CSci 8801
Advanced Software Engineering

Mats Heimdahl
Professor, Director of UMSEC
Critical Systems Research Group
Department of Computer Science and Engineering
University of Minnesota
4-192 EE/CS; 200 Union Street SE
Minneapolis, MN 55455

Topics for Today

• Course overview
  ▪ Instructor and teaching model
  ▪ Introduction
  ▪ What you should already know
• What to expect
  ▪ Assignments
• Questions
Professor

• Professor Mats Heimdahl
  ♦ Office: EE/CS 6-201
  ♦ Email: heimdahl@cs.umn.edu
  ♦ Office hour: Tuesday and Thursday 2:30-3:30

• Class information
  ♦ Class web page available from Professor Heimdahl’s web page
  ♦ www.cs.umn.edu/~heimdahl/csci8801

Communicating with Instructor

• Problems with homework and assignments
  ♦ Contact Dr. Heimdahl
  ♦ Contact each other

• Problem with Dr. Heimdahl
  ♦ Contact CS front office and they will help you
Prerequisites

- You ought to have an introductory software engineering class
  - CSci 4081, CSci 5801, or old CSci 5180
- Some background in discrete mathematics
- Interest in learning things that are more advanced than what is industry practice
  - What you learn here you may not be able to use when you get a job
  - But, you can be part of changing the world

Learning Modes

- Lecture
- Textbooks
- Class Discussions and Class Presentations
- Homework
Assignments and Grading

• Homework assignment
• Student presentations
• Semester projects
  ◆ A research proposal

Syllabus and Schedule

• Review syllabus

• Review schedule
  ◆ Schedule is still under development
  ◆ Topics will just take the time it takes
Lecture Plan

- Requirements engineering  
  - 2 weeks
- Formal modeling  
  - X weeks
- Analysis techniques  
  - Y weeks
- Testing  
  - Z weeks

Questions?

Introduce yourselves
Domain of Concern

How we Develop Software

Concept
Formation

Requirements
Specification

Design

Implementation

Object Code

Analysis

Test

http://www.umsec.umn.edu
Model-Based Development

Visualization → Properties → Analysis → Testing → Prototyping

Code → Specification Model

Model-Based Development Tools

- Commercial Products
  - Esterel Studio and SCADE Studio from Esterel Technologies
  - SpecTRM from Safeware Engineering
  - Rhapsody from I-Logix
  - Simulink and Stateflow from Mathworks Inc.
  - Rose Real-Time from Rational
  - Etc. Etc.
Research Tools (many):
RSML-e and Nimbus

Simulations of environment
RSML-e Formal Models (~20 running concurrently)

How we Will Develop Software

Concept Formation

Requirements

Prop

System Test

Analysis

Specification/Test

Integration

Implementation
ROI with Model Based Development

Source: Esterel Technologies

A Simplified Development Model
Benefits of SCADE™: From V to Y Cycle

Manual coding
Use of a “regular” automatic code generator
Use of the qualifiable code generator as a verification tool
Use of the qualifiable code generator as a development tool
Use of proof technology

Source: Esterel Technologies

Ongoing Research

RSML-e, SCR, SpecTRM, Statecharts, Esterel, SCADE, Simulink, etc.
CMU, SRI, Stanford, UC Berkeley, VERIMAG, NASA, etc., etc.
Minnesota, Pennsylvania, George Mason, NRL, NASA Ames, etc.
RSML-e, SCR, SpecTRM, Statecharts, Esterel, SCADE, Simulink, etc., etc., UML

Properties
Visualization
Analysis
Testing
Prototyping
Specifications
Model
Code

Proof carrying code, Provably correct compilers, Test for correctness
RSML-e, SCR, SpecTRM, Statecharts, Esterel, SCADE, Simulink, etc., etc., UML
Problems…

- Can we trust execution environment?
- Trust the results?
- Can we trust execution environment?
- Can we really trust the code?
- Are the languages usable—syntax and semantics?
- Can they play nice together?

Perfection is **Not** Necessary

- Tools and models only need to be better than manual processes…
- **How do we demonstrate this?**
  - Empirical studies are of great importance

I think many already are
We Have Learned

• What CSci 8801 is all about (sort of)
• What is expected from you
  ♦ Prerequisites
• A quick view of model-based development

• Next time
  ♦ SE Fundamentals