Today: Part II
CSE403

• What is our approach?
• What is the structure and expected work?
• What can a 10 week project do and not do?
• ...

Today: Part I
Software engineering

• What is it?
• Is it engineering?
• Is it the same as programming?
• Is it challenging? If so, why?
• Is it important? If so, why?
• ...

Your first task
List 0-5 preferred teammates on Catalyst WebQ
(open after class until 5PM Wednesday)
Final projects teams set at section on Thursday

David Notkin • Spring 2010

Part IA

1. What is software engineering?
2. Is it engineering?
3. Is it the same as programming?

What do you think?

My partial responses

1. What is it?
   a. Software from womb to tomb
   b. ...the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software [IEEE]

2. Is it engineering? Yes ... at least to some degree
   a. In this sense of “designing and constructing ... works of public utility” [OED] or that we “make [software] machines to serve useful purposes in the world” [Michael Jackson]
   b. Doesn’t – yet, or perhaps ever – look that similar to traditional engineering disciplines

3. Is it the same as programming? No
   a. Building the right software system is as important as building the software system right
   b. Scale causes the problems to qualitatively differ

A standard SE introduction

• Software projects are too expensive and cancelled too often
• Software quality is appalling
• ...

• Make Software Engineering a real engineering discipline
• Define strong mathematical basis, standard of practice, etc.
• ...

• Define and adhere to a standard lifecycle
• State requirements, design, etc. precisely and rigorously
• ...
Appalling quality and cost: examples

- Zune leap year bug
- Mars Polar Lander crash
- ...

Your favorite examples?

Notkin’s responses

Software Crisis
- For the AIDS crisis, we’d like to eliminate AIDS
- For the Cuban missile crisis, we wanted to eliminate missiles in Cuba
- For the sub-prime loan crisis, we’d like to eliminate the effect of sub-prime loans
- If software is a crisis, give us another technology like it!

Engineering
- Many of the goals and steps toward Software Engineering are reasonable
- Many of the analogies to traditional engineering disciplines are flawed
- We need to improve our ability to efficiently produce high-quality software, but these analogies distract rather than enhance

Process
- Process and methods are good, when they are consistent with the problem to be solved
- No process or method is appropriate in all contexts to help solve all problems
- Many processes and methods, in pursuit of Software Engineering, seem to wish to take the “soft” out of software

An alternative introduction

Software is a powerful vehicle for building machines, or parts of machines, that benefit society
- Software’s importance has grown wildly over the past 50 years
- Software’s malleability is its strength – it can be molded in incredible forms
- Software systems are intertwined in society in amazing ways; this is a testament to the power of software and the ingenuity of software engineers

Software is challenging to engineer

- Discrete nature of software
- Scale and complexity of software – even given abstraction
- Ability to adapt software – and subsequent pressures to do so
- Astonishing demand for software
- Exceedingly rapid changes in the underlying technologies
- Frequent lack of clarity about requirements
- Communication among teams can be difficult
- ...
So, what about 403?

Most of you likely rationally understand the distinctions between programming and software engineering.

Experience, however, shows that few of you are likely to understand the distinctions viscerally.

Thus, our primary vehicle for the course is a group project — groups of six who take a high-level specification through implementation.

The overarching intent of the project is to spread this understanding from your neurons to your viscera.

What's a 10 week project to do?

Can approximate:
- Ill-defined requirements
- Time-pressure
- Teamwork
- Different team roles
- Control over design
- …

Can’t approximate:
- Global, distributed teams
- Full womb-to-tomb
- Pressure from competitors
- Feedback from real users
- …

Not a mere matter of programming!

Your biggest challenges are to define an appropriate scope for the project and to structure your team, your process, and your product to allow for planned and unplanned adjustments.

Each team: select from two project choices

- There are, intentionally, many open questions in terms of function and user interface
- I value a working system that does less over a non-working system that potentially does more
- I value a system that reflects realism over unrealistic conceptual beauty — but this is a tough line to toe

- Computer science co-authorship network explorer
- Software clone detector

Option 1: Co-authorship network explorer

Our field, like many, publishes research results in journal papers, conference papers, books, etc. Some of my papers:

- Who has co-authored with whom?
- Is there a co-authorship chain from Notkin to Brooks?
- Do UW CSE members co-author together more often than UCSD CSE members?
- …
- How is the information found, entered, queried and visualized?
- …
Option 2: Software clone detector

- “Clones are segments of code that are similar according to some definition of similarity.” —Ira Baxter, 2002
- Studies show at least 5-20% of code in software systems are clones, often created from copy-and-paste
- They are generally thought to be bad — although this is questionable to some degree

```java
3/29/2010
CSE403 Sp10

int x = 0;

void foo(Iterator iter)
{
    for (item = first(iter); hasMore(iter); item = next(iter))
        x = x + value(item);
}
}
```

Projects and software

- Each group about six people
- Form groups primarily on your own
  - Use WebQ on Catalyst to submit preferred teammates
  - I reserve the right to make adjustments as needed
- Beware: an inability to find consistent and common group meeting times is a huge minus
- Software available on CSE machines — beyond this, you are responsible for installation, etc.

Pitfalls to watch for include...

- A slow start
- Insufficient team meeting time
- Choosing project software solely because you want to learn it
- Ignoring the importance of understanding the domain
- Extended differences of opinion about alternative decisions
- “Super-programmers” who try to take over and make it a “mere matter of programming”
- Too much/too little time getting tools to work
- Too much/too little focus on documentation
- Isolating or marginalizing one or more team members
- Overly high expectations about what is achievable
- Nothing works unless everything works

```sql
CSE403 Sp10

4/03
```

Project Readings Lecture Sections

- Administrivia: see web page
  - Grading [subject to change]
    - Two exams: 12.5% each
    - Class participation: 5%
    - Project: 70%
  - TAs — introduce thyselfs!
- Notkin travel: ICSE 2010 (Cape Town, Apr 20, May 3-5-7) NCWIT (Portland, May 19) — too much, very sorry.
Questions?

Your first task:
List 0-5 preferred teammates on Catalyst WebQ
(open after class until 5PM Wednesday)
Don't forget!

Final project teams set at section on Thursday