Feature Description and Feature Interaction Analysis with **Use Case Maps** and **LOTOS**

Daniel Amyot et al.

SITE, University of Ottawa, Canada

damyot@site.uottawa.ca

FIW'00, Glasgow, May 19, 2000

Collaborators

- Leïla Charfi, *University of Ottawa*
- Nicolas Gorse, *University of Ottawa*
- Tom Gray, *Mitel Corporation*
- Luigi Logrippo, *University of Ottawa*
- Jacques Sincennes, *University of Ottawa*
- Bernard Stépien, *University of Ottawa*
- Tom Ware, *Mitel Corporation*

Introduction

- New methodology for feature design, specification and validation
- Jointly by U. of Ottawa and Mitel Corp.
- Application to new product
  - Enterprise private networks
  - Agent-based call model
  - Features: Outgoing Call Screening, Call Forward Always, Call Forward Busy, Call Hold, Recall, Call Pickup, Call Transfer

Approach

- ***Use Case Maps***
  - Causal scenario notation
  - Description and documentation of requirements and high-level designs
- ***LOTOS***
  - Formal algebraic specification language
  - Powerful validation & verification tools and techniques, enabling FI detection
- Both have an FI history, in isolation

Related Work

- Formal Methods
  - Precise, mathematical, but low penetration
- Scenario-Driven Approaches
  - Higher level of acceptance, accessible to a broad range of readers; but integration of scenarios and V&V remains difficult
- Some Well-Known Approaches
  - SDL and Message Sequence Charts
  - Unified Modeling Language

Two Complementary Techniques

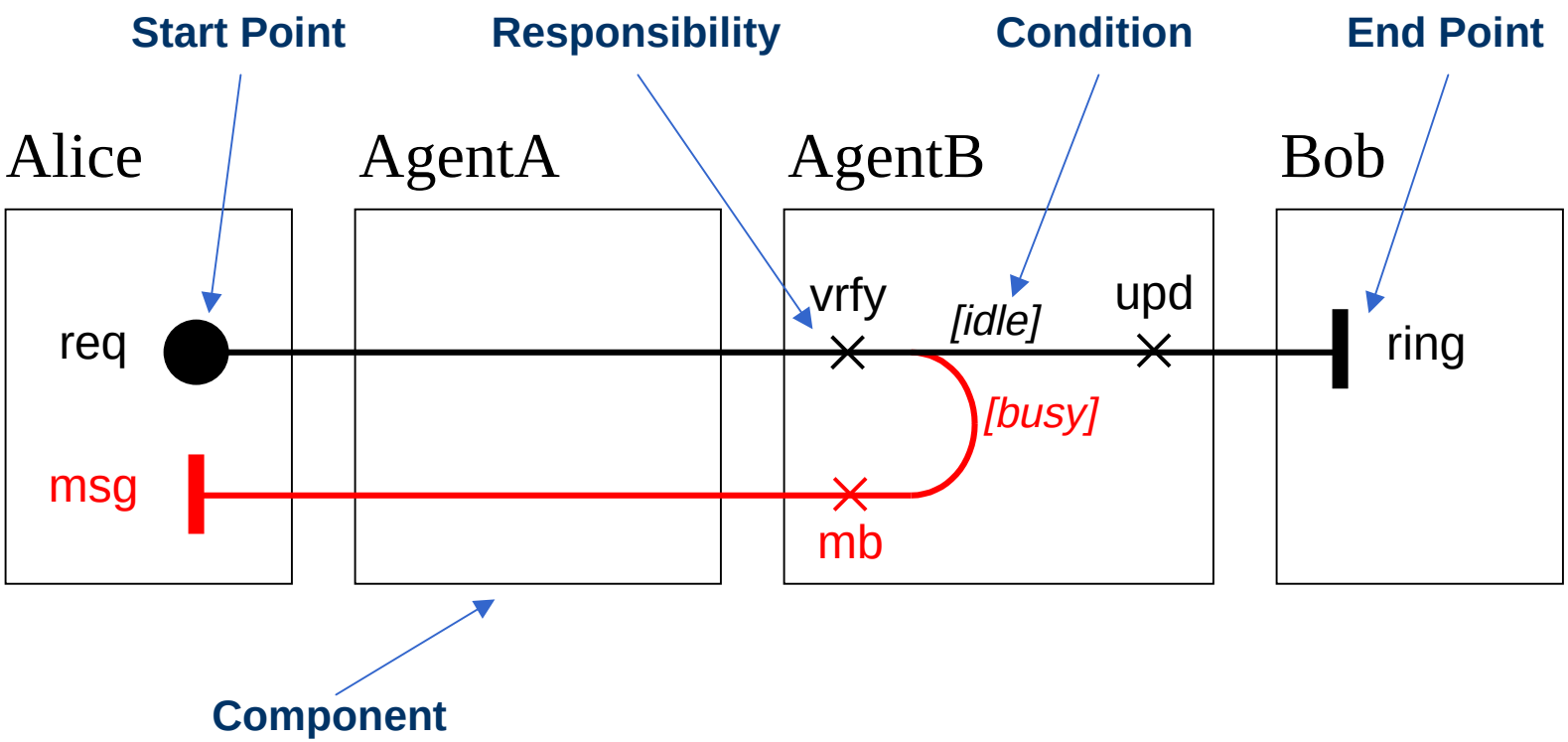
- Use Case Maps
  - Visual and intuitive scenario notation
  - Capture, integrate, and help reasoning about functional requirements
  - FI avoidance
- LOTOS
  - Formalization, abstract prototyping and validation
  - Automated FI detection

In This Presentation...

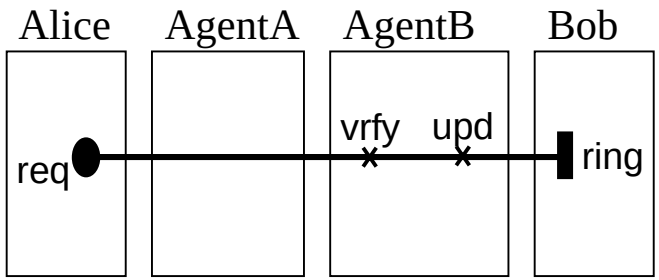
- Use Case Maps Notation
- System Architecture with Call Model UCMs
- UCM-Based FI Avoidance
- From UCMs to LOTOS
- Validation and FI Detection with LOTOS
- Traces, MSCs and Animations
- Conclusions

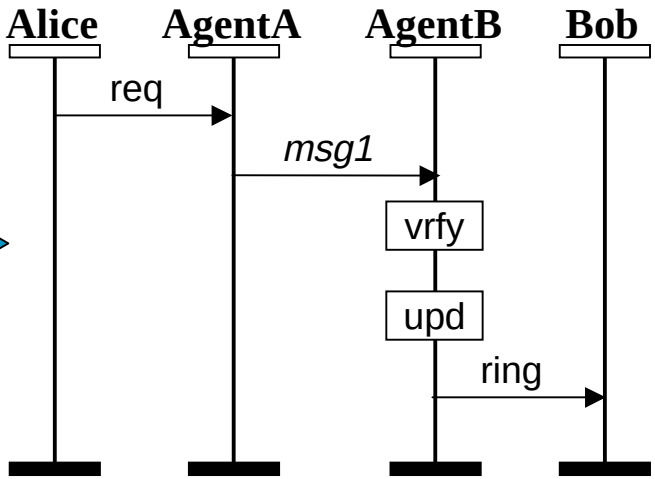
Use Case Maps Notation

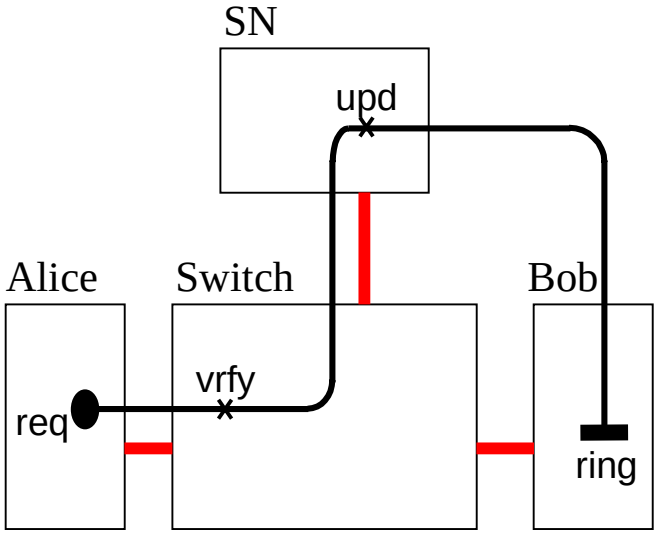
- Visualization of causal relationships between responsibilities allocated to abstract components

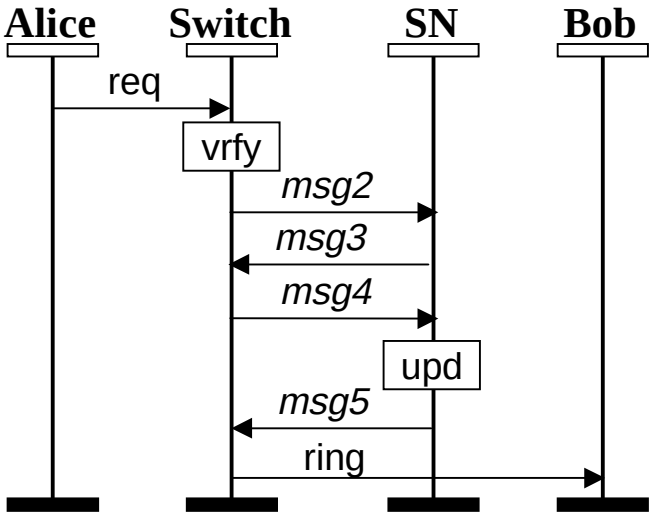


Refining UCMs with Message Exchanges

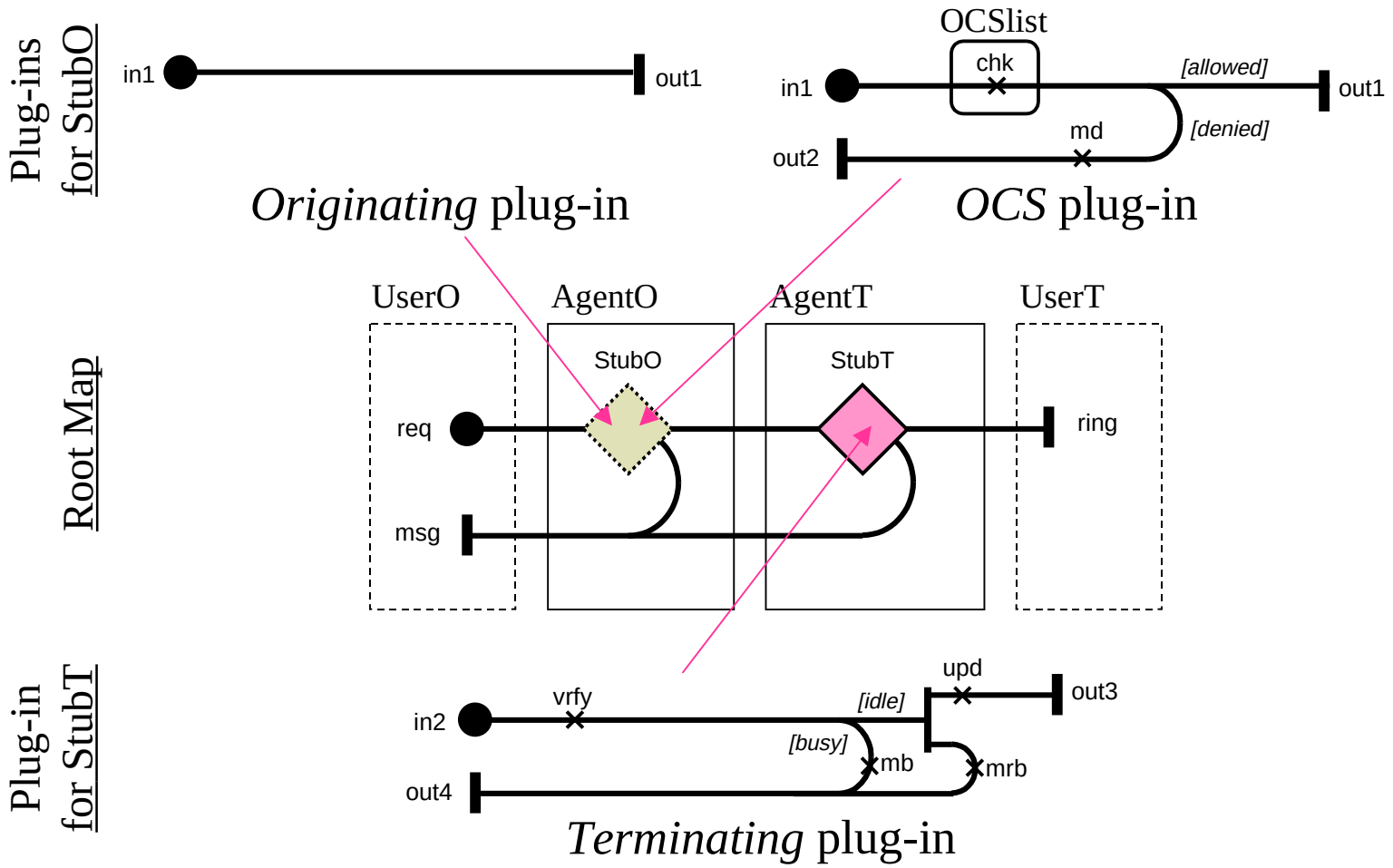








Integrating UCM Scenarios

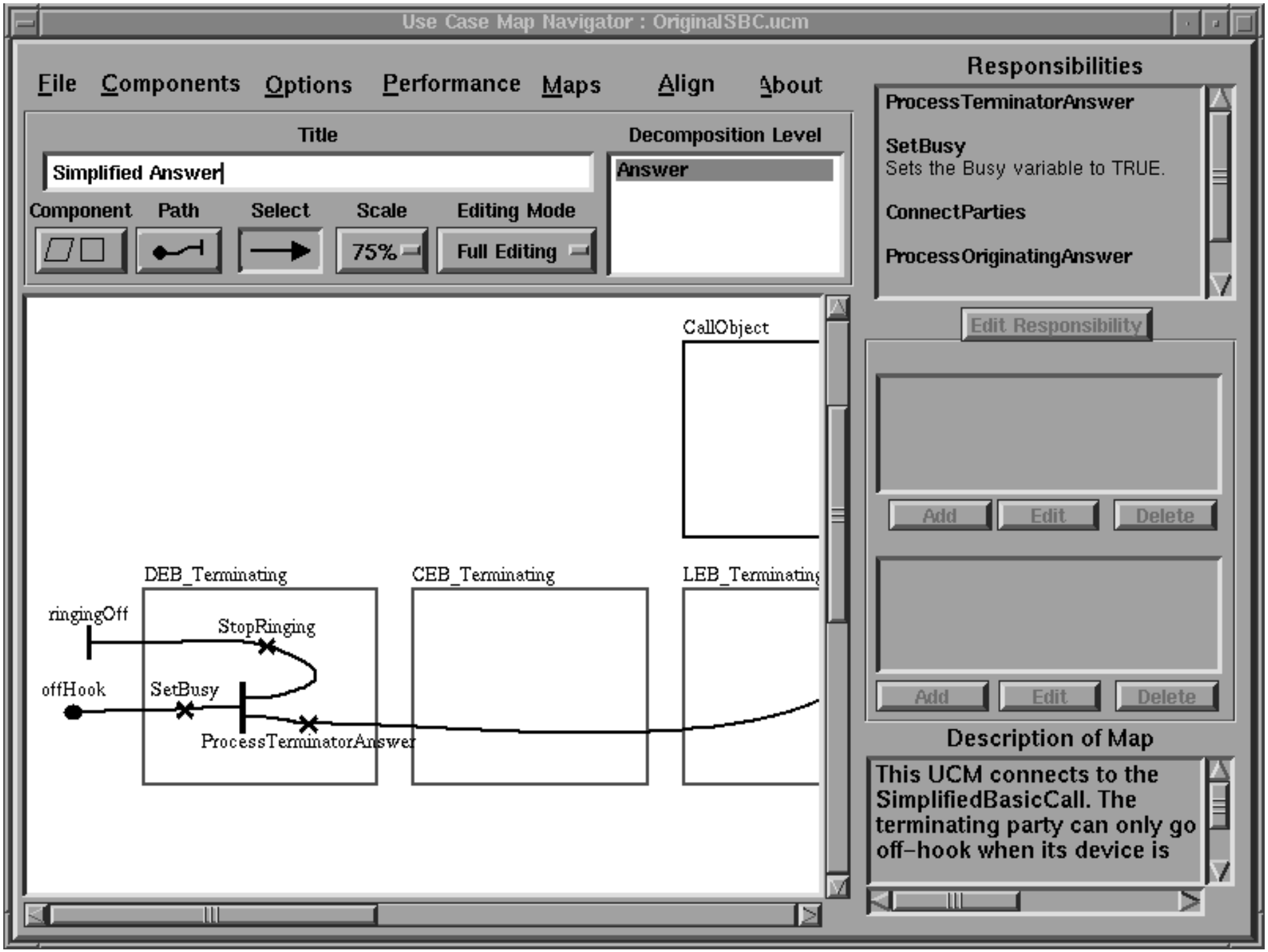


System Architecture

- Agents types:
  - Device Agents (DAGENT or DEB)
  - Personal Agents (PAGENT or CEB)
  - Functional Agents (FAGENT or LEB)
- Agents roles:
  - Originating, Terminating, 3rd party
- Call objects instantiated dynamically

Design of the Call Model UCMs

- Created by industrial partners
  - 1 senior designer and 2 junior designers
- More than 100 UCMs
  - Basic call and 10 features
  - Structured with 60 stubs
  - 7 levels deep
  - Many plug-ins reused
  - Recently added 3 features, low impact
  - Use of the *UCM Navigator*


Use Case Map Navigator : OriginalSBC.ucm
File
Components
Options
Performance
Maps
Align
About
Title
Simplified Answer
Decomposition Level
Answer
Component
Path
Select
Scale
75%
Editing Mode
Full Editing
CallObject
DEB_Terminating
CEB_Terminating
LEB_Terminating
ringingOff
StopRinging
offHook
SetBusy
ProcessTerminatorAnswer
Responsibilities
ProcessTerminatorAnswer
SetBusy
Sets the Busy variable to TRUE.
ConnectParties
ProcessOriginatingAnswer
Edit Responsibility
Add
Edit
Delete
Add
Edit
Delete
Description of Map
This UCM connects to the SimplifiedBasicCall. The terminating party can only go off-hook when its device is

FI Avoidance at UCM Level

- Many FI solved at integration time
- Before the generation of a prototype
- Remaining FI mostly in dynamic stubs
- Several problems detected by inspection
  - Non-determinism in selection policies
  - Erroneous UCMs
  - Ambiguous UCMs, lack of comments
- New techniques (e.g. Namakura et al.)

Towards LOTOS

- ISO standard, process algebra
- Powerful constructs
  - Composition: multiway rendezvous
  - Hiding
  - Abstract Data Types (ADT)
  - Flexible inter-process synchronization
- Constructs similar to those of UCMs

From UCMs to LOTOS

Start/end points	→	Visible gates
Responsibilities	→	Hidden gates
Agents/components	→	Processes
Stubs	→	Processes (implement selection policies)
Plug-ins	→	Processes
Inter-path causality	→	Hidden inter-process synchronization (msg)
Databases, conditions	→	Abstract Data Types

Validation

- Scenarios derived from UCMs paths for:
  - Basic System Properties
  - Individual Features Properties
  - Feature Interaction
- Scenarios simpler than specification
  - Few features considered at once
  - No component, close to requirements
- Verdicts obtained with LOLA

FI Analysis Phase

- FI team: 2 students
- No major fault, but several problems detected
- LOTOS specification: 2450 lines
- 36 test scenarios: 1300 lines
- Currently being extended in new phase
- Other LOTOS-based techniques and tools to be used

Feature Interaction "Suspiscion"

- Derivation of properties of individual features
- Analysis in Prolog to determine:
  - direct and transitive FI
  - non-determinisim
  - loops
- Generation of FI prone scenarios and configurations

Traces, MSCs and Animations

- LOTOS traces are translated to MSCs by associating direction to gates and identifying sender and receiver entities
- Translation of MSCs to LOTOS permits validation against *external* scenarios
- A graphical animator displays a given trace as a structural diagram of the system, in a step-by-step fashion

Conclusions

- UCM-LOTOS approach for specification and validation of telecommunications systems seems feasible and effective
- Encouraging results so far, more to come in the near future…
- Technology transfer in progress