The COS 333 Project

Copyright © 2020 by
Robert M. Dondero, Ph.D.
Princeton University
Overview

• A simulation of reality
• In groups of 3-5 people...

• Build a substantial networked three tier application
Three-Tier App

- **Data management tier**
  - Stores data persistently
  - Usually a DBMS

- **Processing tier**
  - “Business logic”
  - Ties data management tier to user interface tier
Three-Tier App

• User interface tier
  – Options…
  – Textual program
    • Desktop, laptop
  – Graphical program
    • Desktop, laptop, tablet, phone
  – Browser
    • Your processing tier provides code; browser interprets
    • Desktop, laptop, tablet, phone
Potential Projects

• **Approach 1**: Develop an app to fulfill your needs
  – Maybe a simulation of reality

• **Approach 2**: Develop an app to fulfill the needs of others
  – Reality!

• I strongly recommend approach 2
  – See *Project Ideas* web page
Working with Instructors

• You will manage your project
• Your TA will help
  – First-level adviser
    • Not your manager
    • It’s your project, not the TA’s
• Lead instructor will help
  – Directly, or through your TA
  – Second-level adviser
    • Again, not your manager
Process

• Use an orderly process

• This is **NOT** a process:
  – Talk about the app at dinner
  – Hack some code together
  – Test it a bit
  – Do some debugging
  – Fix the obvious bugs
  – Repeat until the semester ends
Process

• Formal software engineering processes
  – Waterfall, rational unified, agile, extreme,…
  – Overkill for 333
  – But some process is essential

• I recommend this informal process…
An Informal Process (1)

• **Step 1**: Select a topic, form a team
  – To select a topic:
    • Check out *Previous Projects* web page
    • Check out *Project Ideas* web page
    • Talk to CS faculty & students
    • Talk to faculty & students in other depts
    • Look around you!
An Informal Process (1)

• **Step 1**: Select a topic, form a team (cont.)
  – To form a team:
    • Use ProjectFinder app (required)
    • Use Piazza (optional)
An Informal Process (2)

- **Step 2**: Choose a leader
  - Goal: *conceptual integrity* (Brooks)
    - Make app so coherent that it appears to be the product of a single mind
  - Implications:
    - Everyone has to pull together
    - Someone has to be in charge
  - *Successful software development projects are not democracies*
An Informal Process (3)

• **Step 3**: Define requirements
  – **Who** are the users?
  – **What** should the app do?
    • Decide what user need(s) the app will fulfill
    • Gather requirements
      – Conduct interviews, watch users work, …
    • Structure requirements:
      – Compose models of user world, use cases, story boards, …
      – **Involve the users!!!**
An Informal Process (4)

• **Step 4**: Design
  – **How** will the app work?
  – Step 4.1: Experiment
  – Step 4.2: Make pervasive design decisions
  – Step 4.3: Make buy vs. build decisions
  – Step 4.4: Compose module interfaces
An Informal Process (4.1)

• **Step 4.1: Experiment**
  – Do informal thinking and exploring early
    • So you have time to let ideas gel
  – Do many simple experiments early
    • So you learn what works or doesn’t
An Informal Process (4.2)

- **Step 4.2**: Make pervasive design decisions
  - User interface tier decisions
    - **Type**: Browser, desktop/laptop program, native Android program, native iOS program, ...
    - **Technologies**: HTML, CSS, JavaScript, AJAX, jQuery, AngularJS, React, Java, Objective-C, Swift, ...
An Informal Process (4.2)

**Step 4.2: Make pervasive design decisions**

- Processing tier decisions
  - **Language**: Java, Python, PHP, C, C++, C#, Perl, Ruby, JavaScript...
  - **Data comm format**: text, serialized objects, XML, JSON, …
  - **Dev framework**: Flask, Django, Spark, Spring, Rails, Android SDK, …
  - **Dev tool**: Eclipse, Visual Studio, Android Studio, XCode, …
  - **Hosting service**: your computer, CS Dept computer, OIT computer, Heroku, AWS, …
An Informal Process (4.2)

• **Step 4.2:** Make pervasive design decisions
  – Data management tier decisions
    • **Data store type:** flat files, spreadsheets, relational DB, NoSQL DB, …
    • **DBMS:** SQLite, MySQL, PostgreSQL, MongoDB, …
    • **Hosting services:** your computer, CS Dept computer, OIT computer, Heroku, AWS, …
Informal Process (4.3)

• **Step 4.3**: Make buy vs. build decisions
  – OK to use modules/code from elsewhere
    • E.g., copy or adapt open source
  – However:
    • Must identify what you have used, and where it came from
    • Overall project design must be your work
    • Selection and assembly of components must be your work
    • Most of the code must be your work
An Informal Process (4.4)

- **Step 4.4: Compose module interfaces**
  - Module = interface + implementation
  - **Interface**
    - The “public” part of a module
    - A module’s “advertisement” to clients
    - A module’s contract with clients
      - What are the inputs?
      - What are the outputs?
      - Who manages resources?
      - Who detects/reports errors?
An Informal Process (4.4)

• **Step 4.4**: Compose module interfaces
  – Hide design decisions behind interfaces
  – Common comment: “I wish we had done interfaces better”
  – Less common comment: “We thought hard about the interfaces so it was easy to make changes without breaking anything”
  – **Try to stay friendly!**
An Informal Process (5)

• **Step 5: Implement**
  – Compose module implementations
  – Involve all team members
    • Every team member, including the project leader, must do a significant part of the coding
An Informal Process (6)

• **Step 6: Test**
  – Does the app work as **you** intend?
  – Integrated with Implement step
    • Make sure it always works
    • Fix bugs before adding features
  – Additional distinct step at the end
An Informal Process (7)

• **Step 7: Evaluate**
  – Does the app work as its *users* intend?
  – Does the app fulfill the users’ needs?
  – Approaches:
    • Evaluation by users
    • Evaluation by experts (you!)
An Informal Process (8)

• **Step 8**: Document
  – Integrated with previous steps
  – Additional distinct step at the end
    • User’s guide, programmer’s guide, …
An Informal Process (General)

- Iterate
  - Do not build an app that requires a “big merge” where nothing works until everything works
  - Always be able to stop and declare success
  - Iterate between Implement and Test frequently
  - Revisit Requirements and Design less frequently
An Informal Process (General)

- Do *least-risk design*
  - Minimize risk
  - The module to develop next should be the one which, if problematic, will have the largest negative impact on the app as a whole
An Informal Process (General)

• Use version control system for all code
  – Git (or, by permission, some other VCS) is mandatory
An Informal Process (General)

• Plan time for “overhead” activities
  – **Changing your mind:** You will reverse decisions; you will redo work
  – **Disaster:** Deleted files, broken hardware, ...
  – **Sickness:** You will lose time for unavoidable reasons
  – **Health:** There is more to life than this project!
  – **Deliverables:** You must package your system for delivery…
Deliverables

• See *Project* web page for details
• See *Schedule* web page for due dates
• All deliverables are graded
Deliverables

• Pre-project deliverables:
  – Entry in *ProjectFinder Application*
    • Goal: Help you form a project team
    • Goal: Inform others what you’re doing
    • Add row and repeatedly update it to indicate your:
      – Technical interests, project interests, project name (blank if still looking), project descrip (blank if still looking)
  • [https://cos333.cs.Princeton.edu/ProjectFinder](https://cos333.cs.Princeton.edu/ProjectFinder)
Deliverables

• Pre-project deliverables:
  – Approval Meeting
    • Team meets with lead instructor, to be sure your project idea is OK
    • Team presents one reasonably firm consensus idea, not several vague ones
Deliverables

• Early-project deliverables:
  – *Team Directory*
    • On Google drive of team leader
  – *Project Overview* document
    • In Team Directory
    • Elevator speech, overview, requirements, functionality, design, milestones, risks
Deliverables

• Mid-project deliverables:
  – *Timeline* document
    • In Team Directory
    • Updated weekly
  – Weekly status meetings
    • Attendance is mandatory
    • Be prepared to describe what you accomplished, what you didn’t, what you plan to do next, anticipated risks
Deliverables

• Late-project deliverables:
  – Demonstration of prototype
    • During team meeting
  – Demonstration of alpha version
    • During team meeting
  – Demonstration of beta version
    • During team meeting
  – Presentation
    • Slides in Team Directory
Deliverables

• Late-project deliverables (cont.)
  – *User’s Guide* document
    • In Team Directory
    • Audience: (non-technical) end user
    • Description of *what your app does*
    • Probably the most important document
  – *Programmer’s Guide* document
    • In Team Directory
    • Audience: maintenance programmer
    • Description of *how your app works*
Deliverables

• Late-project deliverables (cont.)
  – *Product Evaluation* document
    • In Team Directory
    • Results of testing
      – Does the app work as you intend?
    • Results of formal evaluation
      – Does the app fulfill its users’ needs?
  – *Project Evaluation* document
    • In Team Directory
    • What went well? What did not go well? Surprises? Lessons learned? Advice to future teams?
Deliverables

• Late-project deliverables (cont.)
  – Source code
    • In Team Directory
  – The application
    • Give instructors a URL, installation instructions, a computer, a phone,…
Keys to Success

• Keys to success in COS 333:
  – Find a good project
  – Find good teammates