CMPSC 280
Principles of Software Development
Fall 2015

Laboratory Assignment Two: Starting with Team-Based Implementation of Software

Introduction

Now that you understand how to use the Git version control repository, the Bitbucket site for managing these Git repositories, and the use of Markdown language system for the purpose of documenting software, we are going to start team-based implementation of a simple software system. Paying careful attention to both the phases of the software lifecycle and the roles of members of a development team, as explained in Sections 1.5 through 1.8 of the textbook, you should interact with a customer to learn the requirements of a program and then design, implement, document, and release the system. Your team of software engineers should also select one individual who will serve as the maintainer of the system and thus be responsible for interacting with any future customers or developers who will want to use or extend your product.

Understanding the Lifecycle and Organizing Your Team

Before you start working on this assignment, please carefully review the content at the end of Section 1.8 in SETP to learn more about the software development lifecycle. Students who want to better understand the lifecycle and how it might be organized are also encouraged to read SETP's Section 2.2. Once all of the members of your team understand the phases of the software development lifecycle, discuss the roles in a development team, as outlined in Section 1.7 of the textbook.

Although your team may have fewer members than there are roles, you should pick a single individual to serve in each of the capacities mentioned in SETP. For instance, one member of your team should be responsible for requirements analysis and definition. To ensure that this assignment can be completed in a timely manner, your team is not required to pick people to serve in the dedicated role of a software tester. As we will comprehensively test and evaluate your product in a later laboratory assignment, your developers can also perform informal testing.

Even though the program under development for this laboratory assignment may be simple enough to not necessitate structured implementation in a team following a development lifecycle, you and your team members should resist the temptation to simply start to implement the program. For the sake of learning and to ensure that you are best prepared to implement much more complex systems, immerse yourself in the software development lifecycle and your roles. In particular, your team should also ensure that you do not "blur" the roles of your team members — that is, the person(s) in charge of implementation should not be involved in interviewing the customer.

Eliciting Requirements from the Customer

You are responsible for implementing a data generator that should take a list of numbers as input and produce a list of lists as output. You are given the following specification for the data generator.

> For an input list of objects, denoted L, the data generator must produce all of the lists that can be obtained by swapping two adjacent items in L.

For list $L = \{1, 2, 3, 4\}$, the customer wants the data generator to output $L = \{L_1, \ldots, L_6\}$ with

$$L_1 = \{2, 1, 3, 4\}$$
$$L_2 = \{3, 2, 1, 4\}$$
$$L_3 = \{4, 2, 3, 1\}$$
$$L_3 = \{1, 3, 2, 4\}$$
$$L_4 = \{1, 4, 3, 2\}$$
$$L_6 = \{1, 2, 4, 3\}$$

The customer knows that the component that you must create will be a part of a larger system that has not yet been fully implemented. You are responsible for implementing this data generator so that it functions according to the provided specification. However, please note that the stated requirements may not be entirely correct! It is the job of your team to interact with the customer to ensure that the system is implemented as desired. Using Markdown, you should write a requirements document that fully explains the inputs, outputs, and behavior of the data generator.

Designing the System

Working with the members of your team and leveraging the content in the requirements document, you should create a design for your system. As you are finalizing the object-oriented design, you should try to develop answers to relevant questions such as: How many classes will you use? What will be the relationship between the classes? What methods will the classes have? What will be the inputs and outputs of the methods? Is the design easy to understand? After answering these questions, you should use Markdown to write a design document with text and diagrams that explain the system. Since the main focus of this assignment is not design diagrams, students may use any reasonable tool to create their diagrams and, if necessary, include them as separate files in their team's repository. When clarification of the system's specification is necessary (e.g., due to writing ambiguities), the designers should interact with the individual(s) who elicited the requirements.

Implementing the System

The developer(s) and tester(s) on your team should take the requirements and design documents and start to think about how the system will be implemented. Your task is to ensure that the program faithfully adheres to the descriptions already produced by other members of your team. When an aspect of the requirements and design documents is not clear, the developer(s) must talk with the people who created these documents to resolve any outstanding concerns. The creators of these documents must quickly commit any changes to their repository so as to best ensure that the requirements and design of the system are in sync with its implementation.

Ensuring the Effective Operation of Your Team

When you start to work on this laboratory assignment, it may seem as though the designer(s), developer(s), and tester(s) "do not have any work to do" because the requirements of the system have not been established. Yet, if you carefully think about the work that you must complete for this assignment, it will become clear that this is not the case! For instance, one member of your team should be tasked with creating all of the needed means for communication with tools such as Slack and Bitbucket. Additionally, team members who are waiting to complete their chosen tasks should consider investigating the use of tools, like Trello, to organize their team's efforts. Finally, it is important for team members to spend time creating the templates for their deliverables and then carefully "sharpening their tools". For instance, the developer(s) on the team can ensure that they have a smoothly functioning development environment that will support the implementation of well-documented and correct code. Please see the instructor if you have questions about this matter.

Carefully Review the Honor Code

The Academic Honor Program that governs the entire academic program at Allegheny College is described in the Allegheny Academic Bulletin. The Honor Program applies to all work that is submitted for academic credit or to meet non-credit requirements for graduation at Allegheny College. This includes all work assigned for this class (e.g., examinations, laboratory assignments, and the final project). All students who have enrolled in the College will work under the Honor Program. Each student who has matriculated at the College has acknowledged the following pledge:

> I hereby recognize and pledge to fulfill my responsibilities, as defined in the Honor Code, and to maintain the integrity of both myself and the College community as a whole.

It is understood that an important part of the learning process in any course, and particularly one in computer science, derives from thoughtful discussions with teachers and fellow students. Such dialogue is encouraged. However, it is necessary to distinguish carefully between the student who discusses the principles underlying a problem with others and the student who produces assignments that are identical to, or merely variations on, someone else's work. While it is acceptable for students in this class to discuss their deliverables with their classmates, items that are nearly identical to the work of others will be taken as evidence of violating the Honor Code. In particular, make sure that your team members do not inappropriately communicate with the members of other teams.

Summary of the Required Deliverables

This assignment invites you to submit printed and signed versions of the following deliverables; please see the instructor if you have questions about any of these items. Please make sure that your team creates a version control repository in Bitbucket to store all of the deliverables for this assignment; since you will again select your own teams, please create a repository with a numerically-based name and share it with the course instructor. Also note that all of the written documents must be prepared with Markdown and, for submission, converted to PDF using `pandoc`.

1. A complete description of the roles that each team member fulfilled.
2. A requirements document that fully describes the features of your system.
3. Expressed in writing and as a simple technical diagram, the design of your system.
4. Well-documented Java source code that fulfills all of the system's requirements.
5. A tutorial that explains how to use all of the features of your finished system.
6. Individually completed and submitted reviews of all of the team members.