Software Development Studio

Christina Wang

December 10, 2004

Note: This document is the final report for an independent research project on developing a course to teach students how to write "good code", where an understanding of what good code is and how it might be taught were part of the goals of this project. Rather than try to define "good code", we decided to create a course that would provide practice and feedback on writing and reading programs, so that students will experience the difference between good and bad code themselves.

Table of Contents

Course Justification

The Department of Computer Sciences offers many courses for students to learn about theory, software, and hardware. There are also classes considered as electives that let the student explore different areas of study within the Computer Science field. However, not many of these classes offer students the chance to apply what they have learned to something similar to a real world situation. Software Development Studio is a class that can provide this experience to students by training them for a profession in Computer Science and giving them a perspective of working in industry.

This class would help enhance a student's programming skills in the context of a team. Only recently have the CS classes introduced the concept of pair programming for projects, and only a few classes give students the opportunity to work on a project as a team. Writing code as a team is different than writing code for you or for one other person. One of the activities the students will do in this class is look at examples of other people's code. These examples will demonstrate maintainability, usability, and readability of code. In the project, students will experience the importance of writing code with these attributes. They will also have to learn how to work with another person's code and possibly how to continue work on another person's code, a common activity when coding as a team in industry.

This class is important because it also provides a controlled environment for students to apply their skills. Like other professions such as architecture, law, engineering, or medicine, students must practice applying what they have learned so that when they have graduated, they will know what is expected of them. In architecture, students are required to take a theory class one semester, and then a studio class the following semester where they apply what they learned in a project with a real client. Law and medical students must also spend time in their field before they actually start practicing. This class is constructed to prepare the students for what might happen when they start working in industry except with less risk. Students will understand the severity of missing deadlines

without the possibility of losing their job. Students will receive feedback on performance without the possibility of losing a monetary reward. Therefore, the environment of this class will allow the student to mature without severe consequences.

Lastly, this class will further prepare a student to work in industry by developing their soft skill in areas such as working in teams, presentation skills, and debating skills. As mentioned earlier, students who go into industry after graduation will find themselves assigned to a team and will need to know how to communicate with their team members for the team to run efficiently and effectively. They will also eventually be asked to present in front of an audience, and having the opportunity to practice this more will help them improve. Also, in the beginning stages of development, lots of ideas will be proposed but only one will be implemented. Students will need to learn how to communicate to others that their idea is the better idea. Having spoken with some recent graduates in industry, they all agreed that these soft skills are essential to being successful in a company.

This class is different in structure and topic than any other CS class being offered. It does, however, bear a slight resemblance to the Software Engineering class, CS 373. The difference between Software Engineering and this class is that Software Engineering focuses more on the processes involved with software development, whereas Software Development Studio will focus more on the programming aspect of software development as well as its business value. This class is meant to complement the Software Engineering class but can be taken by students who have not taken Software Engineering. Overall, this class will be very valuable to students who are interested in working in industry.

Software Development Studio
Course Syllabus

Course Objectives

This course is designed to approximate an experience of working in a commercial software development team, and therefore you should come in with a professional manner and attitude. The class will be divided into teams that work together to complete a multi-component product by the end of the semester. You will face the kinds of obstacles developers in industry often encounter, including requirement changes and task reassignments. You will also participate in class discussions, team presentations, and feedback and evaluations. The obstacles are designed to teach you the importance of the code you write and also manage your working environment and adapt to it. Through the structured activities of the class, you will improve your programming skills, learn how to work in small development teams, and gain soft business skills.

Resources

Required text: Course packets for this class are for sale at the Co-op for $XX.00
Class newsgroup: utexas.class.cs3XX. (newshost: news.cs.utexas.edu)

Project Overview

This project is a class effort to develop a product and will require interaction with the other teams. The project is divided into five phases and an overview of each phase is described below. The specific details of each phase will be described in a Project Guidelines handout.

Phase 0 – Initial Phase
The professor will give the class the initial set of requirements with an overall design already in place. Students will examine and discuss in class the proposed design for the project and finalize the overall design plan. The class will then be divided into teams so that each team is assigned at least one component of the product. There will be one team dedicated to testing.

Phase 1 – Design Phase
Once your team has been assigned a component, your task is to come up with a design plan for your component, a test plan for your component, and assign each team member responsible for developing a part of the component. The testing team will create a design plan for system integration testing. At the end of this phase, you will give a presentation highlighting your design decisions and discuss your reasoning.

Phase 2 – Development Phase
You will begin developing once the professor has approved your design plan. At the end of this phase the team will go through a code review and again give a presentation on what they have done so far. The purpose of this presentation is to update the class on your progress.

Phase 3 – Reassignment and Requirement Change Phase
In this phase, the professor will announce a change in the requirements. This change may affect everyone or perhaps just one team. Each team member will also be reassigned to work on a different part of the component. If the team has been following the team's coding standards, diligently documenting their code, and logically organizing and maintaining the code, this transition should be relatively easy. Each team member will continue to develop their newly assigned part.

The testing team will have completed their test cases for system integration testing by this time. All other teams should be submitting their components to the testing team for system integration testing and continue working on defects found by the testing team.

At the end of the phase, the team should be near completing the component. The team's presentation for this phase will be about how the team managed the task reassignment and requirements change and discuss what you feel would have made the situation better.

Phase 4 – Testing Phase
Teams are now working out remaining defects found by the testing team and should be finalizing their components. At the end of this phase, teams will present on their testing experiences and how close they followed their initial testing plan.

Evaluations

Peer Evaluations
At the end of Phase 2 and Phase 4, you will provide constructive criticism on ability to cooperate, performance, contributions to the team, attitude towards work, areas of strength and areas needing improvement for each of your teammates. These will be submitted to the professor who will then anonymously relay to each teammate.

Professor Evaluations
The professor will provide feedback on the activities of the team and their progress after each phase so that the team knows where they stand and what they need to improve.

Class Evaluation (extra credit)
Since this is a new class, we ask that you write an evaluation of the class and how well it went. Was it everything you expected or less? Did you gain the knowledge you were looking for? What are some things that could have been done differently that would improve the class and still meet its objectives?

Grading

There are no exams for this class. Your class grade will be determined according to the following percentages.

Development work: 40%
Presentations: 30%
Evaluations: 20%
Class Participation: 10%

Software Development Studio Instructor's Manual

December 2004

Table of Contents

Introduction

This manual is written for the professor and the TA's administering the class. The first section will briefly describe an overview of the class and the following sections will describe in detail about lectures and the project. Following those sections will be a suggested schedule and section on grading policy.

Class Overview

The class will compose of some lectures and a class project. The purpose of the lectures is to help guide the students in their upcoming tasks for the project. When a lecture is not scheduled, students will either have presentations or have a work day where they work in their teams. On work days, you should go around and talk with each team to get an idea of how they are doing, where they stand, give advice, and answer any questions if they have some.

This class would be best taught in a room with several tables for teams to sit together and work together. The room should also have several computers, a wireless connection, and a media console available.

Class Lectures

There are ten topics to cover in lecture and they are listed below in the order they should be presented, coinciding with the project schedule.

1. Fundamentals to working in teams
2. How to design programs
3. Design trade offs
4. Unit testing
5. Presentation Skills
6. Coding Standards
7. Things to consider when coding
8. Examine selected examples of well-written code
9. Refactoring code
10. Testing approaches

These topics can be combined to form one class period or can span more than one class period. Below is a list of each topic and a suggested amount of time spent on the topic.

Fundamentals to working in teams	1 hour
How to design programs	1.5 hours
Design trade-offs	1 hour
Unit testing	0.5 hour
Presentation skills	1 hour
Coding standards	1 hour
Things to consider when coding	0.5 hour

Examine selected examples of well-written code	2 hours
Refactoring code	2 hours
Testing approaches	1 hour

Class Project

Project Phases

The class project is to develop a product. This is broken up into several phases so that it can be easily explained.

Phase 0 – Initial Phase

In the initial phase, due to the limited amount of time, you will provide the class with the initial set of requirements with an overall design idea already in place. Before the semester begins, you should have decided on a project for the students to work on and created an initial design plan for it. The plan should divide the project into several components so that each team will have at least one component to implement. This information will be passed online to the students in the Project Guidelines handout for which there is a template found in Appendix A. Students will examine and discuss in class the proposed design for the project and finalize a design plan. You will then divide the class into teams so that each team is assigned at least one component of the product. There will be one team dedicated to testing. It would be best to form teams where the teammates have different strengths, but if this is not possible, you can randomly form teams or have the students choose who they want to work with. When assigning components, it would be best to assign a component to the team that is best skilled to work on it, but again, if this is not possible, you can randomly assign components to teams or have the teams choose. Be sure to include these dates in the Project Guideline handout if you intend for the students to choose.

Phase 1 – Design Phase

In this phase, teams are assigned to come up with a detailed design plan and a test plan for their particular components. They also need assign each team member the responsibility of developing a part of the component. This is turned in to you so you are able to assess the development factor of their grade. The testing team will create a design plan for system integration testing. At the end of this phase, each team will give a presentation highlighting their design decisions and discuss their reasoning.

Phase 2 – Development Phase

Students will submit to you their design plans and you will look over them and approve them. Then students will be able to start developing. While they are developing, you are to act as their manager and provide any guidance they may need. At the end of this phase the teams will conduct a code review and again give a presentation on what they have done so far. The purpose of the presentation is to update the class and you on the team's progress.

Phase 3 – Reassignment and Requirement Change Phase

In this phase, you will announce a change in the requirements. This change varies depending on the product. It may affect everyone or perhaps just one team. Also, each team member will also be reassigned to work on a different part of the component and this list of new assignments will be submitted to you for grading purposes. If the team has been following the team's coding standards, diligently documenting their code, and logically organizing and maintaining the code, this transition should be relatively easy. Each team member will continue to develop their newly assigned part.

The testing team will have completed their test cases for system integration testing by this time. All other teams should be submitting their components to the testing team for system integration testing and continue working on defects found by the testing team.

At the end of the phase, teams should be near complete with implementing the component. The team's presentation for this phase will be about how the team managed the task reassignment and requirements change and discuss what they feel could have made the situation better.

Phase 4 – Testing Phase

Teams are now working out remaining defects found by the testing team and should be finalizing their components. At the end of this phase, teams will present on their testing experiences and how close they followed their initial testing plan.

Project Specifics

Team Presentations

Each team presentation has a different purpose, depending on what phase just finished. When teams present, the audience is instructed to provide them feedback on their presentation skills. Each person in the class must present at least twice to receive credit. The purpose of presenting twice is so the student will have an opportunity to practice any suggestions they received as feedback the first time they presented. To space it out, at least three members of a team must be presenting at each presentation. Any team members not presenting should be at the front with their team, acting as support.

Evaluations

Peer Evaluations

At the end of Phase 2 and Phase 4, each student will provide constructive criticism on each of their teammate's ability to cooperate, performance, contributions to the team, attitude towards work, areas of strength and areas needing improvement. You can administer this as a questionnaire online, a form you hand out in class, or as a paper the students should write. When you receive these, you will have to parse the information and regroup all the feedback by individual students so that you can anonymously relay that feedback to them.

Professor Evaluations

You will also provide feedback on the activities of the team and their progress after each phase so that the team knows where they stand and what they need to improve. This falls under a manager's responsibility.

Class Evaluation (extra credit opportunity)
Since this is a new class, student feedback on the class is an excellent way to constantly improve this class and make sure that it is useful for the students. They have the option of writing a paper that evaluates the class and how well it went. Some questions to pose are Was it everything you expected or less? Did you gain the knowledge you were looking for? What are some things that could have been done differently that would improve the class and still meet its objectives?

Task Reassignment

The obstacle of task reassignment was added so students can experience first hand the importance of writing code that is readable and maintainable. Knowing ahead of time that someone else will be reading and reviewing their code will hopefully encourage students to learn how to write neat and organized code. This aspect of the project will help guide students in reaching one of the course objectives. The task reassignment is completely up to the teams.

Requirements Changing

This obstacle introduces students to an event they may potentially have to face at work. This will also reinforce the importance of creating well designed components and writing well organized code. Before this occurs students will participate in class discussions about refactoring and ways to modify existing code. They will then be able to apply these methods to the project.

You can choose to modify as many requirements as you see fit, as long as it is still possible for the product to be completed by the end of the semester. Some requirement changes may be the addition of requirements or changes may occur in the interface or maybe some rules need to be changed. The requirements changed may affect only one team or several teams. The only thing to keep in mind is not to change a requirement such that a new component is needed, and hence a new team is needed to work on it.

Optional Guest Involvement

If it can be worked into the schedule, people who work in industry can be invited to help mentor, advise, and perhaps even critique the students and their work. An ideal guest would be one who deals with issues similar to those presented in the class. You may also like to invite guest speakers to present on some of the lecture topics that should be covered.

Suggested Weekly Schedule

Week	**Activities**
Week 1 – Phase 0	Lecture: Fundamentals to working in teams

	Lecture: How to Design Programs
Week 2 – Phase 0	Discussion: Project and Initial Design Plan Finalize Design Plan Divide class into teams and assign components
Week 3 – Phase 1	Lecture: Design trade-offs Lecture: Unit Testing
Week 4 – Phase 1	Lecture: Presentation skills
Week 5 – Phase 1	Presentations Lecture: Coding standards
Week 6 – Phase 2	Lecture: Things to consider when coding Lecture: Examine selected examples of well-written code
Week 7 – Phase 2	
Week 8 – Phase 2	Code Review
Week 9 – Phase 2	Presentations Team member Evaluations Lecture: Refactoring code
Week 10 – Phase 3	
Week 11 – Phase 3	
Week 12 – Phase 3	
Week 13 – Phase 3	Presentations Lecture: Testing approaches
Week 14 – Phase 4	
Week 15 – Phase 4	Team member Evaluations Class Evaluation (optional)

Class Deliverables

1 Component Design Document

This document describes how the team plans to design their component.

2 Component Test Plan

This document describes how the team plans to test their component.

3 Team Assignment Memos

After each rotation, the team will turn in a memo listing which team members are working on what parts of the component.

4 Presentation Feedback

Teams will submit the feedback they received on their presentations in memo form so that you can assess their improvements in future presentations

5 Code Review Results

This is a memo that records the results of the teams' code reviews.

6 Peer Evaluations

Students will provide peer evaluations for each of their team members.

7 Final Code

The final code should be commented and complete.

Grading

There are no exams for this class. A student's class grade will be determined according to the following percentages.

Development work: 40%
Presentations: 30%
Evaluations: 20%
Class Participation: 10%

Development work

Each team will submit a list of the team members with their assigned part of the component twice in the semester. Students should include their name in the comments of the code they write, but you should also double check this. When looking through the code, look to see if they have considered the issues that were brought up in class around writing code. Their grade will be determined based on the overall presentation of their code.

Presentations

Students will have to present at least twice. Their grade will be determined based on how well they improved from the first time they presented to the last time they present.

Evaluations

Students must complete evaluations for each team member as well as provide feedback during class presentations to receive credit. The grade will be determined based on the student's improvement in the second half of the semester.

Class Participation

Students will get points for participating in class discussions and also attending on work days. If a student attends every work day, they should have enough points to give them full credit for class participation.

Appendix A
Project Guidelines Template

Project Overview
This project is a class effort to develop a product and will require interaction with the other teams. The product this year will be to develop a _(fill in blank)_.

Project Phases
The project is divided into five phases and these phases are explained below.

Phase 0 – Initial Phase
The professor will give the class the initial set of requirements with an overall design already in place. Students will examine and discuss in class the proposed design for the project and finalize the overall design plan. The class will then be divided into teams so that each team is assigned at least one component of the product. There will be one team dedicated to testing. The scheduled dates for this phase are listed below.

Date	**Activity**
	Discuss initial overall design
	Announce teams and component assignment

Phase 1 – Design Phase
Once your team has been assigned a component, your task is to come up with a design plan for your component, a test plan for your component, and assign each team member responsible for developing a part of the component. The testing team will create a design plan for system integration testing. At the end of this phase, you will give a presentation highlighting your design decisions and discuss your reasoning.

Date	**Activity**
	Component Design Document due
	Component Test Plan due
	Team Member Assignments due
	Class Presentations

Phase 2 – Development Phase
You will begin developing once the professor has approved your design plan. At the end of this phase the team will go through a code review and again give a presentation on what they have done so far. The purpose of this presentation is to update the class on your progress. When you have completed component testing on your component, you can start submitting code to the testing team. During this phase, you will also fill out team member evaluations for each of your team members.

Date	**Activity**
	Team Code Review
	Team Member Evaluations due

	Class Presentations

Phase 3 – Reassignment and Requirement Change Phase
In this phase, the professor will announce a change in the requirements. This change may affect everyone or perhaps just one team. Each team member will also be reassigned to work on a different part of the component. If the team has been following the team's coding standards, diligently documenting their code, and logically organizing and maintaining the code, this transition should be relatively easy. Each team member will continue to develop their newly assigned part.

The testing team will have completed their test cases for system integration testing by this time. All other teams should be submitting their components to the testing team for system integration testing and continue working on defects found by the testing team.

At the end of the phase, the team should be near completing the component. The team's presentation for this phase will be about how the team managed the task reassignment and requirements change and discuss what you feel would have made the situation better.

Date	**Activity**
	Team Member Assignments due
	Final date to submit code to Testing Group
	Class Presentations

Phase 4 – Testing Phase
Teams are now working out remaining defects found by the testing team and should be finalizing their components. At the end of this phase, teams will present on their testing experiences and how close they followed their initial testing plan. At the end of this phase, you will conduct another evaluation of your team members.

Date	**Activity**
	Final code due
	Team Member Evaluations due
	Class Presentations

Presentations

Each person in the class will present at least twice. Since there are four different class presentations, each team will have at least three members of the team presenting at once to make sure that everyone has a chance to present twice. The dress for your presentations is business casual. If you decide to present using a PowerPoint presentation, be sure to include on the title page your team name and the date of the presentation. After the title slide, please include a list of the members in your team and a slide for an overview of the presentation. The presentation slides should be numbered and copies should be given to the professor and TA the day of the presentation. Specifics of the presentations are described below.

Phase 1 Presentation

Describe your component design to the class. Discuss the different ideas the team came up with and reasons for why you decided on the ideas you did. Your presentation should be about _(fill in blank)_ minutes long with _(fill in blank)_ minutes for questions totaling _(fill in blank)_ minutes.

Phase 2 Presentation
Update the class on how much development work the team has accomplished. Discuss what strategies, if any, were used to assign tasks for each member. Also discuss any challenges you guys have encountered or are currently facing. Your presentation should be about _(fill in blank)_ minutes long with _(fill in blank)_ minutes for questions totaling _(fill in blank)_ minutes.

Phase 3 Presentation
Describe your experience with the task reassignment and the requirements change. Were you able to adapt easily? What happened with your team? What happened with your team's code? What did you learn from this obstacle? Also explain, in hindsight, what you think you could have done to prevent any hardships brought about by change. Your presentation should be about _(fill in blank)_ minutes long with _(fill in blank)_ minutes for questions totaling _(fill in blank)_ minutes.

Phase 4 Presentation
Describe your initial testing plans your team created in Phase 1 and discuss how closely your team aligned with them. Analyze the types of defects found and discuss how some these could have been prevented or even caught at earlier stages. Explain how your team handled the defects found by the testing team for your component.

Requirements
(Describe your requirements.)

Product Design
(Describe your product design here. Please write it as a Design Document so that students can have something to use as a model when they draft their own Design Document for their components.)