups are formed and assigned to projects based on students' preferences, but with the following constraints: (1) A maximum of 4 groups per project is allowed, and (2) the preferences of students with more ECTS will have more weight. To allow flexibility in project selection, students are asked to specify three preferences in a ranked order. The CP coordinators and the EEMCS Education Department will do the final assignment.

The CP is going to take place in the first 10 weeks of Q4, starting from week 4.1 (from Tuesday 22 April) to week 4.10 (until 27 June). Lectures for the abovementioned courses are given in the first weeks of the project (week 4.1-4.4), while the rest of the time is dedicated to the realization of the envisioned software product. This realization goes through a number of

milestones, each of which requires concrete results – *deliverables* (see Section 6 for details). The project will be completed by the final assessment of the product (per project) and a plenary session (involving all projects). Both events are scheduled on **Friday, 27 June** (end of week 4.10). Please see refer to the course schedule/Blackboard for details of the planning.

The best performing team in a given project will be selected and awarded with a presentation and demonstration of their developed software product at the plenary session. The session is open to everyone involved in the CP. The plenary session will be followed by a demo market, in which all software products realized in the CP will be demonstrated to a broad audience. Software can be presented with laptops and/or posters. Tables and poster boards will be available. Please refer to the course schedule/Blackboard for details of the planning.

5. Role division between students and supervisors

The project teams will be supervised by the professors or (senior) lecturers and teaching assistants (TAs), and will also be guided through the discussions with external parties. The supervisors will operate in the **reactive** mode: Idea generation, proposals for meeting agenda's, and requests for feedback and support towards the supervisors will be the initiatives of the project team. The success of the project is the responsibility of a project team. We briefly describe different roles in the following.

Role of the professor/lecturer

- Project coordination
- Regular meetings with the TAs (weekly)
- Facilitates the involvement of the external party in the CP

Role of the teaching assistants (TA)

- Daily coaches and contact persons for the project teams
- Meet weekly with the students to formally check progress
- Keep overview of the delivered results, act towards students in case of delays
- Make sure that project teams follow the software development process
- Make sure that project teams take into account the context/user requirements
- Give feedback on draft versions of the documents, ideas, and solutions

Role of the external party

- Gives input to project teams in terms of problem description, desired solutions, and related requirements
- Provides feedback to the teams during the project
- Participates in the assessment of the developed software product at the end of the project

6. Deliverables

During the project, project teams need to submit a number of deliverables. Some of them serve to assess the progress of the project based on intermediate results, and some of them provide direct input for the final assessment of the project (see Section 8 for assessment process and criteria). For the latter category, the team first submits a draft version to the TAs and uses their feedback to generate the final version to be considered for the CP grade.

General guidelines for the documents

Some of the deliverables are documents and some are software. Each document deliverable should contain the following basic elements:

- Title page, including the title of the document
- Name, netid, and StudentIDs of the students and date
- Short abstract about the content of the document
- Table of Contents (TOC)
- Document specific content (the actual content of the deliverable)
- References to literature and other sources of information

List of deliverables and schedule

The list of the deliverables and corresponding deadlines for the submission of the draft and final versions is as follows:

- Product vision (draft: W3, final: W4)
- Product planning (draft: W4, final: W5)
- Emergent architecture design (drafts: W4, W9; final: W10)
- Lightweight SCRUM plans (week before each iteration)
- Final report (draft: W9, final: W10)
- Input for software quality evaluation by SIG (first version: W6, final code: W9)

The exact deadline is **Thursday, 23:55h** in the given week, EXCEPT for the final version of the Final Report in W10, which has the deadline on **Wednesday, 25 June 2014**. Not handing in documents in time will lead to exclusion from the project.

The following provides a guideline on how to generate each deliverable. Page numbers refer to "Document specific content", excluding title page, abstract, table of contents, and references.

Product vision (max. 8 pages A4)

This document must be created by following these fundamental questions [1]:

1. Who is going to buy the product? Who is the target customer?
2. Which customer needs will the product address?
3. Which product attributes are crucial to satisfy the selected needs, and therefore to the success of the product?
4. How does the product compare against existing products, both from competitors and the same company? What are the product's unique selling points?
5. What is the target timeframe and budget to develop and launch the product?

Each of the questions will lead to a corresponding section of the document. For background information and detailed instructions on how to write each section, please consult
[1] Roman Pichler, *Product Vision*,
http://www.scrumalliance.org/community/articles/2009/january/the-product-vision

NOTE: Question 4 requires a literature study and analysis of the existing alternatives to the envisioned realization of the target software product. The corresponding section of this deliverable will provide input for the grade of the Information Skills course (see Section 8 for assessment process and criteria).

Product planning (max. 6 pages A4)

This document will be created according to the following Table of Contents (TOC):

1. Introduction
2. Product
   2.1. High-level product backlog (set of epics aligned with the product vision)
   2.2. Roadmap (major release schedule, release goals)
3. Product backlog (first version with estimates and prioritized user stories)
   3.1. User stories of features
   3.2. User stories of defects (if applicable)
   3.3. User stories of technical improvements (if applicable)
   3.4. User stories of know-how acquisition
   3.5. Initial release plan (milestones, MRFs per release)
4. Definition of Done (backlog items, sprints, releases)
5. Glossary

For an overview of what the document should contain, consult Kenneth S. Rubin, "Essential Scrum: A Practical Guide to the Most Popular Agile Process" (login first at http://proquest.safaribooksonline.com/), in particular chapters 15 and 17. An example document will be posted on Blackboard.

Emergent architecture design (final version max. 10 pages A4)

Note that this document needs to be updated during the various sprints to present the current state of the design. The initial version can contain sketches/ideas of the architecture design; the document is then continuously extended throughout the sprints. An example document will be posted on Blackboard.

1. Introduction
   1.1. Design goals
2. Software architecture views
   2.1. Subsystem decomposition (sub-systems and dependencies between them)
   2.2. Hardware/software mapping (mapping of sub-systems to processes and computers, communication between computers),
   2.3. Persistent data management (file/ database, database design)
   2.4. Concurrency (processes, shared resources, communication between processes, deadlocks prevention)
3. Glossary

Lightweight SCRUM Plans (before each iteration)

Iterations in the implementation phase of the system last for one week. Before each iteration of the system's implementation, the project group submits a plan to the teaching assistant. The planning comprises:

- The selection of a set of backlog items (ordered by importance)
- A list of the tasks for each selected item
- The assignment of group members to tasks
- An estimation of the effort per task
- The actual effort per task (after the iteration is completed)
- Short reflection on the main problems and adjustments of the iteration planning

The planning of the previous iteration is used as input for the planning of the next iteration. At the end of the project the plans are added as an appendix to the final report. The students are free to choose the template for the planning.

Final report (max. 10 pages A4)

This deliverable is the main document about the developed, implemented, and validated software product. It will present the main functionalities of the product and discuss to which extent they satisfy the needs of the user. For this purpose, an evaluation of the functionalities performed using a well-justified method needs to be presented, as well as a failure analysis – where the product does not perform as needed. Furthermore, the Final Report will also contain a section describing the HCI module that was realized for the user interaction with the developed solution. This section will reveal what the students learned in the Interaction Design course and will be evaluated by the corresponding lecturer. The grade for Interaction Design will be assigned based on the content of this section (see Section 8 for assessment process and criteria). Finally, an outlook will be given regarding the possible improvements in the future and the strategy to achieve these improvements.

Note that this report should not repeat the material from Product Vision, but should complement it by providing results as response to expectations and strategy described in the Product Vision document.

The TOC of the Final Report will be as follows:

1. Introduction, including a brief problem description and end-user's requirements
2. Overview of the developed and implemented software product
3. Description of the developed functionalities
4. Special section on interaction design (development of the HCI module)
5. Evaluation of the functional modules and the product in its entirety, including the failure analysis
6. Outlook

Input for Source code quality assessment

The source code has to follow sound software engineering methods (*e.g.*, including unit tests). The Software Improvement Group (SIG) is going to evaluate the source code (including tests), once during the development phase in W6 and once at the end of the project in W9. The feedback given by SIG for the version uploaded in W6 should be considered for the final version. The user will evaluate the product in W6 and feedback will be given to steer further development. The evaluation of the final version in W9 is part of the final grade.

7. Implementation

The system implementation starts in W5. The implementation follows an incremental and iterative software development process.

Planning

Each iteration needs to be planned according to the lightweight SCRUM planning documents described before. At the end of each iteration the plan is checked: which features have been finished, which tasks are still open. The results of the check are considered for the planning of the next iteration.

Test-driven

The implementation should be test-driven: First, implement a test for a feature and then start implementing the feature. Make use of a xUnit test framework.

Integrate the different parts of your implementation as soon and as much as possible. Use build environments (*e.g.*, Ant and Maven). Develop corresponding integration tests to check whether the integrated system works.

Always have a running version

The implementation follows the concept of "always have a running system". After each iteration, each project group gives a 5-minute demonstration session of the most recent implemented feature(s) followed by a 20-minute discussion of the design, implementation, tests, and iteration planning.

8. Assessment

The assessment of the CP will be guided by the following end-terms that are derived from the learning objectives.

The student is successful in:

- Formulating a vision regarding the role and added value of ICT in that context and specifically in the scope of the defined project in order to maximally satisfy user's needs.
- Performing a 'state of the art' analysis on the topic addressed in the project assignment
- Finding relevant scientific literature, assessing the value and implications of the existing work for the envisioned problem solution and assessing the added value of new proposed components of the developed solution with respect to the existing work.
- Selecting and justifying the appropriate software engineering methods for the software product
- Understanding the requirements related to the software quality of the developed solution, having the necessary development rigor and implementing the software product as a solution to the user's problem according to these requirements.
- Understanding the principals of the scrum approach for software engineering, and planning and managing the project accordingly
- Validating, presenting (in written and oral form), and demonstrating the implemented software product from technical, functional, and end-user perspective

Based on the above, the grade assigned to the project team will be composed as follows:

Overall score = 0.35 * Context + 0.35* Product + 0.05 * Process + 0.25 * (Final assessment)

Grade components:

- Context
  - Developed, implemented and validated solution for the context problem
  - Product vision
  - Information skills
- Product
  - Final architectural design
  - Source code quality (also evaluated by SIG)
  - Interaction design

- Process
  - Product planning
  - Lightweight SCRUM plans

- Final assessment
  - Presentation and demonstration
  - Final report

The **Context** component stands for the ability of the students to understand the problem of the user and the constraints and requirements of the context domain in which the problem is defined, and to recognize the possibilities to solve the problem in a way that the solution has maximum possible utility for the user. The **Product** component stands for the development, implementation, and software quality of the developed solution. The **Process** stands for the ability of the students to organize and plan their work and to understand and follow the standard software engineering procedures (i.e. development rigor). The **Final assessment** regards the ability of the students to present and demonstrate their solution to the user.

To pass the project, the Project Skills course needs to be passed and passing grades need to be obtained for ALL 4 components of the final grade. Passing grade per component is obtained if both following conditions are fulfilled:

  - Two (in case of Process and Final assessment) or two out of three sub-grades (in case of Context and Product) are at least 5.8,
  - The overall grade for the component is at least 5.8.

A grade is assigned to the entire project team. Deviations from this grade per team member may be applied if a member performs significantly better or worse than the rest of the team. The justification for such deviation must come from the team and should be communicated to a member of the supervision team.

Turning a non-passing grade per component into the passing one is possible, but requires successful completion of additional assignments given to the project teams or individual students.

The following paragraphs describe the grade requirement and criteria for Project Skills and per grade component.

Project skills

This course will be assessed based on an essay, in which the students will reflect on the process of working in the group, the role and accomplishments in the process of themselves and their colleagues. Based on the essay, the Pass/Fail assessment will be made. Students need to pass this course in order to pass the CP. For more information on this course, requirements and planning of the realization of the essay, please contact the organizer, dr. Laurens Rook.

Developed, implemented and validated solution for the context problem

This component refers to the software product being the main result of the project. The product will be assessed during the final presentation and demonstration in W10. Assessment will be performed by the supervision team and the external parties and will be done based on the extent to which it satisfies the needs expressed by the external party in the given problem context. Note that this assessment will focus on the implemented functional modules only and their performance from the perspective of the end user. In other words, the sole criterion for the assessment here is the **utility** of the implemented product for the end user.

Product vision

This deliverable will be assessed based on the extent to which it fulfills the requirements specified in Section 6.

Information skills

The assessment will consist of two stages. The first stage includes a practical part that needs to be passed in order to complete the CP. At the second stage, it will be assessed how the acquired information skills have been deployed in the project. Since this deployment is primarily done when writing the Product Vision report, the related aspects of this report, and then in particular its part related to question 4 (see Section 6 for details), will be assessed and a grade will be assigned. This grade will become a part of the Context grade component. For more information on this course, contact the organizer, Nicole Will.

Final architectural design

This deliverable will be assessed based on the extent to which it fulfills the requirements specified in Section 6.

Source code quality evaluated by SIG

the Software Improvement Group (SIG) is going to evaluate the source and test code of the final product (if it has been implemented with PHP, Java, C, C++, or C#). The code criteria used by SIG are:

- Readability (naming of classes, methods, attributes and variables)
- Structure (size of classes and methods)
- Complexity (complexity of methods, depth of inheritance trees)
- Documentation

In projects developed with other technologies for which an automated assessment by SIG is not possible, the Final Product will be assessed by the context teacher and assistants using the same criteria.

Interaction design

The assessment will consist of two stages. The first stage includes a practical part that needs to be passed in order to complete the CP. This part consists of online quizzes. The second stage includes the assessment of the HCI component of the final software product, which is to be described in the dedicated section of the Final Report (see Section 6 for details) and which will be assessed by Dr. Brinkman. This grade will be a part of the Product grade component. For more information, contact the course organizer, dr. Willem-Paul Brinkman.

Product planning

This deliverable will be assessed based on the extent to which it fulfills the requirements specified in Section 6.

Lightweight SCRUM plans

This deliverable will be assessed based on the extent to which it fulfills the requirements specified in Section 6.

Presentation and demonstration

Here the students are required to present and demonstrate the developed software product, highlight its main functionalities, convince the external party regarding the utility of the product and reflect on the development process. Each presentation lasts for 20 minutes and needs to include a live demonstration of the system. The presentation is followed by 10 minutes of discussion.

Final report

This deliverable will be assessed based on the extent to which it fulfills the requirements specified in Section 6.