Formal Modeling and Test Generation Automation with Use Case Maps and LOTOS
Leïla Charfi

School of Information Technology and Engineering
University of Ottawa

September 7th, 2001
Outline

- Motivation
- Previous work
- Framework of our research
- Contributions, usefulness and limitations
- Case Study
- Perspectives
Motivation

- The thesis addresses the problem of formal modeling and test generation in telecommunication systems
- Telecommunication systems are evolving
- New users requirements have to be met
- Time-to-market

→ Need for development methodologies allowing fast and robust software design
Previous work

- Existing development methodologies:
  - Informal requirements, UCMs (semi-formal), ...
  - Formal specification languages: LOTOS, SDL, ...

- Existing testing tools:
  - LOLA, TGV, TorX for LOTOS specifications and tests
  - TAU for SDL specifications and TTCN test suites
Framework of our research

- Use existing development methodologies essentially using:
  - UCMs,
  - LOTOS, SDL

- Develop better solutions to meet the expectations of the industry
  - Better and faster design
  - Better and faster testing
Contributions

• Design of a LOTOS model for a new Mitel PBX

• Automatic LOTOS scenario generation from UCMs

• New development methodology with fast test suite generation
Use Case Map

TTCN test case
Contribution 1  
Design of a LOTOS model

- Follow an existing development methodology that uses:
  - Use Case Maps to express the requirements
  - LOTOS and SDL at specification stage
  - MSCs for verification and validation
  - TTCN for conformance testing with the implementation

- Build a LOTOS specification that meets the requirements
  - Mapping rules from UCM elements to LOTOS elements

- Build a set of scenarios used for:
  - Validation of the specification
  - Cross validation between the LOTOS and the SDL specifications
Contribution 2

Automatic scenario generation from UCMs

- Developed and implemented a tool for the automatic generation of LOTOS scenarios from a UCM map: *Ucm2LotosTests*
Contribution 2

Automatic scenario generation from UCMs

• Usefulness:
  → Automatic generation → saves time
  → Set of scenarios cover the UCM map since the algorithm visits all the routes
  → Depending on the UCM to LOTOS mapping provided, the scenarios can be used for:
    - White-box testing
    - Grey-box testing
    - Black-box testing
  → Obtained LOTOS scenarios useful for:
    - Verification of the LOTOS specification (white-box, grey-box testing)
    - Validation of the LOTOS specification (black-box testing)
    - Conformance testing (by using scenarios as inputs for the TTCN test suite generation)
Contribution 2

Automatic scenario generation from UCMs

• **Consideration:** Generation of rejection scenarios

UCM

Rejection scenarios
Contribution 3

New development methodology with fast test suite generation

- LOTOS specification
- UCMs
- Verification and validation of the specification with CAESAR
- Test suite generation with TGV
- LOTOS scenarios
- TTCN test suites
- Ucm2LotosTests
- UCM routes

Automatic path
Manual path
Contribution 3
New development methodology with fast test suite generation

• Usefulness:
  - “Fast” methodology because of automation
  - Use of the semi-formal notation UCMs allows preliminary semantic check of the system’s behavior
  - Use of the formal language LOTOS
    → Specification is robust and not ambiguous
  - Automatic TTCN test suite generation from UCMs

• Limitations:
  - Use of CAESAR implies:
    - The LOTOS specification style is restricted
    - We must build a LOTOS specification that generates a small number of states
Case Study

- Mitel’s Basic call specification and test generation

→ Tool problems:
  - Use of CAESAR \(\rightarrow\) Simplified Basic call specification
  - Use of TGV \(\rightarrow\) LOTOS to TTCN test conversion was not straightforward.
Perspectives

- Improve Ucm2LotosTest
  - Handling recursive stubs
  - Regression testing
- Improve CAESAR and TGV
- Automate UCM to LOTOS specification
- Investigate the use of LOLA instead of CAESAR
- Investigate the use of SDL and the powerful SDL based tools