Software Architecture Project Info (Requirement & Architecture phase)

This document contains info about the main task of the software architecture project, as well as templates for the first delivery. As you should be well acquainted with the COTS (Khepera simulator, XNA, Android and iPhone SDK) by now, we will not go into details about that.

1. The Task

1.1 The robot simulator

The task is to create a robot that will collect all balls on a map and place them near the light. The map to be used is maze.

The following information is given:

- There is one light
- There is at least 4 balls, but the exact number is unknown
- Balls are not placed in awkward positions, like in corners, or in extensions to wall ends.
- The robot can be set to start anywhere, and can start facing left/right/up/down. It will not start at an angle.
- The navigation does not need to be carried out using a map.

1.2 A Game in Android or XNA or iPhone

The task is to make a functioning multiplayer game using XNA , Android or iPhone SDK, based on your own defined game concept. However, the game must be design according to a specified software architecture. The game must be a multiplayer game of some complexity, but it can be turn- based or similar (does not need to be a network game).

2. Group Specific Tasks

Requirements of the quality attributes focus are following:

The group should develop robot controller / game with focus on a primary and secondary quality attribute:

- Primary quality attribute (all groups have to focus on this quality attribute):
  - Modifiability: The software architecture and implementation should be easy to change in order to add or modify functionality.
- Secondary quality attribute (the groups can choose one of the following):
  - Testability: Applicable for all COTS
  - Availability: Only applicable if you go for a network game.
  - Safety: Only applicable for the robot to avoid that the robot getting stuck.
  - Usability: Applicable for XNA, Android, and iPhone.

The choice of the quality attributes must be reflected in quality requirements, in the architectural design and in the final tests of the application.

3. Your Solution

The perfect implementation is not the ultimate quest (holy grail) of this project, but the quality of the implementation counts together with all the documents in the final grade. Also, the thing to remember is that the implementation and the architectural description should reflect each other. Most likely you have to change/modify/update the architecture based on what is being implemented. The final code and implementation should be well documented, easy to compile-and-run. Much hassle and too much time spent on trying to get a system to run will only irritate the course staff and possible degrade the result.

4. About the Documents

The document should be written in English and only acceptable format of delivery is the PDF format. The project will be graded on the final delivery, but documentation with big holes during the project can also be penalized.

4.1 Requirements Document

In this document you are to list the requirements for your application. We should use the following structure:

- Front page that includes:
  - Group name
  - Group members
  - Chosen COTS (Robot, XNA, Android or iOS)
  - Primary chosen quality attribute
  - Secondary chosen quality attribute
- Introduction:
  - Description of the project and the phase (requirement).
  - If game project: Description of the game concept should be sufficiently described and explained here.
  - Structure of the document.
- Functional Requirements:
  - A complete list of functional requirements you have to fulfill in order to complete the task. Each requirement must have an unique ID, e.g. FR1, FR2, FR3 etc... Can also be decomposed into sub-requirements such as FR3.1, FR3.2 etc.
  - Tables can be used for this purpose.
- Quality Requirements (scenarios):
  - Write at least scenarios for the most relevant quality attributes (e.g. modifiability, testability, safety, usability etc.).
  - Use (textual/table) scenarios of the type used in chapter 4 of the book.
  - Make the quality requirements measurable/testable with some values that can be checked later. Make estimates for the response measure.
  - Tables are recommended to specify quality requirements.
  - Every quality requirement must have an ID, e.g. A1, A2, M1, M2 etc.
- COTS Components and Technical Constraints:
  - Describe the constraints your architecture has due to your choice of COTS (Khepera, XNA, Android or iOS framework). If you have some other constraints relevant for your project, it should be stated here.
- References:
  - List references to books, articles, web-pages, documents etc. that you have used.
- Issues:
  - Optional point of issues you faced working with this of the project and the document.
- Changes:
  - To be describes changes carried out with this document from first draft until final delivery including all improvements based on feedback from course staff and others.

4.2 Architectural Description Document

This is your main document, with the architectural description for the project. The document should contain (based recommendations from IEEE 1471):

- Front page that includes:
  - Group name
  - Group members
  - Chosen COTS (Robot, XNA, Android or iOS)
  - Primary chosen quality attribute
  - Secondary chosen quality attribute
- Introduction:
  - Describe the project, document structure and the phase (requirement).
  - If game project, describe game concept sufficiently here!
- Architectural Drivers/Architectural Significant Requirements (ASRs):
  - The main drivers (functional, quality and business requirements) that affect architecture and the system mostly.
- Stakeholders and Concerns:
  - The stakeholders of the system, and their concerns.
- Selection of Architectural Views (Viewpoint):
  - A list of the views (viewpoints) that you will use in the architectural documentation, their purpose, target audience (which stakeholder are you addressing), and what notation will be used for each view.
  - All the groups need to choose at least a *logical view*, a *process view* and a *development view* or similar.
  - Candidate views (viewpoints) can be found in TextBook or 4+1 paper
- Architectural Tactics:
  - Describe the tactics for architecture to meet the quality requirements.
- Architectural and Design Patterns:
  - Selection of architectural/design patterns to be used in your architecture.
- Views:
  - Document architecture using your selected views (at least logical, process and development) through diagrams and supporting/explaining text.
- Consistency Among Views:
  - Describe inconsistencies among the views in your architecture.
- Architectural Rationale:
  - Here you will explain why you have chosen the architecture you have chosen. This part should motivate for why the architecture will work and fulfill the quality, business and the functional requirements.
- Issues:
  - Describe issues you faced working with and describing the architecture.
- References:
  - A list of any references to books, articles, web-pages, documents etc. that you have used.
- Changes:
  - To be describes changes carried out with this document from first draft until final delivery including all improvements based on feedback from course staff and others.

(4.3 Code Skeleton) - optional

We recommend that you implement and hand in a code skeleton for your architecture if you have time. It is really important to start the implementation as early as possible, and to ensure that it is possible to implement the architecture you have designed within the constraints given.

4.4 Template for references

Here are some short examples of how to write correct references to books, articles and web- pages respectively:

Book:

• Len Bass, Paul Clements, Rick Kazman, "Software Architecture in Practice – Third edition", Addison- Wesley, September 2012.

Article:

• Phillipe B. Kruchten, "The 4+1 View Model of architecture", IEEE Software Magazine, Volume 12, Issue 6, 1995.

Webpage:

- Microsoft, "Xbox Live Indie Games – xbox live indie games development"; Web: http//www.xna.com/ Accessed 19th February 2013.