Software Engineering

David Socha
UrbanSim Project
socha@cs.washington.edu
What is software engineering?
What is software engineering?

• One view:
  – Social engineering
    • A user / organizational / social intervention (usually requested)
Software as an intervention

- Customer is asking you to change how they do “business”
  - means changing user’s behavior
  - means the users are an integral part of the system you are designing
  - means you cannot understand your system until your users are using the system
Why can’t you understand the system until users are using it?

• Our history shows this
  – e.g.: Historically, nearly 85% of software project fail

• We are working with complex adaptive systems
Three rules:
1. Each individual shall steer toward the average position of its neighbors.
2. Each individual shall adjust its speed to match its neighbors.
3. Each individual shall endeavor not to bump into anything.

A Complex Adaptive System…

Boids

You cannot accurately predict it.
Your end-user cannot accurately predict it.

Now what?
How do you work effectively given you can’t predict well?

• Empirically
Questions?
What does “empirically” mean for software engineering?
What does “empirically” mean for software engineering?

- Focus on the system, not just the software
How do people change?
How do people change?

• #1: Via small changes
  – So, use small steps and gather feedback, e.g.:

developers + SCRUM

What type of feedback?
What type of feedback?

- **Video, not words**
  - Video them
  - Development team watches video
  - Ideation
  - Pick low hanging fruit
  - Build
  - Deliver / Intervene
  - Repeat
How do people change?

• #2: By wanting to
  – So, create a path so seductive that they cannot resist following it
How collaborate with users?

• Understand them, including their domain
• Use a language they understand

Which language?

What is the easiest path?
Use the language your users already know

• Their current mental model
  – So, put their mental model into software

  Nouns & verbs → classes & methods

  Model-Driven Development
How get people to actually use your software?
How get people to actually use your software?

• Give them something that makes their life easier **RIGHT NOW**
  – So, build only what they will use **RIGHT NOW**
  – Only plan enough to do the work

• Don’t force them to change too much at once
  – So, deliver small pieces

Don’t need MS Project!
Where do some of the best ideas come from?
Where do some of the best ideas come from?

• Teams with diversity
• The users - don’t underestimate what they can create
  – So, collaborate
  – Create extensible / adaptable system
What are some qualities of great designs?
What are some qualities of great designs?

- As simple as possible, and no simpler
- Solves more problems than originally intended
- Sustainable

How learn design?
Where can we learn more about good designs?

• Study existing designs (via open source)

• Read books on software design & development

• Practice designing
Where can we learn more about good designs?

- Study existing designs (via open source)
- Read books on software design & development
- Practice designing

How about GREAT design?
Where can we learn more about great designs?

- Biomimicry
Where can we learn more about good designs?

- Biomimicry
Where can we learn more about good designs?

• Biomimicry

The theory of the owl's wing is adopted to this part.
Summary

• Software engineering
  – Is an intervention done for an emergent system, by an emergent system
  – Includes design, all the time
  – Requires empirical processes and techniques, including
    • User observation
    • Lots of iteration with feedback
    • Bags of tricks

• We are all designers
Questions?

socha@cs.washington.edu

www.urbansim.org