CPS353 - Principles of Software Engineering: Project

Di Wang

Learning Objective

Software engineering not only embodies a set of key concepts and principles, it also entails a set of skills. Without proficiency in at least some of these skills, you cannot call yourself a software engineer. This course includes a project to help you get a taste of these skills as well as experience the key lessons of software engineering first-hand.

Few interesting programs can be built by a single person, so the project for this course is team-oriented. In addition to all the other skills and lessons that are to be conveyed, you will also be learning how to cooperate and communicate effectively.

In this project, you and your teammates will be defining a software product and seeing it through every stage of development. Because of the short span of this class, you are not expected to deliver a marketable product, but the result should be at least a compelling prototype that could serve as the basis for defining a real product or attracting venture capital. :)

Project Scenario

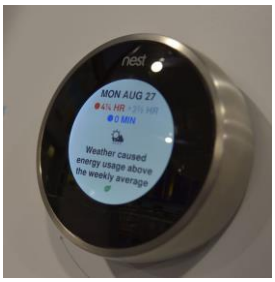
You and your teammates have the flexibility--within certain parameters--to choose the product to develop and the features it provides. To provide some "spiritual" guidance, your product should attract the attention of the " Utopia World Venture Capital" investment fund, which specializes in promoting unusual products that make a better world through strong communities. They have special interests in software that are targeted to a large group's needs. I will personally act as the Technical Consultant (Liaison) of SWVC, and each team will be coming to me to seek guidance about what they can do to have a better chance of attracting additional funding. Each team has considerable autonomy, however, in defining its product and managing its team. For example, you are responsible for choosing a product that is within your means to develop within the allotted time. From time to time I may sense opportunities in the marketplace and suggest small changes to improve your market position. I will consider the possibility of teams developing competing products.

Risk Control

Consistent with the concept in class, your project is to be conducted in a risk-centric fashion. Everything you do is about minimizing downside risks and maximizing return on up-side risk. Everything you turn in should be justified in terms of addressing a risk.

Here are some places to get inspiration (Android Bluetooth Smart Switch):

- Many organizations like academic departments, hobbyist or enthusiast communities, public agencies, and the like could benefit from IoT(Internet of things) which allow the internet to embedded with or work for a physical device.



- There is a lot of interest these days in mobile computing with applications to help the disabilities (e.g., remote controlling, speech recognition, etc.)

It is not common for projects to be built on existing infrastructures or libraries, or to integrate existing components.

Team and Project Organization

It is highly recommended that each team develop some sort of organizational structure (division of responsibilities, mechanism for making decisions, means for communication) to achieve both coherence and effective parallel effort. You may want to go agile management (e.g., Scrum). You may want a team leader; you may want to have a managerial lead and a design lead. You may want to have functional specialization and testing; you may want to assign your teammates to implement a certain component; you may want to do Agile software development.

You really, really, really should have regular meetings (Meeting Agenda is required); you should have a mailing list and chat set up, as well as a repository (Bitbucket) and issue tracker for all project information.

[Add Meeting Title/Name]

Location – Address

Meeting Minutes [Date] - [Time]

Call to order	[Record call to order here]
Roll call	[Present List]
	[Not Present List]

MEETING MINUTES FORM	
1. Point one to discuss	[Start text here]
2. Point two to discuss	[Start text here]
3. Point three to discuss	[Start text here]
4. Point four to discuss	[Start text here]
5. Point five to discuss	[Start text here]

ADJOURNMENT	This meeting was adjourned at [current time], by [scribe's name]
Minutes submitted by: **[Name]**	Minutes approved by: **[Name]**

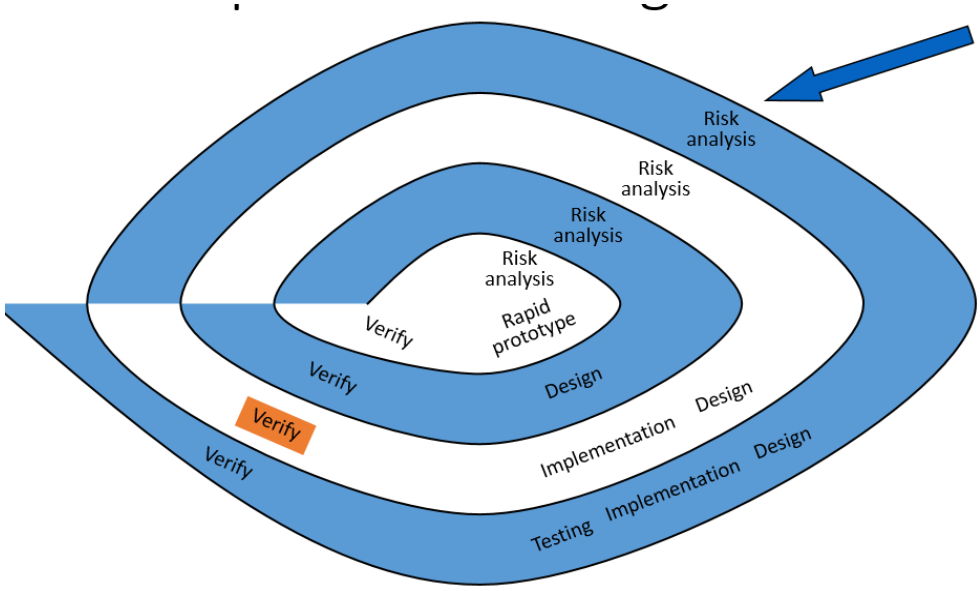
Project Schedule and Documents

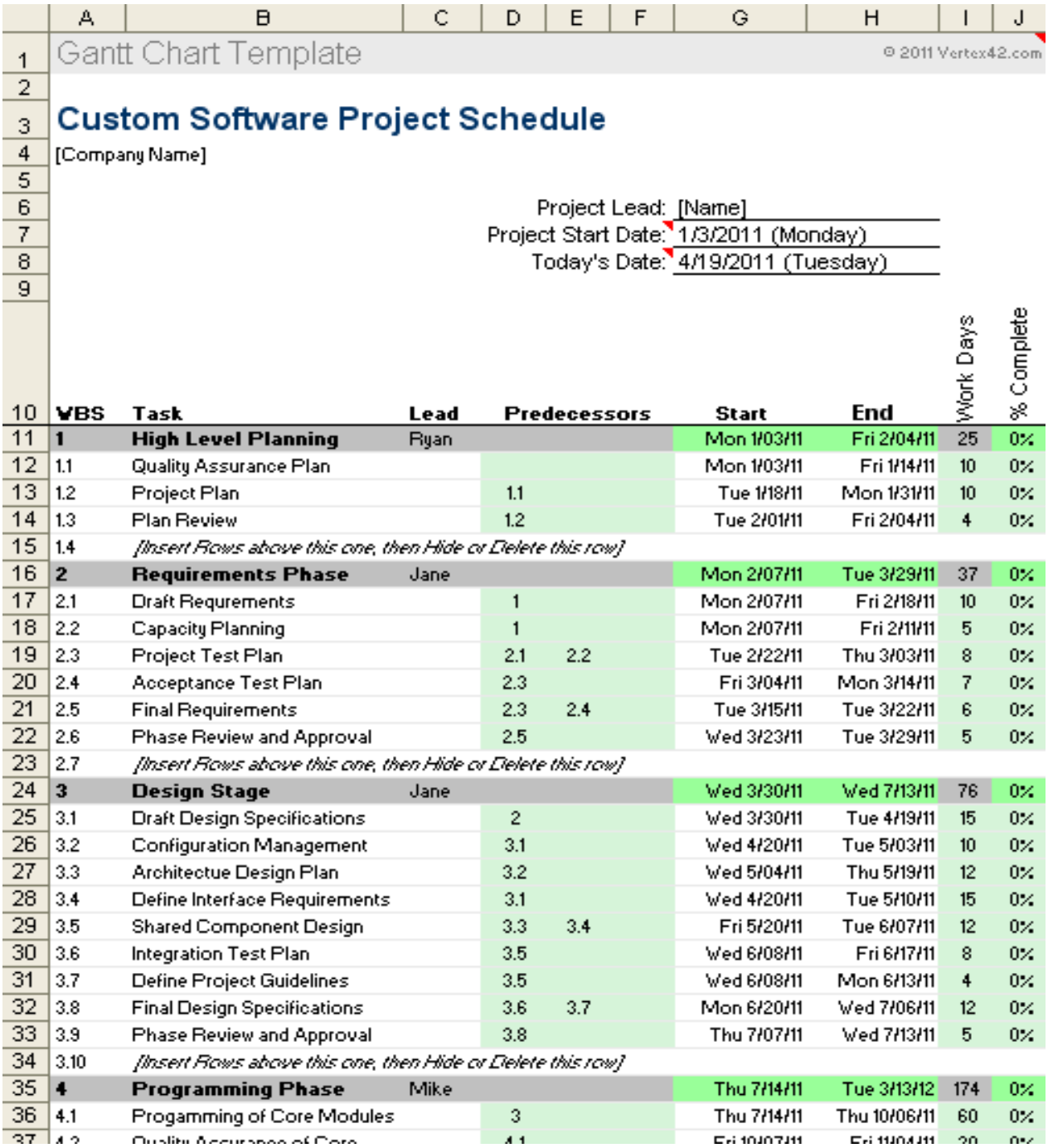
I want to recommend your project is going to be managed based on Boehm's Spiral development model. This model guides the team through a series of repeated cycles of increasing detail (hence the term spiral). Such a model is necessary in situations where there is significant uncertainty in the product development process. The spiral model allows the delaying of decisions until the cost of making the decision is lower than the cost of not making the decision. In the extended definition of the Spiral model--which is what we are using here--there are five primary steps to each cycle. Each step is focused primarily on the issues identified for the current cycle.

- Identify ``stakeholders'' and their win conditions
- Determine objectives, possible solutions (may change stakeholders), constraints on solution (e.g., time, win conditions)
- Evaluate possible solutions, identify, and resolve risks (i.e., unknowns)
- Develop and verify next-level “product” or “prototype”, plan next cycle
- Validate and review product, ending with commitment for next cycle

Although each cycle focuses on resolving a certain class of risks, any previously unresolved risks should be monitored for realization or (preferably) elimination. Realization of unresolved risks requires a reprioritization.

If there were any money involved, there would also be a budget document. You may also want to keep supporting documents such as meeting notes and e-mails.





Gantt Chart Template

Custom Software Project Schedule

[Company Name]

Project Lead: [Name]
Project Start Date: 1/3/2011 (Monday)
Today's Date: 4/19/2011 (Tuesday)

WBS	Task	Lead	Predecessors		Start	End	Work Days	% Complete
1	**High Level Planning**	Ryan			Mon 1/03/11	Fri 2/04/11	25	0%
1.1	Quality Assurance Plan				Mon 1/03/11	Fri 1/14/11	10	0%
1.2	Project Plan		1.1		Tue 1/18/11	Mon 1/31/11	10	0%
1.3	Plan Review		1.2		Tue 2/01/11	Fri 2/04/11	4	0%
1.4	*[Insert Rows above this one, then Hide or Delete this row]*							
2	**Requirements Phase**	Jane			Mon 2/07/11	Tue 3/29/11	37	0%
2.1	Draft Requirements		1		Mon 2/07/11	Fri 2/18/11	10	0%
2.2	Capacity Planning		1		Mon 2/07/11	Fri 2/11/11	5	0%
2.3	Project Test Plan		2.1	2.2	Tue 2/22/11	Thu 3/03/11	8	0%
2.4	Acceptance Test Plan		2.3		Fri 3/04/11	Mon 3/14/11	7	0%
2.5	Final Requirements		2.3	2.4	Tue 3/15/11	Tue 3/22/11	6	0%
2.6	Phase Review and Approval		2.5		Wed 3/23/11	Tue 3/29/11	5	0%
2.7	*[Insert Rows above this one, then Hide or Delete this row]*							
3	**Design Stage**	Jane			Wed 3/30/11	Wed 7/13/11	76	0%
3.1	Draft Design Specifications		2		Wed 3/30/11	Tue 4/19/11	15	0%
3.2	Configuration Management		3.1		Wed 4/20/11	Tue 5/03/11	10	0%
3.3	Architectue Design Plan		3.2		Wed 5/04/11	Thu 5/19/11	12	0%
3.4	Define Interface Requirements		3.1		Wed 4/20/11	Tue 5/10/11	15	0%
3.5	Shared Component Design		3.3	3.4	Fri 5/20/11	Tue 6/07/11	12	0%
3.6	Integration Test Plan		3.5		Wed 6/08/11	Fri 6/17/11	8	0%
3.7	Define Project Guidelines		3.5		Wed 6/08/11	Mon 6/13/11	4	0%
3.8	Final Design Specifications		3.6	3.7	Mon 6/20/11	Wed 7/06/11	12	0%
3.9	Phase Review and Approval		3.8		Thu 7/07/11	Wed 7/13/11	5	0%
3.10	*[Insert Rows above this one, then Hide or Delete this row]*							
4	**Programming Phase**	Mike			Thu 7/14/11	Tue 3/13/12	174	0%
4.1	Progamming of Core Modules		3		Thu 7/14/11	Thu 10/06/11	60	0%
4.2	Quality Assurance of Core		4.1		Fri 10/07/11	Fri 11/04/11	20	0%

Also, the documents mentioned above should be delivered to me before the deadline---May 1st. My grading of these documents will constitute the “review and commitment'” as the final parts of the developing life cycle.

Project Description

In this project-centered course, you'll design, build, and distribute your own unique application for the Android mobile platform. I'll provide you with a set of customizable building blocks that you can assemble to create many different types of apps, and that will help you become familiar with many important specificities of Android development. When you have ability to building an android app, in addition to having a personalized app that you can use and share, you'll have the skills and background you need to move on to cooperate with your teammates to create an advanced android project.

What you'll need to get started:

This project-centered course is designed for learners who have some prior experience programming in Java, such as an introductory of Java Programming.

You will need a computer with a stable Internet connection; an Android phone; a Bluetooth module; and an Arduino controller board - we'll use open-source hardware Arduino board that you can use your mobile device to control the board via Bluetooth. We'll use Android Studio as IDE; it is compatible with most computer and operating systems. You can find detailed system requirements here: https://developer.android.com/sdk/index.html#Requirements.

Project Schedule

- **Week 3:** Team formed.

  Constitute a team with a size as specified in syllabus. Submission before Feb 10th includes a list of team member names, student IDs, and e-mail addresses:

  Name1Student ID1e-mail address1
  ...

  Week 7: Customer Requirements and Project proposal.

  *Discover the **actual** needs of your customers.* You are starting with a problem from the customer's point of view and elaborating that into a set of requirements for an improved or entirely new system.

The requirements should be recorded in a way that they are easily accessible and analyzable in future stages (e.g., pictures, scenarios, stories). Use of scenarios to record the current work practices of your customers is highly recommended. You should analyze these scenarios (and other data) to determine requirements. You can plan your project proposal based on the requirement. The due time of submission is March 3rd.

- **Week 8:** Time Estimation

  Software time estimation is the process of predicting the most realistic amount of time required to develop or maintain software. This generally involves estimating the number of people who will work on the project, what they will work on, when they will start working on the project and when they will finish. Once you have this information, you need to lay it out into a calendar schedule. That is important on the way of controlling the risk management. In this project, you need to create a Spreadsheet which can check whether each phrase can be finished on time.

1. Four estimates

   Design: 30 minutes
   Code: 45 minutes
   Testing: 1 hour
   Debug: 1 hour

2. My Spreadsheet

Sessions	Start from	Break(5mins)	End at	Tips
Design	22:33 10/9/14	22:49 10/20/14	23:03 10/20/14	On time
Coding	23:10 10/9/14	23:36 10/20/14	00:00 10/21/14	On time
Testing	00:20 10/9/14	1:00 10/21/14	1:40 10/21/14	Over 15 mins
Debugging	9:20 10/21/14	none	11:21 10/21/14	On time

- **Week 9:** Product Design and Software Architecture

  Define the product. Based on customer requirements, time constraints, and other factors, design and specify the best possible product. You may wish to construct a crude prototype, mock-ups, or crucial scenarios. Use of the system architecture diagram is meant to emphasize the external communication between software and hardware, that is how a user interacts with and perceives it, rather than its internal structure.

  The structure of a software product often has little to do it with defining its function, but has everything to do with being able to implement, test, and enhance the software at a reasonable cost, e.g., on schedule.

- **Week 10:** Software Design.

  Building on your software architecture, design your software with the following development artifacts in mind:

  - UML Diagram
  - Design Pattern
  - Coupling and cohesion

  Your design methods, principles, and sample of source code should be documented.

- **Week 11:** Software Testing.

  In many Agile methodologies, test cases are written first as a way of defining the project and driving the development process. Here, I encourage you design test case to validating the correctness and integrity of your final software product.

- **Finals Week:** Product Demonstration.

  Prepare a demo for the manager and the venture capitalists. The success of this demo and the frankness of the postmortem will determine continued funding of your project.

  The postmortem is an analysis of your project to determine success and (primarily) failure factors: What you did, what you would do the same,

and what you would do differently next time? The focus is not on blame, but on improvement.

Demos and Post-mortem presentations will take place during the time scheduled for the final. The time will be divided evenly among the teams (about 5-10 minutes) plus time for responses from the project manager and a class postmortem. To provide context for the other students in the class, provide a description of your product, a sketch of the relevant parts of your software development process and risk management before getting into the details of the introduction.

Lists of Documents (Not all required):

1. Meeting agenda
2. Project schedule
3. User case scenario
4. Time estimation spreadsheet
5. System architecture diagram
6. UML Diagram
    a. User case diagram
    b. Activity diagram
    c. State diagram
    d. Sequence diagram
    e. Class diagram
7. Test case