Introduction

CSE 403, Winter 2006
Software Engineering

http://www.cs.washington.edu/education/courses/403/06wi/

Readings and References

- Reading
  » *Rapid Development*, Steve McConnell
    • Chapter 1, Welcome to Rapid Development
    • Chapter 2, Rapid Development Strategy
    • Chapter 3, Classic Mistakes

- Other References
  » *everything* about this class is on the web
  » http://www.cs.washington.edu/education/courses/403/06wi/

Goals

- Develop a good understanding of the context in which software development takes place
- Learn practical ways to be productive within this context and gain some experience on development projects during the quarter
- Believe that the difficult task of efficient and effective software development can be an interesting and fun challenge, worthy of an entire career - *you gotta believe!*

LittleApp Context

- Many of us build small applications for our own use or the use of our friends
  » shell scripts, buttons and lights controllers, little simulators, web page builders, off-the-wall homework projects for next quarter, etc …
- Requirements are limited
  » probably owned by one person or at most two
- One developer
- One release (plus a few service packs …)
Advantages of LittleApp

- Great communication between customer and developer
  » clear picture of simple requirements
  » requirements can be pruned and grown in an instant with little follow-on impact
- Pretty good schedule adherence
  » dream it up at lunch, deliver it at midnight
- Simple to use, no later releases, one developer
  » you may get away with no documentation ...

Disadvantages of LittleApp

- The ideas that created it are probably fairly specific to the original user/developer
  » Everyone in the world is not a CSE major
  » Many people have great ideas about software for their knowledge domain that we would never think of on our own
- LittleApp is little!
  » Even Superman can only do so much in a day
- It's under-documented … a support nightmare

BigApp Context

- Potentially huge customer base
  » retail transactions, financial accounts, imbedded apps, office worker desktops, ...
  » The company doing the development takes on a big risk and spends big money in the hope of gaining a big reward (staying in business, expanding the business, …)
- Lots of customers and developers
- Long, complex, integrated schedule

Advantages of BigApp

- Lots of customers can mean that the product actually meets a widely felt need
  » Creating a successful product that is used by thousands or millions of people is very satisfying
- Lots of developers means that a larger skill set can be brought to bear on the problem
  » Working with experts in other fields can raise the overall product quality significantly, and it's fun
- Money. A half-ton of money can work miracles
Disadvantages of BigApp

- The customer is a many headed beast that is never satisfied
- Lots of developers means that communication is critical
  » commitments, personalities, changing cast
  » once you've said something, people go off and do things based on that - unwinding is very hard
  » management, staff, factory, supplier, …
- Money. Big money makes people act weird

Productivity - processes and tools

- There are lots of techniques and tools that can help manage some of the chaos that is part of a BigApp project
  » clearly stated objectives and definite schedule
  » motivated teams with clear responsibilities
  » good support for communication
    • features, bugs, clarifications, meetings, schedules
  » solid development tools and recommended practices
    • editors, compilers, source control, bug tracking, build management, test suites, simulators, etc, etc

BigApp Development

- BigApp system development is a social activity
  » groups of people can do amazing things together
  » individuals do all sorts of unexpected things along the way - expect the unexpected
  » Focus and communicate
  » Use the tools but don't expect miracles from them
    • a skilled craftsman knows his tools and their limitations
It's a challenge - enjoy it!

• *Every* project has its ups and downs
• *Every* project has weird requirements, too little time, bizarre management decisions, blockheaded coworkers, disappointing suppliers, rewards and glory for the wrong people, and generally miserable days
  » so don't be surprised or upset
• *Every* project has the potential for major satisfaction - enjoy it where you find it!

Our projects

• The projects for this class will be based on a client / server architecture
• Project teams will
  » Define the specific functions of the applications
    • web services, API, client functionality
  » Develop the code that runs on the server side
    • single service, collate multiple services, ...
  » Develop the code that runs on the client side
    • browser integrated, standalone app, multi-target, ...

Project Components

![Diagram of network components](drawing is from IBM Web Services Handbook, http://www.redbooks.ibm.com/)