Grading
Project Grades

• We’d like your grade to reflect your contribution and commitments to the project and project team

• We’ll use the Assessment of Contributions of Group Members (from the Center for Engineering Learning and Teaching – see handout)
Here’s how it works

• Three assessment points
  o After SRS (instructional, informational only, wk3)
  o Around Beta Release (week 6)
  o After Final Release (week 10)

• You have 100 points to allocate among your team members. No two members may receive the same amount of points.

  ~ Weighted ranking of contribution ~
How the grade is computed

Project grade $\times \min((\text{sum of points}), 100)$

$+ [\text{staff input, self review}] = \text{Student grade}$

Example: Betty

The whole project team thought Betty (as well as most of the team!) did a great job. She took initiative, delivered on time and worked well with others. They awarded her: 20, 20, 21, 19, 20 = 100

Project grade was 95%.

Betty’s grade was $95\% \times 100 + [ ] = 95\%$
Example 2: Jughead’s heart was in the right place but often got sidetracked by the kitchen. He delivered half of what he originally committed to, and was notorious for missing meetings and not keeping teammates well informed.

Jughead’s values were: 15, 11, 14, 10, 12 = 62

Project grade was 95%.

Jughead’s grade was 95% x 62 + [ ] = 60%
Miscellaneous

- Grades are posted in the CSE grade database.

- We will entertain questions about grades only for 1 week after they are posted in the grade db. *Check for missing assignments.*

- Questions about *assignment grades* should be written up and submitted to the staff via email.

- *Summary* questions can be discussed directly with the TA’s.
Software Development Teams

Talent wins games, but teamwork and intelligence wins championships.

Michael Jordan
Team exercise

Part of being successful is having a shared vision. This applies to team operation, too.

For your 403 team:
What are the top 5 responsibilities of a team member? Write in terms of, “I will ....”
Team responsibilities …

- Spring 08 thoughts: I will
  - Show up to meetings on time
  - Contribute to the meetings
  - Be prepared for the meetings
  - I’ll hold up to my project development commitments and meet my deadlines
  - Communicate quickly and clearly any problems or conflicts, and what I’m doing
  - Be ambitious in the workload I take on
  - Try to be self aware of how you’re affecting others
  - Open to opinions of others
Team member responsibilities

- From Sp07
  - I will do my work on time or give sufficient notice (backed by some personal contact)
  - I will value others' opinions
  - I will collaborate with all teammates for making major decisions
  - I will do what I say I will
  - I will not start fights
  - I will respond constructively to feedback
  - I will attend all meetings, including one on the weekend, and bring refreshments to some meetings
  - I will not hesitate to voice my opinion
  - I will stand behind and support team decisions
  - I will respond to email within the day it was sent
Team Exercise – 5 minutes

- Discuss your team responsibilities
- By Monday, agree on and post on your wiki, your top 5 team member responsibilities

Will your team be as successful as the early Microsoft team??
Requirements

Pragmatic Programmer Tip: Don’t Gather Requirements – Dig for them

Requirements rarely lie on the surface. They’re buried deep beneath layers of assumptions, misconceptions, and politics
Readings

- Requirements and Use cases:
  - Writing Effective Use Cases, Alistair Cockburn
  - Pragmatic Programmer, p. 202-208

- Survival Guide Chapter 8: Requirements Development
Outline

- What are requirements?
- Some interesting requirements facts
- How can we gather requirements?
- How can we specify requirements?
Software requirements

Requirements specify what to build
- tell "what" and not "how"
- tell the problem, not the solution

Some goals of doing requirements:
- understand precisely what is required of the software
- communicate this understanding precisely to all development parties
- control production to ensure that system meets specs (including changes)
General classes of requirements

Spring 08:
- performance
- reliability
- privacy, security
- functionality
- documentation
- ease of use
General classes of requirements

Examples requirements types:
- Feature set
- GUI
- Performance
- Reliability
- Expansibility (ie. support plug ins)
- Environment operates in (ie. HW, OS, browsers)
- Schedule
How do we gather requirements?

Let’s start with two facts:

- Standish group survey of over 8000 projects, the number one reason that projects succeed is user involvement.
- Easy access to end users is one of three critical success factors in rapid-development projects (McConnell).
How do we gather requirements?

Benefits of working with customers:
- Good relations improve development speed
- Improves perceived development speed
- They don’t always know what they want
- They do know what they want, and it changes over time
The most difficult part of requirements gathering is not the act of recording what the users want; it is the exploratory, development activity of helping users figure out what they want.

McConnell, SG
Work with a User to Think Like a User – it’s the best way to get insight on how the system is easily used

Pragmatic Programmer Tip
Throughout your travels with the customer, be sure to set reasonable customer expectations.
I'll need to know your requirements before I start to design the software.

First of all, what are you trying to accomplish?

I'm trying to make you design my software.

I mean what are you trying to accomplish with the software?

I won't know what I can accomplish until you tell me what the software can do.

Try to get this concept through your thick skull: the software can do whatever I design it to do!

Can you design it to tell you my requirements?
How can we specify requirements?

So… we’re working with the customer to understand their needs, how do we capture these requirements?

Possibilities include:
• Prototype
• **System Requirements Specification Document**
  • Use Cases
  • Feature List
  • Paper UI prototype