Alloy: A Language for Modeling and Analyzing Software Systems

Eunsuk Kang
Software Design Group
MIT
Software Design Group

**Director**
Prof. Daniel Jackson, CSAIL

**Research Topics**
Software reliability, testing, verification, safety-critical systems, security

**Collaborators**
AT&T, NASA, IBM, Nokia, Northrop Grumman, Telcordia, Massachusetts General Hospital

**Industrial Case Studies**
Radiation therapy machine, mobile networks, biometric systems, electronic voting, network configurations, web applications, etc.
Designing Software
Design Sketches
Design Documents
Need a Better Approach?

**Sketches**
Lightweight, informal
Good for brainstorming, prototyping
Need a Better Approach?

**Sketches**
Lightweight, informal
Good for brainstorming, prototyping

**Documents**
Completeness, heavyweight
Still informal, not analyzable
Need a Better Approach?

**Sketches**
Lightweight, informal
Good for brainstorming, prototyping

**Documents**
Completeness, heavyweight
Still informal, not analyzable

**Can’t ask:**
Is design consistent?
Does design satisfy property X?
Does code conform to design?
Desiderata

Simple, lightweight language
Easy to learn, low burden
Desiderata

Simple, lightweight language
Easy to learn, low burden
Precise, analyzable
Ask questions
Desiderata

Simple, lightweight language
Easy to learn, low burden
Precise, analyzable
Ask questions
Instant feedback & animation
Support rapid prototyping
Demo: Simple File System in Alloy
Features of Alloy

**Syntax**
A small number of constructs
Declarative, with relational operators
Subtyping, module systems
Features of Alloy

Syntax
A small number of constructs
Declarative, with relational operators
Subtyping, module systems

Semantics
Pure first-order logic
No built-in idiom, flexible
Features of Alloy

**Syntax**
A small number of constructs  
Declarative, with relational operators  
Subtyping, module systems

**Semantics**
Pure first-order logic  
No built-in idiom, flexible

**Analysis**
Assertion & simulation  
By translation to a constrain solving engine
Testing vs. Analysis

**Testing**
- Sparse behaviors
- Need manual test cases
- Can’t show absence of bugs

**Alloy Analyzer**
- Exhaustive
- Fully automatic
- Verifies “deep” properties
Applications
Radiation Therapy System

Massachusetts General Hospital, Boston
Safety Analysis using Alloy

Beam scheduler
Multiple simultaneous requests issued
Can the order of requests lead to a failure? Yes!
Analysis revealed several undesirable cases

Cyclotron

Radiate room 2

Master Control Room

Treatment Room 1
Treatment Room 2
Treatment Room 3
Web Security Protocols

Alloy analysis of well-known protocols

By researchers from Stanford & Berkeley

Analyzed Origin tracking, referrer, HTML5, WebAuth, CORS

Previously unknown vulnerabilities in 3 protocols

http://seclab.stanford.edu/websec/

Robust Defenses for Cross-Site Request Forgery
Adam Barth, Collin Jackson, and John C. Mitchell

Towards a Formal Foundation of Web Security
Devdatta Akhawe, Adam Barth, Peifung E. Lam, John C. Mitchell, and Dawn Song
Chord Protocol

A peer-to-peer protocol for distributed hash tables
Developed by researchers at MIT
4th most cited paper in computer science
Proven Correct?

Three features that distinguish Chord from many other peer-to-peer lookup protocols are its simplicity, **provable correctness**, and provable performance.

*Ion Stoica et al. Chord: A Scalable Peer to Peer Lookup Service for Internet Applications, SIGCOMM 2001 (also TON, 2003)*
Three features that distinguish Chord from many other peer-to-peer lookup protocols are its simplicity, **provable correctness**, and provable performance.

*Ion Stoica et al. Chord: A Scalable Peer to Peer Lookup Service for Internet Applications, SIGCOMM 2001 (also TON, 2003)*

Modeling and analysis have shown that the Chord routing protocol is **not correct according to its specification**. Furthermore, not one of the six logical properties claimed as invariant is invariantly maintained by the protocol.

*Pamela Zave. Invariant-Based Verification of Routing Protocols: The Case of Chord, 2009*
Other Applications

- Smart Cards
- Electronic Voting
- JVM Security
- Network Configurations
- Flash File System
- Biometrics
- Smart Cards
- Electronic Voting
- JVM Security
- Network Configurations
- Flash File System
- Biometrics
Connecting Code to Design
Program Analysis with JForge

Check source code against design

JForge: Implemented as an Eclipse plugin
Automatic Test Case Generation

Exhaustive test cases from design

**Korat**: Implemented as an Eclipse plugin
Conclusion
Conclusion

Alloy
A lightweight modeling language

Support for incremental design
Model, analyze, fix & repeat

Our experience
Analysis often reveals many surprises
For More Information

Alloy
Tool download, tutorial, case studies
http://alloy.mit.edu

Book
Sample chapters, models
http://softwareabstractions.org

Software Design Group
People, projects, papers
http://sdg.csail.mit.edu

Contact us! (alloy@mit.edu)