Formal Methods
in
ICL
Secure Systems

R.B. Jones

International Computers Limited,
Eskdale Road,
Winnersh,
Wokingham,
Berks RG11 5TT

tel: 0734 693131 x6536,
fax: 0734 697636
email: uucp: rbj@win.icl.co.uk

FORMAL METHODS

applying

LOGIC

and

MATHEMATICS

to

HIGH ASSURANCE
SYSTEMS ENGINEERING

ASSURANCE through PROOF

special
REQUIREMENTS
-
HIGH ASSURANCE

special
METHODS
-
based on
FORMAL SPECIFICATION and PROOF

special
TECHNOLOGY
-
NEW ‘LOGIC LANGUAGES’
NEW ‘LOGIC ENVIRONMENT’

Business Motivators

- Quality
- Secure Systems Market
- Safety Critical Systems Market
- Professional Services Market supporting above
- Certification Schemes for Secure (and Safety Critical) Systems

Application Areas

- Security
  - Non-disclosure of information
  - Authentication of users
  - Integrity
- Safety Critical Systems
  - Aerospace
  - Medical
  - Industrial Plant
  - Nuclear plant
  - Public transport

Security Certification Schemes

US Department of Defence Trusted Computer System Evaluation Criteria "Orange Book"

CESG Computer Security Memorandum No.3 "UK Systems Security Confidence Levels"

Harmonised Criteria Information Technology Security Evaluation Criteria "ITSEC"

Safety Critical

UK MOD interim defence standards 00-55/56

Applications In ICL Secure Systems

- Use of VDM on VDAP compiler

- Z on various Design Study and Security Modelling contracts for HMG

- HOL/Z on CESG/ICL OWR project

- ICL HOL and Z proof tool

- Z for security modelling on major contracts

- VDM/Z used in various bids.

The CESG/ICL OWR Project

- A design-and-implement contract placed with ICL by CESG (Communications and Electronics Security Group, GCHQ)

- Design and make pre-production models of a ONE WAY REGULATOR to the highest achievable standards of assurance.

- Hardware only solution

- Full formal verification of detailed design against formal security policy.

- Product certified as "exceeding the requirements of UKL6"

PROOF themes

focus formality

automate proof

METHODs supporting selective application of formality to best effect in combination with other methods.

TOOLs proving least cost proof development support integrated with support for other methods.

FORMAL DEVELOPMENT for CRITICAL COMPONENTS

REQUIREMENTS supplemented by HAZARD ANALYSIS

IDENTIFY CRITICAL COMPONENTS

Use FORMAL approach for CRITICAL COMPONENTS

Use STRUCTURED approach for NON-CRITICAL COMPONENTS

FORMAL DEVELOPMENT for CRITICAL COMPONENTS - FOR and AGAINST

FOR

formal treatment focused on critical components

AGAINST

critical components may be incorrectly identified

specifications of critical components may be incorrect

FORMAL DEVELOPMENT for CRITICAL REQUIREMENTS

formalise critical requirements on SYSTEM

formally model architecture

formalise critical requirements on SUBSYSTEMS

verify architecture

repeat through structured design process

implement and verify using pre/post conditions

FORMAL DEVELOPMENT of CRITICAL REQUIREMENTS - FOR and AGAINST

FOR

formal treatment focused
on critical requirements

identification of critical components
formally verified

requirements on critical components
formally verified

AGAINST

lack of literature on techniques

PROCESSING of FORMAL (Z) SPECIFICATIONS

SYNTAX CHECKING

TYPE CHECKING

....

CONSISTENCY PROOFS

SEMANTIC WELL-FORMEDNESS PROOFS

PRE-CONDITION SIMPLIFICATION

REFINEMENT VERIFICATION

....

PROOF of CRITICAL PROPERTIES

CODE/HARDWARE VERIFICATON

REQUIREMENTS for PROOF TOOLS

SOUNDNESS/INTEGRITY

PRODUCTIVITY

ADAPTABILITY/EXTENDIBILITY

ICL PROOF TOOL

well established
proof support paradigm
(LCF paradigm)

modern functional metalanguage
(ML)

primary object language HOL
(Higher Order Logic)

support for multiple
'secondary' object languages
(e.g. Z)

THE LCF PARADIGM

- implement proof checker using a TYPED FUNCTIONAL programming LANGUAGE as META-LANGUAGE (e.g. SML)

- abstract data type of THEOREMS GUARANTEED SOUND by the type checker (assuming the logic is well defined)

- META-LANGUAGE is AVAILABLE TO the USER for programming proofs and other customisation, WITHOUT risk of COMPROMISING the CHECKER.

BENEFITS of the LCF PARADIGM

- HIGH ASSURANCE of SOUNDNESS
- EASY to CUSTOMISE and EXTEND
- COMPLETE USER CONTROL of PROOF STRATEGY
- RERUNNABLE PROOF SCRIPTS
- LEG WORK convertible to HEAD WORK by PROGRAMMING in META-LANGUAGE

ICL PROOF TOOL

-

CRITICAL REQUIREMENT

all theorems are true

if

all extensions are conservative

ARCHITECTURE

code for management of theory database and checking of proofs separated out and protected using abstract data type

code for all other functions (e.g. syntax checking, type inference, proof heuristics) written in Standard ML, but non-critical.

System user extensible.

Current Status

- Under development in government supported FST project

- Prototype ICL HOL system developed and used on project since MAY 1990.

- Prototype Z proof support on HOL prototype.

- ‘Product quality’ ICL HOL well progressed.

- Collaboration with PVL to provide integrated support for development through to SPARC (Ada subset) implementation.