CS6610: Software Engineering II

Course Overview
Software is ...

...one of the most complex man made artifacts

“I believe the [spreadsheet product] I'm working on now is far more complex than a 747 (jumbo jet airliner)”
-- Chris Peters (Microsoft, 1992)

“It’s different [from other engineering disciplines] in that we take on novel tasks every time. The number of times [civil engineers] make mistakes is very small. And at first you think, what’s wrong with us? It’s because it’s like we’re building the first skyscraper every time.”
-- Bill Gates (Microsoft, 1992)
Software is ...

...one of the most complex man made artifacts

Microsoft Word is ...1 million lines of code
Microsoft NT ...16 million lines of code

...but perhaps software complexity shouldn’t even be measured in terms of lines of code, but instead, in terms of number of states
States

- The size of a system is sometimes more accurately expressed using a semantic point of view:
  - the number of different *states* a system can reach
  - ... an integer has 4.2 billion possible values
  - ... an object with 2 ints and a boolean field has 40 thousand quadrillion values

- How about Windows Vista?
Software is...

...critical to the conduct of modern life.

- Process Control (oil, gas, water, ...)
- Transportation (air traffic control, ...)
- Health Care (patient monitoring, device control ...)
- Finance (automatic trading, bank security ...)
- Defense (intelligence, weapons control, ...)
- Manufacturing (precision milling, assembly, ...)

*Failing software costs money and lives!*
Software is ...

...becoming the dominant component of society’s infrastructure.

In the future...

- Everything will be monitored/controlled
  - networked watches, clothes, ...
  - autonomous vehicles, intelligent highways, ...
  - virtual X rather than physical X

- Failures will be very costly and dangerous
Priorities are changing ...

From: Bill Gates
Sent: Tuesday, January 15, 2002 2:22 PM
To: Microsoft and Subsidiaries: All FTE
Subject: Trustworthy computing

... Trustworthy Computing is computing that is as available, reliable and secure as electricity, water services and telephony.

... In the past, we’ve made our software and services more compelling for users by adding new features and functionality.

... We’ve done a terrific job at that, but all those great features won’t matter unless customers trust our software. So now, when we face a choice between adding features and resolving security issues, we need to choose security.

... These principles should apply at every stage of the development cycle of every kind of software we create.

... Bill
Software is ...

...what you’ll be building after graduation.

- You’ll be developing systems in 2020+
  - in the context we just mentioned
- Given the importance of software
  - you may be regulated, licensed
  - you may be liable for errors
  - your job may depend on your ability to produce reliable systems
Software Development Cycle

Requirements Analysis

Design

Code and Unit Test

Subsystem Test

System Test
Current Software Development Methods Are Insufficient

- Testing
  - samples execution behavior, misses some
- Systematic Inspections
  - don’t scale very well, although they are thorough
- Rigorous development processes
  - helping but most organizations don’t apply them

*Formal methods are becoming more popular*
Famous Picture

Windows

An exception 06 has occurred at 0028:C11B3ADC in \x\D DiskTSD(03) + 00001660. This was called from 0028:C11B40C8 in \x\D voltrack(04) + 00000000. It may be possible to continue normally.

* Press any key to attempt to continue.
* Press CTRL+ALT+RESET to restart your computer. You will lose any unsaved information in all applications.

Press any key to continue
Ariane 5 (1996)

- Ariane 5 used software used prior in Ariane 4
- 64-bit floating-point to 16-bit integer generated conversion an overflow
- Error was caught, sub-system shut down
- Back-up systems failed for same reason.
- Rocket veered off course.
- Control system decided to abort mission.
- Result: Rocket self-destructed
- Cost: $400 million payload
Pentium (1994)

- First release of Intel Pentium chip
- Mistakes when dividing floating-point numbers that occur within a specific range
- Estimated 3 million to 5 million defective chips
- Cost: $475 million
Mars Climate Orbiter (1999)

- One development team used pound/second in their code while the other used Newton/second
- Values passed from one module to another without conversion
- Result: Loss of the craft
- Cost: $125 million
Failing Software Costs Lives

• Potential problems are obvious:
  – Software used to control nuclear power plants
  – Air-traffic control systems
  – Spacecraft launch vehicle control
  – ....
11. EXCLUSION OF INCIDENTAL, CONSEQUENTIAL AND CERTAIN OTHER DAMAGES. TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, IN NO EVENT SHALL MICROSOFT OR ITS SUPPLIERS BE LIABLE FOR ANY SPECIAL, INCIDENTAL, INDIRECT, OR CONSEQUENTIAL DAMAGES WHATSOEVER (INCLUDING, BUT NOT LIMITED TO, DAMAGES FOR LOSS OF PROFITS OR CONFIDENTIAL OR OTHER INFORMATION, FOR BUSINESS INTERRUPTION, FOR PERSONAL INJURY, FOR LOSS OF PRIVACY, FOR FAILURE TO MEET ANY DUTY INCLUDING OF GOOD FAITH OR OF REASONABLE CARE, FOR NEGLIGENCE, AND FOR ANY OTHER PECUNIARY OR OTHER LOSS WHATSOEVER) ARISING OUT OF OR IN ANY WAY RELATED TO THE USE OF OR INABILITY TO USE THE SOFTWARE PRODUCT, THE PROVISION OF OR FAILURE TO PROVIDE SUPPORT SERVICES, OR OTHERWISE UNDER OR IN CONNECTION WITH ANY PROVISION OF THIS EULA, EVEN IN THE EVENT OF THE FAULT, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY, BREACH OF CONTRACT OR BREACH OF WARRANTY OF MICROSOFT OR ANY SUPPLIER, AND EVEN IF MICROSOFT OR ANY SUPPLIER HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
A Formal Specification is...

The expression in some **formal language** and at some level of **abstraction** of a collection of **properties** that some system should **satisfy**

[van Lamsweerde]

- **Formal language**
  - syntax can be mechanically checked for correctness

- **Abstraction**
  - above the level of source code (e.g., C is a formal language)
  - many useful levels of abstraction
A Formal Specification is...

The expression in some formal language and at some level of abstraction of a collection of properties that some system should satisfy [van Lamsweerde]

- **Properties**
  - often expressed in some logic, but not necessarily
  - must have a well-defined semantics to be useful

- **Satisfaction**
  - generally, should be able to tell if system satisfies the specification
  - ideally (but not usually), satisfaction is decided mechanically
Why Use Formal Methods?

A number of reasons (see [van Lamsweerde])

- Forces you to think about issues in a systematic way
  - Leads to better design
  - Earlier detection of inconsistencies and flaws
- A precise reference to see if requirements are satisfied
- Gives direction to latter development phases (leading to coding)
- Provide a basis for reuse via specification matching
- Precise documentation within a team of developers
- ...
Possible Development Cycle

Requirements → Design → Code
In this course ...

- You will study specification languages
  - **Alloy**: high-level semantic design
- You will use tools for each language
  - to check syntax, simulate, and verify semantic properties
- You will learn verification data structures and algorithms
Formal Methods in Cycle

Requirements

Design

Code

Alloy

ESC/Java
Summary

- Software is becoming pervasive and very complex
- Current development techniques are inadequate
- Formal methods are not a panacea, but we believe they will be necessary
- We will learn to use several different formal methods
Who am I?

• Zijiang (James) Yang
  – Ph.D., UPenn 2003
  – 4 years experience in industry
  – Main research areas: Software Verification
  – E-mail: zijiang.yang@wmich.edu
Who are you?

- I study x
- I am interested in learning y
- My background is in z
- I have done project x
- I have heard x
- I have worked for company x
Acknowledgements

- The slides are adapted from CS771 at KSU.
- Slides 17 and 18 showing Alloy examples are adapted from Daniel Jackson’s slides used in a talk given at Kansas State in Spring 2000.
- The JML examples are taken from the examples distributed with ESC/Java.
Big break

- Class on 1/10 cancelled
  - Business trip to Washington D.C.
- No class on 1/15
- See you on 1/17