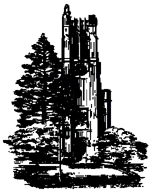
MICHIGAN STATE UNIVERSITY

10. Prototyping



CSE 498, Collaborative Design

Wayne Dyksen
Brian Loomis

Department of Computer Science and Engineering
Michigan State University

Spring 2005

Overview

- Why do we do this?
- How quick should we be?
- What is a prototype?
- Can we use this for the start of the project?

Prototype

“A rapid implementation of a subset of a project’s requirements, built early in the development cycle to resolve unknowns about the final architecture, and usually not given to the customer as a finished product.”

1. A prototype is optional
2. A prototype may reduce risk on a project
3. Prototypes are not finished goods

Why do a prototype?

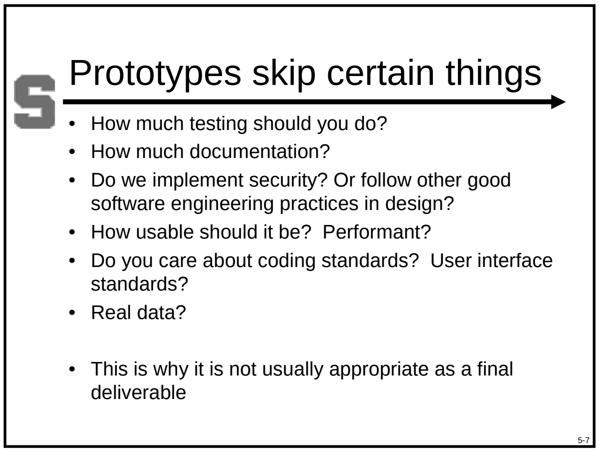
- Answer a question about the app
  - Does the user want this or that feature? (feedback)
  - Can we build a foosit that works with a widget?
  - What makes the application more usable?
  - What is the market for an idea?
- Determine schedule more accurately
  - If we have to build 100 screens and don’t know how long 1 will take…
  - How will I deploy the app to all corporate employees?
- Reduce risk on the app
  - If we are using new technologies that we’ve never used before, how do we know they’ll play together well? (feasibility)

Speed is critical

- Build it fast – how fast?
  - 2-3 day tasks
  - Don’t assume things that are not true
  - Make sure you’re answering the questions
  - Don’t be afraid to break things
- Stop prototyping when you know the answers

Tools are critical

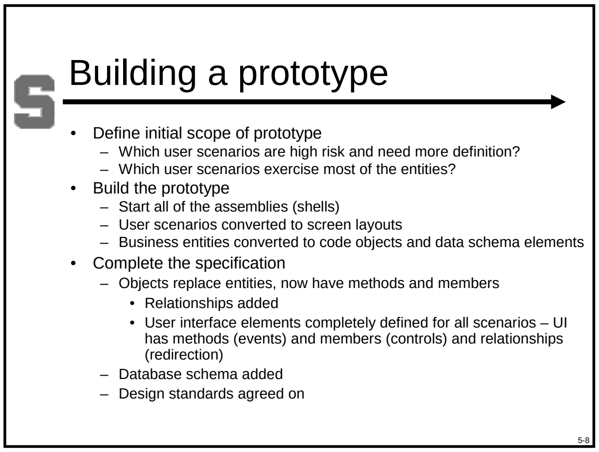
- Make sure you’re using:
  - RAD languages, IDEs
  - Design tools and wizards
- Typical output is a functional system
  - But it may be fragile
  - It should “demo” well to get the point across
  - Does not need the full build environment
- If you make certain tradeoffs for speed of development, you may or may not be able to use the prototype as a basis for the final system



Prototypes skip certain things

- How much testing should you do?
- How much documentation?
- Do we implement security? Or follow other good software engineering practices in design?
- How usable should it be? Performant?
- Do you care about coding standards? User interface standards?
- Real data?
- This is why it is not usually appropriate as a final deliverable

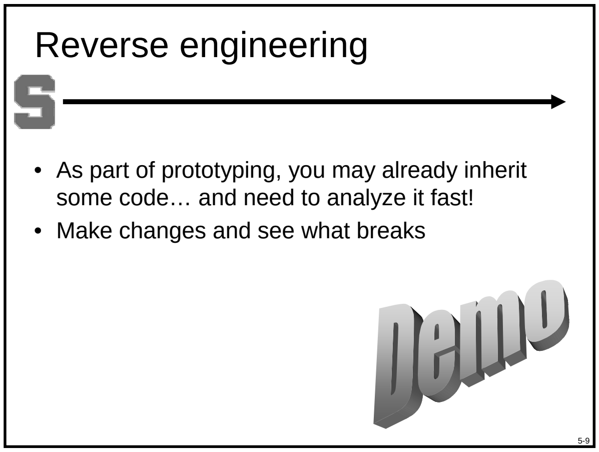
5-7



Building a prototype

- Define initial scope of prototype
  - Which user scenarios are high risk and need more definition?
  - Which user scenarios exercise most of the entities?
- Build the prototype
  - Start all of the assemblies (shells)
  - User scenarios converted to screen layouts
  - Business entities converted to code objects and data schema elements
- Complete the specification
  - Objects replace entities, now have methods and members
    - Relationships added
    - User interface elements completely defined for all scenarios – UI has methods (events) and members (controls) and relationships (redirection)
  - Database schema added
  - Design standards agreed on

5-8

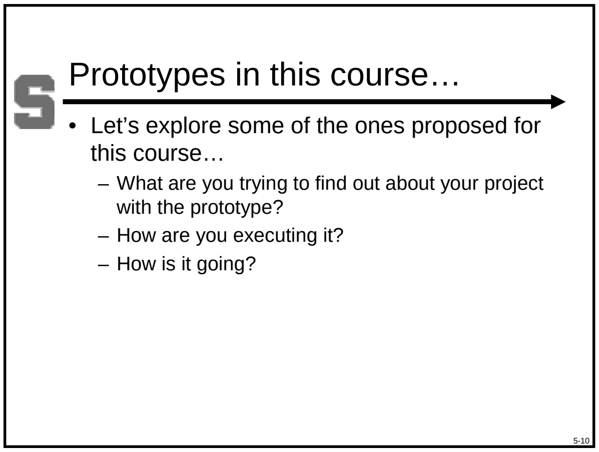


Reverse engineering

- As part of prototyping, you may already inherit some code… and need to analyze it fast!
- Make changes and see what breaks

Demo

5-9



Prototypes in this course…

- Let's explore some of the ones proposed for this course…
  - What are you trying to find out about your project with the prototype?
  - How are you executing it?
  - How is it going?

5-10