Meadan: Online Communication
Project post-mortem

15-413: Introduction to Software Engineering
Team 2:
Douglas Applegate
Matthew Douglass-Riley
Savina Imrhan
Matthew Williamson

Introduction to Meadan

- Created to provide a way for people from the Islamic and Western worlds to communicate
- Meant to create a global dialogue to promote cultural understanding
- Our role: proof of concept
  - Design a fully integrated website from which a user can launch the instant messenger client
Our Goals

- Set up testing server
- Follow XP (to extent possible)
- Meet with client twice each iteration
- Spend six hours, per person, per week (on average)
- Implement a particular subset of our initial group of user stories

User Story Goals

- Setup server
- Create and manage user logins and profiles
- Search user profiles
- Create integrated website with an IM launching from profile
- Bilingual IM, translation and language selection
Architecture

SIMP Server
- Applet
- Login & Chat

MySQL Server
- Login Requests (Web Page)

End User

Apache Tomcat Web Server (JSP)
- Login Requests (SIMP Client)

Web Requests

Demonstration
Requirements Changes

- Anti-requirements creep
- Original plan
  - Website
  - Expand SIMP for translation
  - End to end user experience
- Client shifted focus
  - Website
  - Integrating SIMP into website
The Picture of Success

How did we stack up?

- Follow XP programming practices
- Meet with client twice each iteration
- Adjust workloads for changing schedules
- Implement at least:
  - Set up testing server
  - Manage user login/logout & new users
  - Embed direct IM
  - Create & manage user profiles
  - Search user profiles
  - Bilingual IM conversations
  - User selected languages

The Good …

- Extreme programming
  - Pair programming
  - Unit tests
  - Refactoring eXtremely mercilessly
  - Biweekly client meetings
- Project
  - Adapted to changes in requirements
  - Produced documentation for legacy code
  - Achieved final set of goals provided to us
... the Bad ...

- Communication problems
  - No promiscuous programming
  - Coordination and proximity
  - Suboptimal communication between groups
    - Sometimes hard to divide work
    - Sometimes lost track of progress of other group
  - Motivation problems
    - Entire project as a prototype
    - Felt like we were unproductive

... and the Ugly?

- Spent too much time on overhead
  - 15% of all our time spent on server setup
    - Spent time on development platform
    - Technology integration
      - MySQL, Tomcat, Servlets, Applet, SIMP Server
  - 35% of all our time spent understanding SIMP
    - 278 Class files
    - 1.8 MB of source code
    - Very poor documentation
    - Still do not understand bulk of SIMP
    - Added or modified only 50 lines of code
Lessons Learned

- XP did not work well for our project
  - Designed for workplace
    - We live on opposite corners of campus
    - We have extremely varying schedules
    - We have small, unpredictable blocks of free time
  - Metrics misleading
    - Constant load factor does not always imply progress
  - Designed for coding… not hacking
    - Entire project had prototype feel
  - Modus tollens:
    - XP $\Rightarrow$ (Us Happy)
    - $\neg$ (Us Happy)
    - $\therefore \neg$ XP

Conclusions

- We delivered the goals our client provided
- Project not what we expected, but we adapted
  - Solved systems problems, not software problems
- Have set up a launching pad for next group to develop more
Questions?