Distributed Systems Programming F29NM1

Formal Methods for Distributed Systems

Andrew Ireland

School of Mathematical and Computer Sciences
Heriot-Watt University
Edinburgh

Module Overview

Distributed Systems Programming [module F29NM1]	=	**Formal Methods + Implementation Technologies**

Lecturers: Andrew Ireland (G.57) & Hamish Taylor (1.43)
`air@macs.hw.ac.uk` `hamish@macs.hw.ac.uk`

Lectures: Mon-12.15 in 1.70; Wed-11.15 in 1.83; Wed-13.15 in 3.36

Labs: Wed-13.15 in 2.50 (Linux Lab)

Coursework: Two assignments, one for each part of the module.

Examination: After Easter - questions from both parts.

Materials: Formal methods teaching materials are on the web:

`http://www.macs.hw.ac.uk/~air/spin/`

The Economic Motive

"... the national annual cost estimates of an inadequate infrastructure for software testing are estimated to be \$59.5 *billion."*

Federal Study, US Dept of Commerce, May 2002.

"Worse – and spreading the effect of software flaws far beyond the original customer – several devastating computer viruses have taken advantage of bugs and defects in common operating systems ...

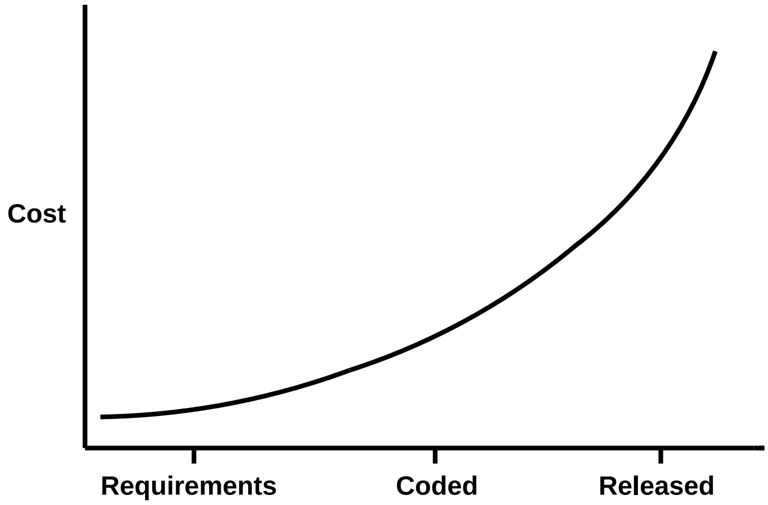
CNET Networks Inc, Aug 2002.

- US Internal Revenue Service – a failed \$4-billion modernization effort in 1997, followed by an equally troubled \$8-billion update.
- FBI – \$170-million virtual case-file management system was terminated in 2005.

More of the Same?

- Conventional modelling techniques rely heavily on natural language and diagrammatic methods.
- Such approaches make it hard to:
  - Write unambiguous models.
  - Analyse properties of our models.
  - Generate effective test cases for our implementations.
- Omissions and defects introduced early within the life-cycle are the most expensive to rectify if they go undetected ...

The Economics of Defect Detection



(Boehm, 1976)
Late life-cycle fixes are generally costly, *i.e.* can range from 40% to 100% more expensive than corrections in the early phases.

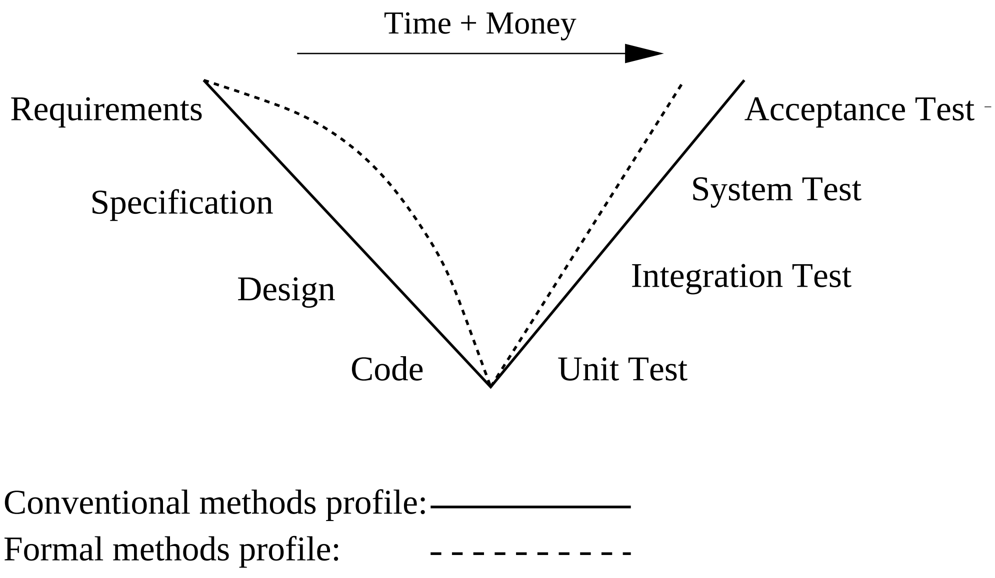
Complementary Methods

- The notion of **formal methods** has emerged over several decades as a way of addressing the weaknesses of the conventional methods highlighted above.
- One definition of formal methods is:

  "... a set of tools and notations (with a formal semantics) used to specify unambiguously the requirements of a computer system that supports the proof of properties of that specification and proofs of correctness of an eventual implementation with respect to that specification."

  M.G. Hinchey & J.P. Bowen (1995)

Drivers: Business & Economic Related



Drivers: Safety Related Standards

- RTCA DO-178B (USA Civil Avionics)
- Def Stan 00-55 (UK MoD)
- IEC 61508 (Generic "Programmable Systems")
  - IEC 601 (Medical Equipment)
  - (Pr)EN 50128 (Railway Industry)
- IEC 880 (Nuclear Power Control)
- MISRA (Automotive Industry)
- FDA (Medical Equipment)

Health Warning

- There are no absolute guarantees.
- When applied correctly, formal methods have been demonstrated to result in systems of the highest integrity.
- Correctness is only guaranteed with respect to a specification — you need to validate the assumptions which under-pin the specification.
- Formal methods complement rather than replace conventional approaches, *e.g.* testing, simulation and prototyping.
- But formal methods are applied by humans who are error prone — so tools are crucial.

When should Formal Methods be Used?

- **Complex:** abstraction is an important technique for managing the complexity of large systems and is central to the notion of a formal method.
- **Concurrent:** distributed systems give rise to concurrency. While we find it hard to reason about concurrency, certain formal methods have been developed which ease this task.
- **Quality-critical:** applications where failure is not dangerous but economically expensive, *e.g.* financial applications and telecommunications.

When should Formal Methods be Used?

- **Safety-critical:** applications where failure may endanger human life, *e.g.* fly-by-wire control systems and railway signalling systems.
- **Security-critical:** applications where failure means unauthorized access to sensitive information, *e.g.* medical records and security databases.
- **Standardized:** where systems are designed to meet specific, internationally recognized, standards then it is important that the standards can be interpreted uniformly, *e.g.* language specifications and protocol standards.

What Do Formal Methods Cost?

- The cost of applying formal methods is high, *i.e.* labour intensive coupled with a skills bottle-neck.
- Need for support tools which are integrated within the conventional software development environments.
- The potential for "re-use" within formal methods is high — At the 4^{th} *NASA Langley Formal Methods Workshop* (1997), work by Rockwell Avionics Research on the formal verification of the AAMP family of microprocessors (designed for embedded real-time applications used on Boeing 737, 747, 757 & 767 aircraft) demonstrated a 6 fold speed up in the formal verification effort when the work under-taken on the AAMP-5 was reused with the AAMP-FV.

The Cost of Failure

- In 1994 a bug in the floating-point hardware of Intel's Pentium microprocessor was discovered. The replacement costs were $>$ \$400 million.

  Intel now has a number of Formal Methods teams in the US ...

- In 1996 on the maiden flight of Ariane 5, just 39 seconds into its maiden flight Ariane 5 initiated self-destruct mechanism ... Ariane 5 cost the European Space Agency 10 years and \$7 billion to produce.

  Ariane 5 was running Ariane 4 software, however, underlying hardware architectures were different – self-destruction occurred when the Ariane 5 guidance system tried to convert a 64-bit number (velocity data) into a 16-bit format – resulted in an overflow error.

The Cost of Failure

- Therac-25: a computer-controlled radiation therapy machine, build by Atomic Energy of Canada Ltd (AECL) used in US and Canadian hospitals and clinics during the 1980's. The Therac-25 was the successor to the Therac-6 and Therac-20 models. Unlike its predecessors the Therac-25 relied more on software control mechanisms. Potential hazards from the Therac machines are **high energy beam** with **inappropriate magnet settings**.

  Hazard analysis for the Therac-25 (March 1983) excluded the possibility of software defects since "extensive testing" had been undertaken. However, software errors resulted in several patients being killed and injured by radiation overdoses during the mid to late 1980's.

Which Formal Method is Best?

- The choice is very much application dependent – indeed a number of complementary methods may often be required for a single application.
- When specifying state based aspects of systems it is best to use a model-based approach such as:
  - Z: *The Z Notation: A Reference Manual*, Spivey, J.M. Prentice Hall 1992.
  - VDM: *Systematic Software Development using VDM*, Jones, C.B. Prentice Hall 1990.

Which Formal Method is Best?

- Distributed concurrent systems:
  - Process algebras provide formalisms for modelling distributed current systems:
    - CCS: *Communication and Concurrency.*
    - CSP: *Communicating Sequential Processes.*
    - LOTOS: *Language Of Temporal Ordering Specification.*
  - Description languages, less formal but greater industrial up-take:
    - SDL: *Specification and Description Language.*
    - Promela: *PROcess MEta LAnguage.*

Examples from Industrial

- SPARK: A programming language derived from Ada that includes annotations – SPARK toolset supports flow analysis and formal verification (Praxis critical Systems, UK).
- ESTELLE (telecommunications) SCADE (embedded systems): Support specification and an notion of *correctness-by-construction*, (Esterel Technologies, France).
- SDV: Static Device Verifier automatically analyzes system software (C programs) – detects violations with respect to application programming interface (API) usage rules (Microsoft Research, US)

  `http://www.microsoft.com/whdc/devtools/tools/sdv.mspx`

Aims and Objectives

- To promote an understand of the issues involved in using formal methods within system design, in particular the design of distributed and concurrent systems.
- To provide practical experience of the formal modelling and analysis of such systems through Promela and the SPIN design verification tool.
- To give an insight into the theory which underpins such formal modelling and analysis tools.

Summary

Learning outcomes:

- Gain an understanding of the:
  - Limitations of conventional modelling and analysis techniques.
  - Complementary nature of formal methods as well as their strengths and weaknesses.

Recommended reading:

- M.G. Hinchey & J.P. Bowen (Eds), *Applications of Formal Methods*, Prentice Hall 1995.
- `http://www.macs.hw.ac.uk/~air/spin/`
- `http://www.afm.sbu.ac.uk`