Alcino Cunha

SPECIFICATION AND MODELING

INTRODUCTION

Universidade do Minho & INESC TEC

2019/20

MOTIVATION

THIS COURSE IN A NUTSHELL

- Languages and tools for (formal) software design:
  - Languages to *model* the system being designed
  - Languages to *specify* the desired properties of the design
  - Tools to *explore* and *analyse* the design

FORMAL SOFTWARE DESIGN

- A software design is a higher-level abstraction of the desired system
- A programming language is not adequate for this task
- The language of mathematics, logic, is a much better alternative
- It enables a formal approach to software design

Leslie Lamport

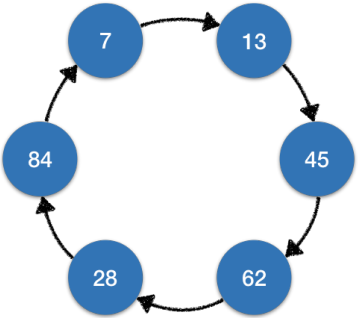
"If you're not writing a program, don't use a programming language"

TYPICAL ANALYSES

- Elicit requirements or structural constraints over a design
- Simulate a design to understand it
- Check consistency of requirements
- Verify that some structural, invariant or temporal properties hold

EXAMPLES

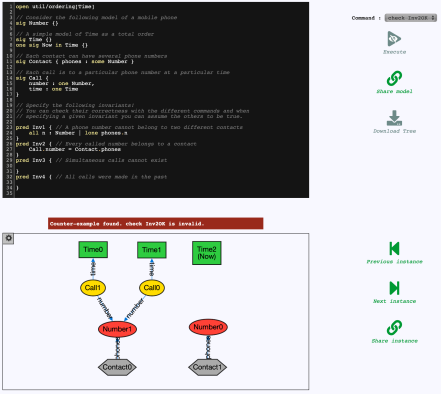
LEADER ELECTION IN A RING



Verify the correctness of the protocol:

- One leader will be elected

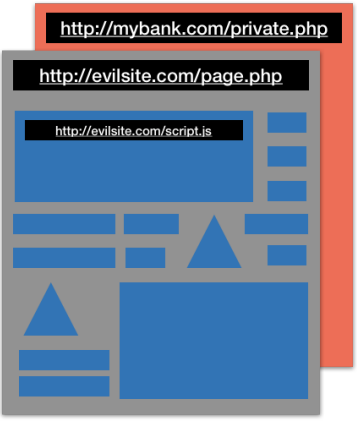
ALLOY4FUN



Explore design alternatives and verify data invariants:

- Non-shared stored models can have at most one derivation

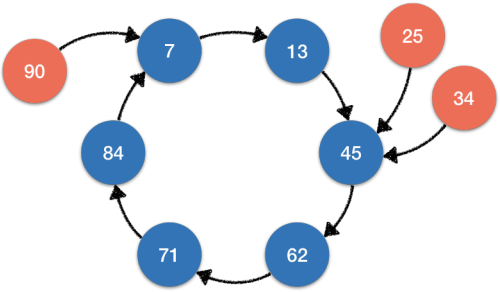
SAME ORIGIN POLICY



Understand and verify the policy:

- Resources can only access resources from the same origin

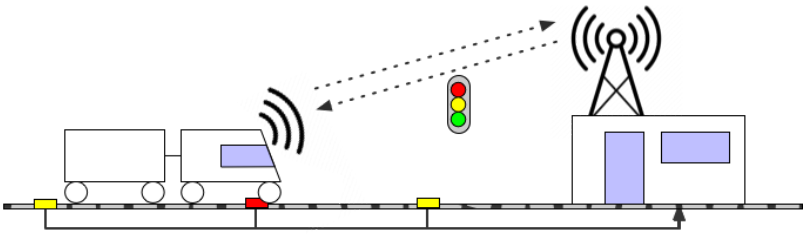
CHORD DISTRIBUTED HASH-TABLE



Explore variants of the protocol and verify correctness:

- If joins and failures cease, the network will eventually become a ring

HYBRID ERTMS/ETCS LEVEL 3



Verify the design of a railway traffic management system:

- Assigned movement authorities are safe

SYLLABUS AND ASSESSMENT

LOGICS

First-order logic

The fundamental logic to specify properties about states

Relational logic

A variant of FOL better suited for software design, where the state is typically described by *relationships* between concepts or objects

Temporal logic

A logic to specify properties about behaviour

ANALYSIS TECHNIQUES

Simulation

Key technique to understand, debug, and explore alternatives of a design

Model-finding

Automatic generation of structures satisfying a set of constraints
Can also be used to automatically check the structural properties of a design

Model-checking

Automatic verification of the temporal properties of a design

MAIN LANGUAGES AND TOOLS

Alloy

Native support for *sets* and *relations*, relational logic, and model-finding
Good for the structural design of complex (graph-like) structures

Electrum

Extends Alloy with temporal logic and model-checking
Good for the behavioural design of systems with complex configurations

OTHER LANGUAGES AND TOOLS

SMV

The quintessential model-checker, with support to various temporal logics
Good for the design of simple systems or as a back-end analysis tool

TLA+

Supports many data-types and (limited) temporal logic specifications
Good for the design of distributed and concurrent algorithms

ASSESSMENT

- Written test with development and problem-solving questions (70%, >= 8)
- Two group assignments, made in groups of 2 elements which are individually evaluated (30%, >= 10).