Product Design Exercise

JD Kilgallin

CPSC:480

09/28/22

Notes

- Agenda: Quiz 4, Project 2 overview, Exercise 3
- Midterm Review on Monday, followed by special office hours. Email me questions ahead of time to ensure the best answer to them.
- Midterm next Wednesday in class.
  - Will be a mix of short-answer and modeling.
  - Will reuse quiz and textbook problems.
  - Exam will be long, but curved.
  - **1 page of notes (2-sided) allowed.**
- Not covered on exam:
  - History of software 1950s-today.
  - Applying to software engineering roles
  - GitHub and markdown
  - Rigorous UML diagram standards

Project 2 Part 2: Requirements Modeling

- Use draw.io, Microsoft Visio, or a similar tool for diagrams. Do not use a simple bitmap editor like Microsoft Paint as these images are difficult to modify as requirements evolve.
- Diagrams do not need to use the formal UML standard but should clearly and accurately convey the relevant information. See examples in lecture slides or textbook.
- From your proposed project, develop a class model capturing the relevant object types, their attributes and operations, and relationships.
- Construct a scenario model (sequence/swimlane diagram) for each functional requirement. Each team member must construct at least one scenario model, but the team as a whole is responsible for completion and accuracy.
- Construct at least one behavioral model (state diagram) modeling part of your system.

Project 2 Part 3: Software Specification

- Write a full specification of the software product, including the following.
  - The artifacts of part 2.
  - All of the diagrams.
  - A paragraph explaining why this project has value.
  - Any operating environment requirements and assumptions.
  - Interfaces for users and other systems.
  - Anything else relevant to the project.
  - Content should all be posted to GitHub, including source materials for all diagrams.
- You may find a template online; the book recommends web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc.

Project 2 Checkpoint

- Due Sunday, Oct 2, 11:59 PM
- GitHub repo should contain:
  - Each student's alternative proposal that was considered.
  - **Source files** for all diagrams (not simply images).
  - Project specification document.
  - At least two commits or pull requests from each team member (e.g. proposal + scenario model)
  - A release containing the pdf version of the specification.

Project 2 Checkpoint grading

- 10% - Original project proposal & requirements from each user (individual)
- 10% - Sequence diagram contribution & 2 commits or pull requests from each user (individual)
- 05% - Proposal submission and prioritized requirements
- 10% - Class diagram
- 20% - Sequence diagrams
- 10% - State diagram(s)
- 30% - Specification
- 05% - Pdf release and directions followed

Project 2 Part 4: Project Planning

- Select 5 or more highest-priority requirements that can & should be met first
- For each selected requirement, create a set of product backlog tasks indicating work to be done to meet the requirement. Each team member should complete task definition for at least one requirement, and the group should then meet to review and agree on task definitions. Omit design tasks; focus on implementation
- For each requirement, create a GitHub issue with requirement description, definition of done (i.e. how can someone verify it is complete), and task list.
- When requirements have been analyzed, meet to discuss tasks identified:
  - Estimate the number of hours needed to complete the task and add it to the item in the task list. The recommended methodis to use the median from Planning Poker.
  - Order the tasks on an issue from highest-priority to lowest.
  - A designated team member should take thorough notes on this meeting content.

Project 2 Part 5: Product Design

- Decide on the programming language(s) to be used for the project and what tools may need to be used to develop the product.
- Define the classes and data types needed for the product, and their attributes (including data types), data structures, and methods (including method signature – input/output types). Where possible, identify what other methods might be called by each method. Design for modularity and functional independence by minimizing complexity of communication between components, using the class model from project specification.
- Define the file structure of the project to reflect the class model within selected language.

Project 2 Part 5: Product Design

- Define how the program will interact with other systems, including how data will be entered into the program, how it will be stored, and how it will be displayed or output.
- Define how the program will be deployed (built, distributed, and launched) and operated.
- Design the user interface for key components of the program. Examine the behavioral model to identify possible components that may require a user interface, and examine the scenario models to identify what elements need to be present on each component.
- Determine what would need to be done to create a minimal prototype of this product and begin meeting requirements with multiple team members contributing simultaneously.

Project 2 Part 6: Sprint

- In order to practice use of a project plan and comparing time spent to original estimates, select the highest-priority tasks that can be completed in two weeks at 5 hours per week per team member based on projected estimates (10 estimated hours per developer). Identify who would complete each task (tasks do **not** need to all be actually completed, although you will need to do some work for at least one task) and assign the issue to that team member. You may do this in a meeting or over a chosen communication platform.
- Each team member should take their assigned tasks and identify what they need in order to complete each task (knowledge, code, tools, additional requirement detail), as if they were going to complete it. Record this information in a comment on the GitHub issue for the task.

Project 2 Part 6: Sprint

- Hold a “standup” meeting (virtual or in-person, but everyone must attend) after everyone has reviewed their assigned tasks, and have each team member describe initial obstacles they are facing in completing the task and what they would expect to have done by the “next” standup meeting. A designated team member should take thorough notes on the meeting content.
- Each team member should make at least one commit related to one of their assigned tasks. It is fine if additional, unstructured progress is made, but work should be linked to a GitHub issue for the task.
- Produce a brief report describing the state of the project as of the due date in markdown.

Project 2 Final submission

- Due **Sunday, Oct 16, 11:59 PM.** 2% extra credit if all submitted by **Friday, Oct 14, 11:59 PM;** 1% by **Saturday, Oct 15, 11:59 PM.**
- Submission is through GitHub. GitHub repo should contain:
  - All content from checkpoint.
  - Issue for each task included in the sprint with title, body, assignee, project, and prereqs linked.
  - Comments for each assigned issue describing developer's needs.
  - Product design document and schematics.
  - Notes on project planning meeting and standup meeting.
  - At least one commit from each developer related to an assigned task.
  - Markdown report on current status.
  - Release containing final version of design doc and standup notes in pdf format.

Project 2 Grading

- 70% of the final project grade is shared and 30% is individual, as follows.
- 25% - Individualized grade on project checkpoint
  - 80% shared component of checkpoint grade = 20% of final project grade
  - 20% individual component of checkpoint grade = 5% of final project grade
- 10% - Task definition (individual)
- 05% - Identification of needs/obstacles, standup contribution, and issue comments (individual)
- 10% - Work item progress and tracking (individual)
- 20% - Design
- 10% - Sprint plan
- 10% - Meeting notes
- 05% - Report
- 05% - Directions followed

Participation

- In the GitHub handouts folder is a template called “Team Participation Survey”. Each team member should submit a copy with their name by **Monday, Oct 17, by 11:59 PM** in physical form or via Brightspace. This will count as a quiz.
- Additionally, you may submit a partial form, anonymous or not, at any point if you would like to raise any concerns. The purpose of this form is to ensure that all members are meeting expectations and credit is being assigned appropriately.
- Students may not receive full credit for team grades if they meet the following criteria:
  - Average of contribution or expectations score from peers of 1.9 or less.
  - GitHub contribution history and/or meeting reports do not reflect appropriate participation.
  - No justification for poor performance or excuse from course requirements is provided
- Conversely, students may receive up to 5% extra credit for the following criteria:
  - Average contribution or expectations score from peers of 4.1 or higher.
  - GitHub history and/or meeting reports reflect outstanding skill and team commitment.

Exercise 3

- Review exercise 2a Pokedex Project Plan example solution. Suppose a user wanted to access the data via a command-line application instead of the GUI.
- Select one or more programming languages (and optionally frameworks) in which to design a product for these requirements. Explain your reasoning.
- Describe the data structures that you might use in the application. Describe how you would maintain the data in persistent storage and how the data would be accessed by the application. Effective architecture will allow the data storage to be changed by substituting just one module.
- Describe the modules you would use, and the interfaces between them. An effective architecture will allow the user to implement the command-line application by substituting just one module.

References

- Flowchart Maker and Online Diagram Software. Draw.io. 2005-2022. JGraph Ltd.
- Software Requirements Specification. Dr. Kirstie Hawkey. 2011. Dalhousie University.
- How do you play Planning Poker? planningpoker.com. 2020.

- *Reading for next lecture: Review lectures, assignments, and other handouts up to this point, and prepare questions for midterm review. Email questions ahead of time for the best answer.*