Project Inception Release

Date: Week 8
Type: Team Project
Grade: 15% of the project

Summary

Execute the first sprint of the project plan to analyze, design and implement selected stories from the use-cases of the project. The project inception release presents the high-level functional areas and shows how the resulting system will solve the target problem. During the inception iteration, analyze the business requirements of the project and derive user and functional requirements to be developed. Define functional requirements using use-cases, UI requirements using wireframes and analyze requirements using a requirements model. Start the implementation of the project and complete a "potentially shippable product increment" that is a UI focused evolutionary prototype that displays the main page / screen of the application and showcases the functional areas of the project with selected stories as described in the sprint backlog for the first iteration.

Submission Checklist

1. **Project Software Model (PSM)** developed with Visual Paradigm, shared by all team members and version controlled using VPository. The version control history is critical in providing evidence of individual work of each team member. Please commit your changes often, every time you work on the project and include a detailed description of the work that was completed. The software model must contain:
    a. *Requirements Model* that defines business requirements, user requirements using use-cases and functional requirements using structured use-case scenarios.
    b. *Domain Model* that describes the domain the system operates in
    c. *Wireframes* that define the user interface requirements for the stories developed during the inception iteration.

2. **Potentially Shippable Product Increment (PSPI).** The first product increment produced is UI focused that is tracked under version control using GIT and BitBucket. This implementation demonstrates the main page / screen of the application and showcases the functional areas of the project with selected stories as described in the sprint backlog for the first iteration.

3. **Project Plan Update**. Updated project plan and the risk management plan to reflect the work done during the sprint.

Detailed Requirements

Part I (45%) Requirements Specification and Modeling. Create a requirements model that contains all use-cases, use-case diagrams, use-case definitions and any associated requirements artefacts. The overall structure of the Requirements model is shown in Figure 1: Requirements Model Structure. For the purpose of this part of the assignment allocate a functional area per team member. Conduct requirements modeling and create the following requirements analysis artefacts:

1. (5%) Create the system context diagram of the project and a system overview diagram with the following characteristics:
    a. Visualize the boundary of the system to help guide what is in the scope and out of the scope of the project.
    b. Identify the primary, supporting and offstage actors. Use custom stereotypes to differentiate the types of actors: *<<actor>>*, *<< supporting>>*, and *<<offstage>>.*
    c. Within the system boundary include the *project summary use-case*
    d. In the system overview diagram, identify the functional areas of the system, their respective summary use-cases and relevant actors.

    NOTE: It is expected that these diagrams are developed collaboratively by all the team members.

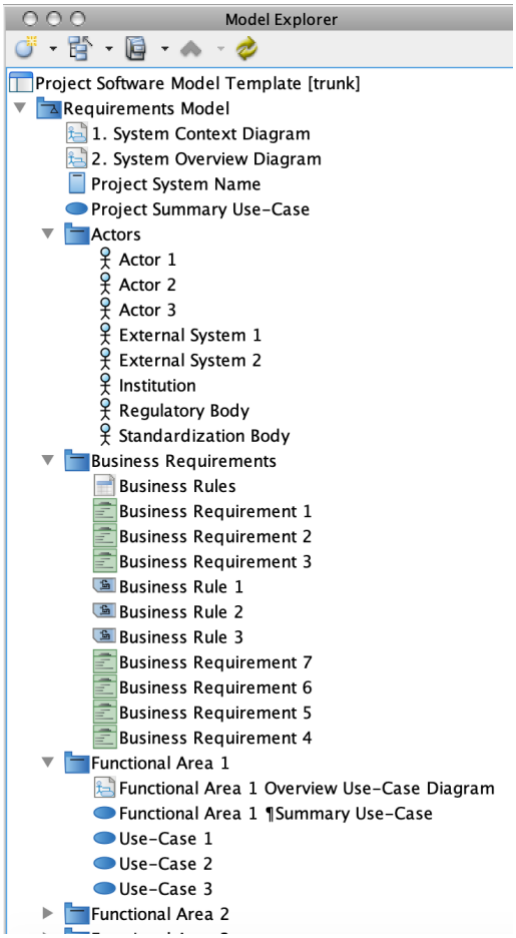

Figure 1: Requirements Model Structure

2. (10%) Define the business requirements and business rules for your project using a requirements list and a business rules grid. Store all business requirements and rules in a package entitled Business Requirements, part of the Requirements Model. Ensure each business requirements has a name that is representative and the correct type. The business requirement *description* shall be done in story format: "*As a <stakeholder>, I want to <business requirement>, so that <business need>*". In the same package, create a business rules grid to store any business rules you may discover during requirements elicitation and analysis.
NOTE: It is expected that business requirements and business rules are defined by each team member for the stakeholder they are championing.

3. (5%) Each functional area shall contain one use-case diagram that provides an overview of the user requirements in the respective functional area and their relationships. *NOTE: It is expected that this*

work is completed individually by each team member in the functional area they are overseeing (one per team member).

4. (10%) Define one project summary use-case that provides the highest level of overview of the project's functionality. Identify and define high-level extensions that are critical to the project. *NOTE: It is expected that team members work collaboratively in the definition of this use-case*.

5. (10%) For each functional area, define the functional area summary use-case that provides the overview of the user requirements in the functional area of the system. The summary use-case shall define a main success scenario as well as well main extensions. It is expected that many of the steps in this use-case are links to other use-cases to be defined in subsequent iterations. The use-case details shall clearly identify the author of the use-case. *NOTE: It is expected that this work is completed individually by each team member in the functional area they are overseeing (one per team member).*

6. (10%) Fully define a user-goal use-case in each functional area of the system. Each use-case shall contain a fully developed main success scenario, an exhaustive list of extensions and the steps associated with each extension. CRUD use-cases are shown only as one use-case where necessary as "Manage <type of data>" (e.g. Manage patient data). The use-case details shall clearly identify the author. *NOTE: It is expected that this work is completed individually by each team member in the functional area they are overseeing (one per team member).*

Part II (10%) Domain Model. Create a domain model within the PSM that defines the project's glossary, the domain concepts and their relationships. Review the domain model with the domain expert to ensure the proper terminology is used and the definitions are clear and correct.

1. Create a *project glossary* that will be used to define the ubiquitous language of the project as defined in Domain Driven Design (DDD) methodology. The development and validation of the project glossary with the domain experts is critical to the success of the project in subsequent phases of the project inception.
2. Create *a domain class diagram* with the following characteristics:
    a. Domain concepts are shown as a classifier with the <<concept>> stereotype customized to show a dotted line boundary (to differentiate it from a design/implementation classes) or domain-specific icons (e.g. medical device, assignment, tool images)
    b. Concepts are linked with association relationship that have the following detailed defined: name, direction and cardinality
    c. Avoid relying on advanced UML semantics such as generalizations in this incipient phase of the project as they will not be clear to domain experts and will impede communication. You can use, for example, simple association labeled "is-a-kind-of"
    d. Do not use design / implementation relationships such as compositions, aggregations, dependencies.

e. Avoid design / implementation relationship names such as “USES”, “HAS-A”, “IS-A”. Instead use domain specific terms.

Part III (10%) User-Interface Requirements. Using Visual Paradigm, create wireframes to describe the user interface requirements of the project associated with the use-cases defined. Link the use-case scenario steps with the wireframes that describe how the steps are accomplished in the user interface. See the Visual Paradigm documentation for a tutorial. NOTE: if students have experience with and would like to use a different wireframing tool they can do so; however, students will have to export the wireframe as an image and create a VP wireframe based on it so that it can be linked with the use-case scenario. Evaluation and feedback will be provided through the Visual Paradigm tool only.

Part IV (20%) Product Increment. Start the development of the project by designing and implementing the user-interfaces related to the use-cases defined in this sprint. This product increment is UI focused and must showcase the different functional areas of the project using a start-up screen / page and navigation menu. Selected stories are implemented as described in the project plan but no other business logic is required, and data shown is hard-coded at this stage. The user shall be able to navigate through the UI screens designed to exercise “mock functionality”.

Part V (10%) Project Plan Update. Update the project plan to reflect all the work that was completed for this iteration and to prepare the product backlog for the next iteration as follows:

1. All product backlog items (PBIs) completed in this iteration have been estimated using story points and have been broken down into dev-tasks.
2. Each dev-task has been estimated in effort hours. The estimation is recorded in the dev-task description. Track the time you have spent implementing each dev-task and record the actual effort upon completion.
3. Update PBIs with short comments that describe the work completed, *each time the PBI is worked on*. Such comments might describe problems faced, solutions found, research done, work that was completed.
4. PBIs that were completed during the sprint are marked as completed and moved from the product-backlog back to their original iteration. All completed tasks have been marked as such.
5. All incomplete PBIs are divided into a completed PBI and a new PBI with the new PBI being added to the high-level product back log, the detailed level product backlog and a future iteration.
6. Risks identified in the risk management plan are monitored as demonstrated by comments posted.

Notes:

1. The **professionalism of your submission**, clarity of written communication is extremely important. The ability to communicate your knowledge is as important as the knowledge itself. Up to 40% of the mark for any written work can be deducted due to poor presentation / communication: document organization (10%), layout (10%) spelling (10%), title page (10%)

2. **All assignment shall be submitted by the deadline.** Late submissions will be penalized with 10% per day for up to 3 calendar days after which the assignment cannot be submitted anymore. **An email must be sent** should you choose to submit a late assignment. If no such emails are received the solution will be posted. **Assignments are not accepted after the solutions have been posted.**

3. In this group assignment, **team members must be responsible for contributing an equal share to *EACH PART* of the assignment**. Each individual member must clearly be able to demonstrate their individual contribution to each of the requirements outlined in this assignment. See the Academic Honesty at Sheridan.

4. Submission is done in electronic format **using SLATE DropBox. DO NOT email your submission.**