Animation and formal verification of a component-based application using live sequence charts (LSCs) and the Play-Engine

Omega workshop

Grenoble – February 17, 2005

Pierre Combes (FTR&D), Hillel Kugler(Weizmann Institute)

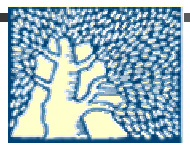
Using the Play-Engine and LSCs for studying a Telecom application (Depannage by FTRD)

Play-Engine Tool

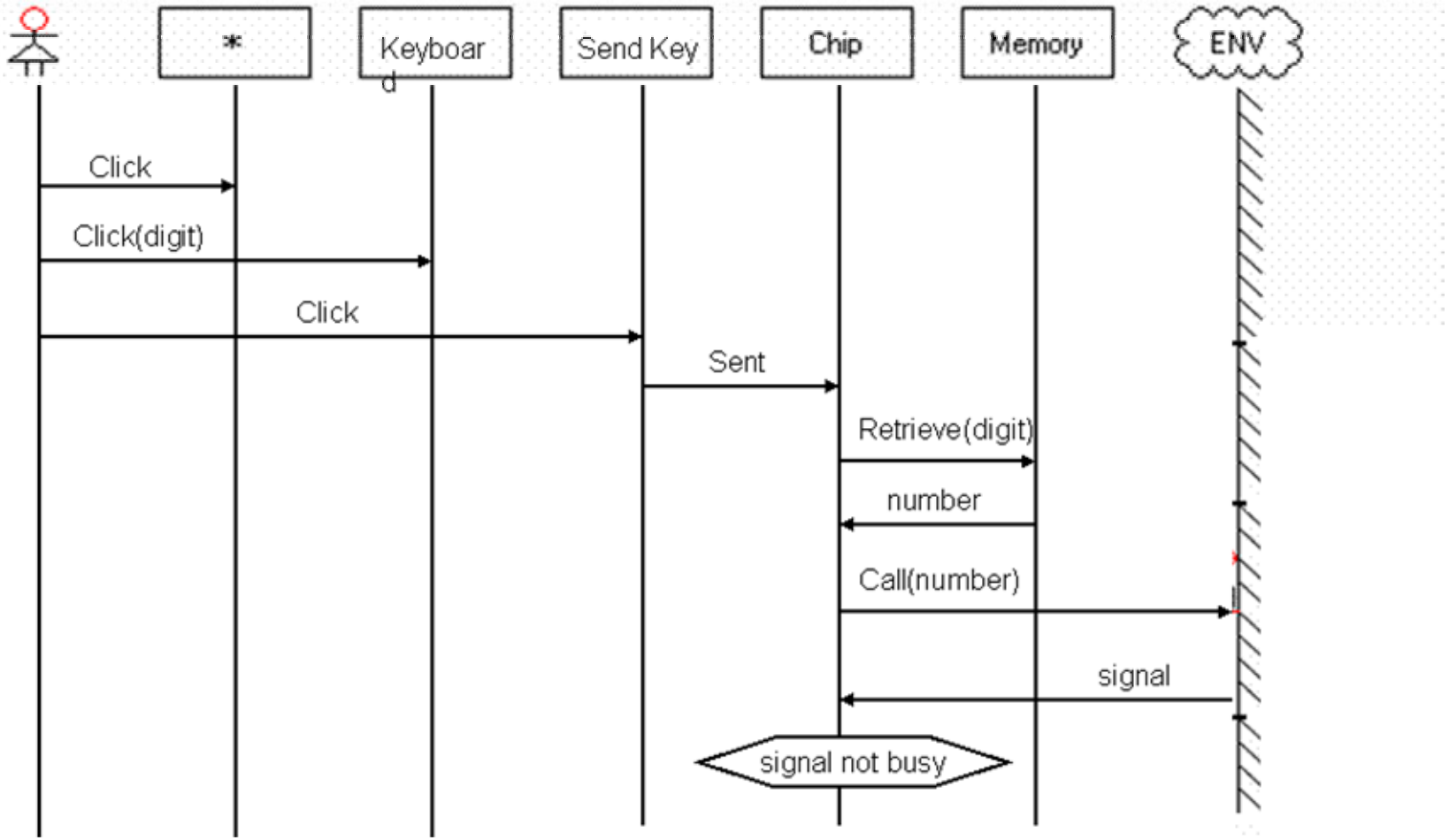
Language : Live Sequence Charts (LSCs)
Specification of Requirements – Play-In
Execution of Requirements – Play-Out
Smart Execution and Verification – Smart Play-Out

Motivation for work

Modeling in LSCs is a new approach
Evaluation by Industrial user and not the tool developer
Case Study Represents broader class of applications

Extend visual formalism used for requirements: message sequence charts (MSCs)

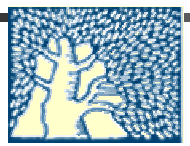


Live sequence charts (LSC's)

"LSC's: Breathing Life into
Message Sequence Charts"

(Damm & Harel, '98)

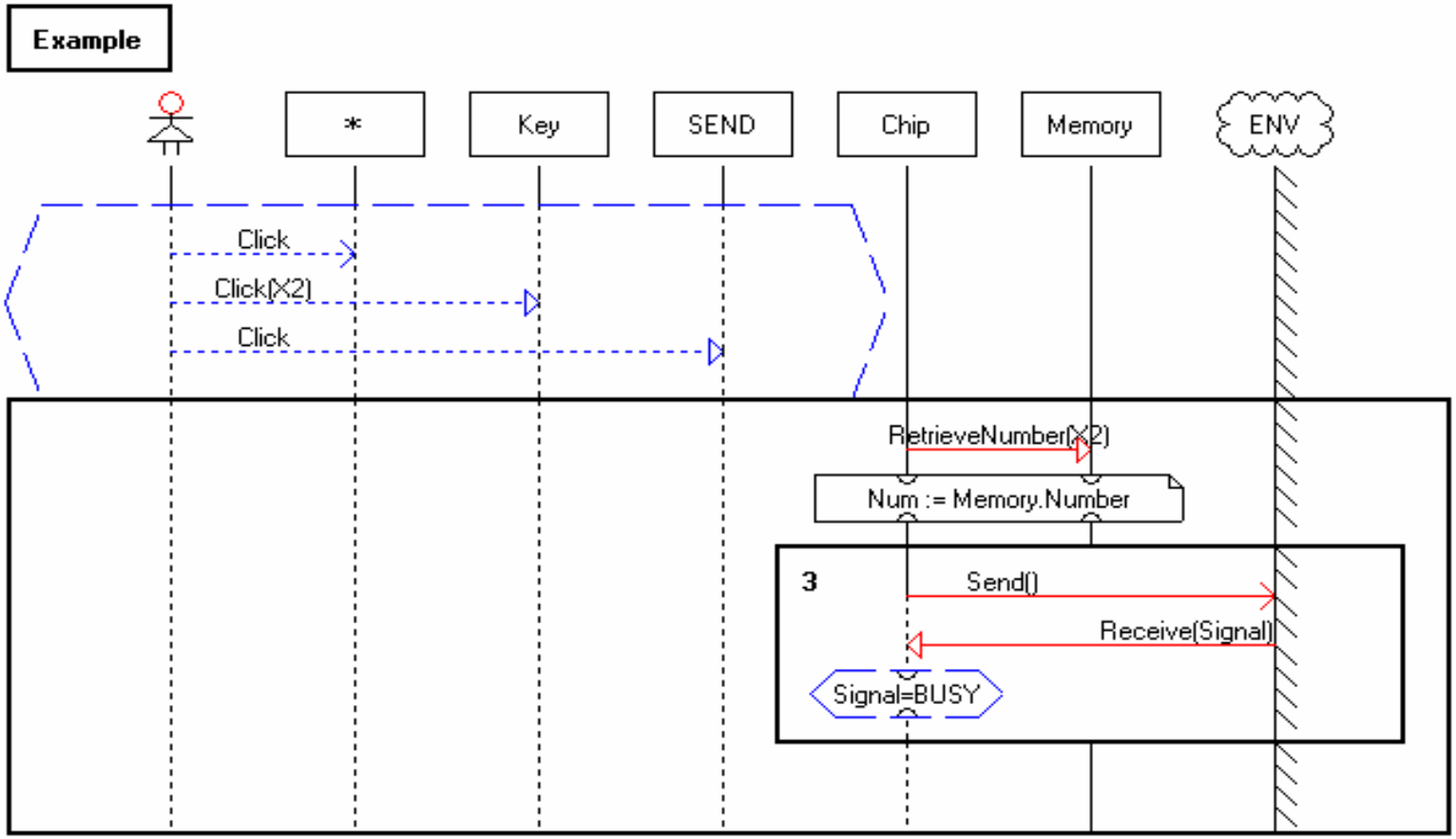
A natural extension of classical MSCs, with **modalities** (universal/existential, hot/cold, etc.) and **structure** (subcharts, conditionals, loops, etc.)

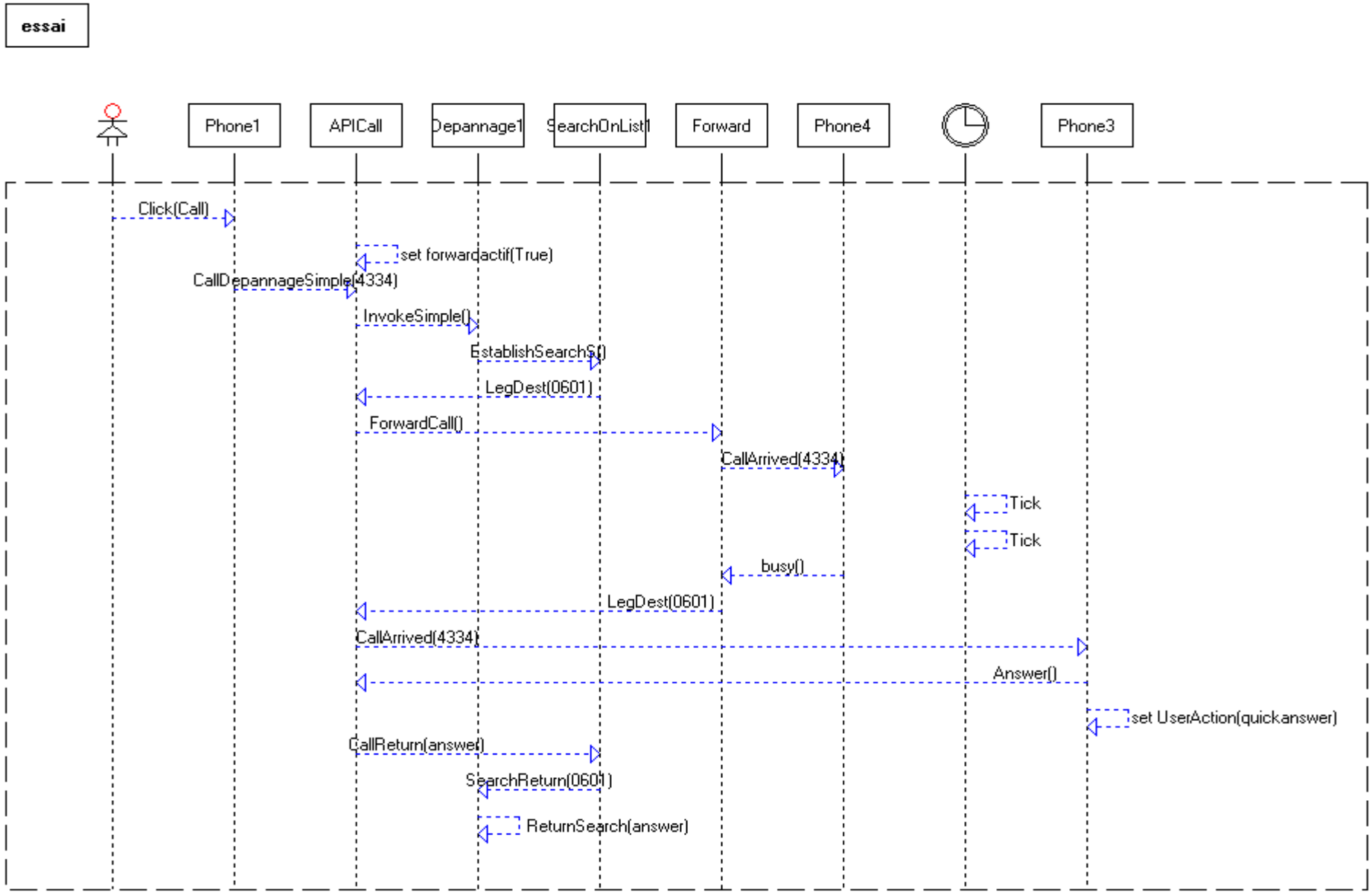
Basic form of a universal LSC

prechart (if)

main chart (then)

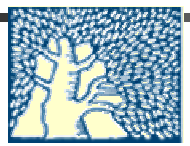


Existential LSC



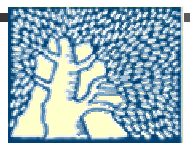
System (composite) and Components

- **A component-based approach**
- **A system (composite) is built from a set of embedded components**
  - ✓ The system (composite) is specified by a set of requirements
  - ✓ The architecture of the system is built from components and connectors: an architectural model
  - ✓ Assumptions could be associated to connectors
    - Introduction of delays, time constraints, loss message rates
- ✓ **Components are described by**
  - ✓ A set of interfaces (required and provided)
  - ✓ Assumptions (abstract behaviors) on their interfaces
  - Components should be reusable



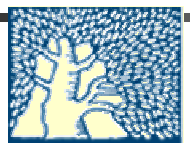
The Service: *Depannage (Emergency)*

- **A telecommunication service**
- **A User (fixed phone but mainly for mobile phone) calls a specific number in order to find assistance service (depannage but also urgency: police, fire brigade, doctor)**
- **The objective is to connect the user, as quickly as possible, to a member of the depannage society**
  - Which is at a location nearby the user location
  - Call attempts are done for different potential called numbers (in sequential or in parallel)
  - In any case, the caller should be connected to a vocal box or a secretariat
- **The depannage society has several employees**
  - Moving and which could be busy (by another client, or by another occupation) or not accessible (in a concert hall!)

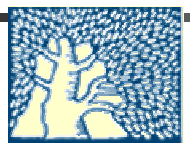
The Service: *Depannage (Emergency)*

- **Based on a set of service and platform components (embedded in mobile terminals or core network)**
  - Service Features: authentication, location, search (in sequential, in parallel), etc
  - Interface Features (for session control, user interface, location, discovery)
  - Platform Components (communication between platform(s) and network)
- **The environment model includes the users, the network and the location architecture**
- **Timed and Un-timed Requirements at the system level**
- **Many Time constraints in service components, component interactions and environment**
  - Time constraints that could lead to unexpected behaviours

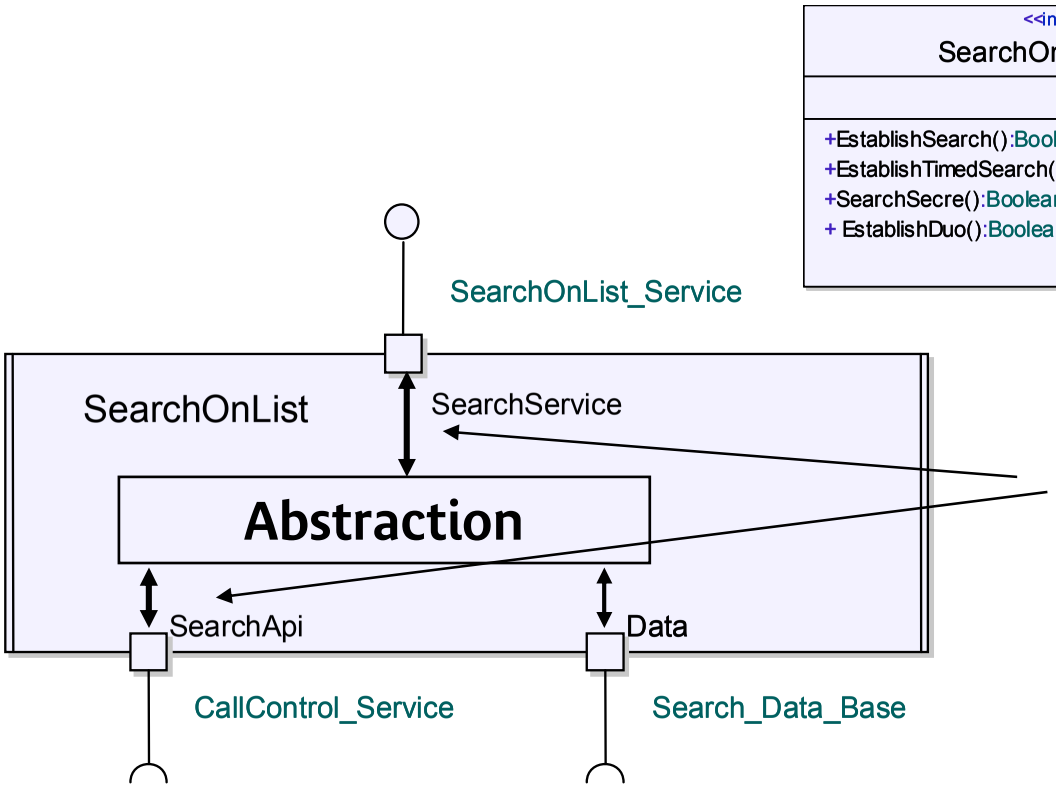


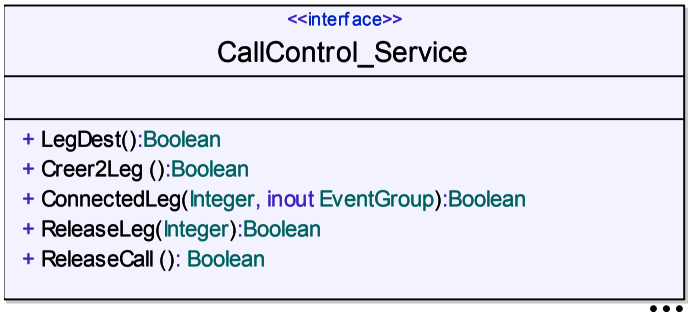
Component modeling with LSC

- **Components:**
  - Described independently of any embedding system
  - Described as a black box
    - Interfaces (signatures) and Ports
  - Described as a *grey* box
    - The abstract view (assumptions) of the behaviour of each component on its interfaces
    - Time constraints and Delays due to the specific platform (on task execution), periodic requests, etc
    - With help of Universal LSC

Component

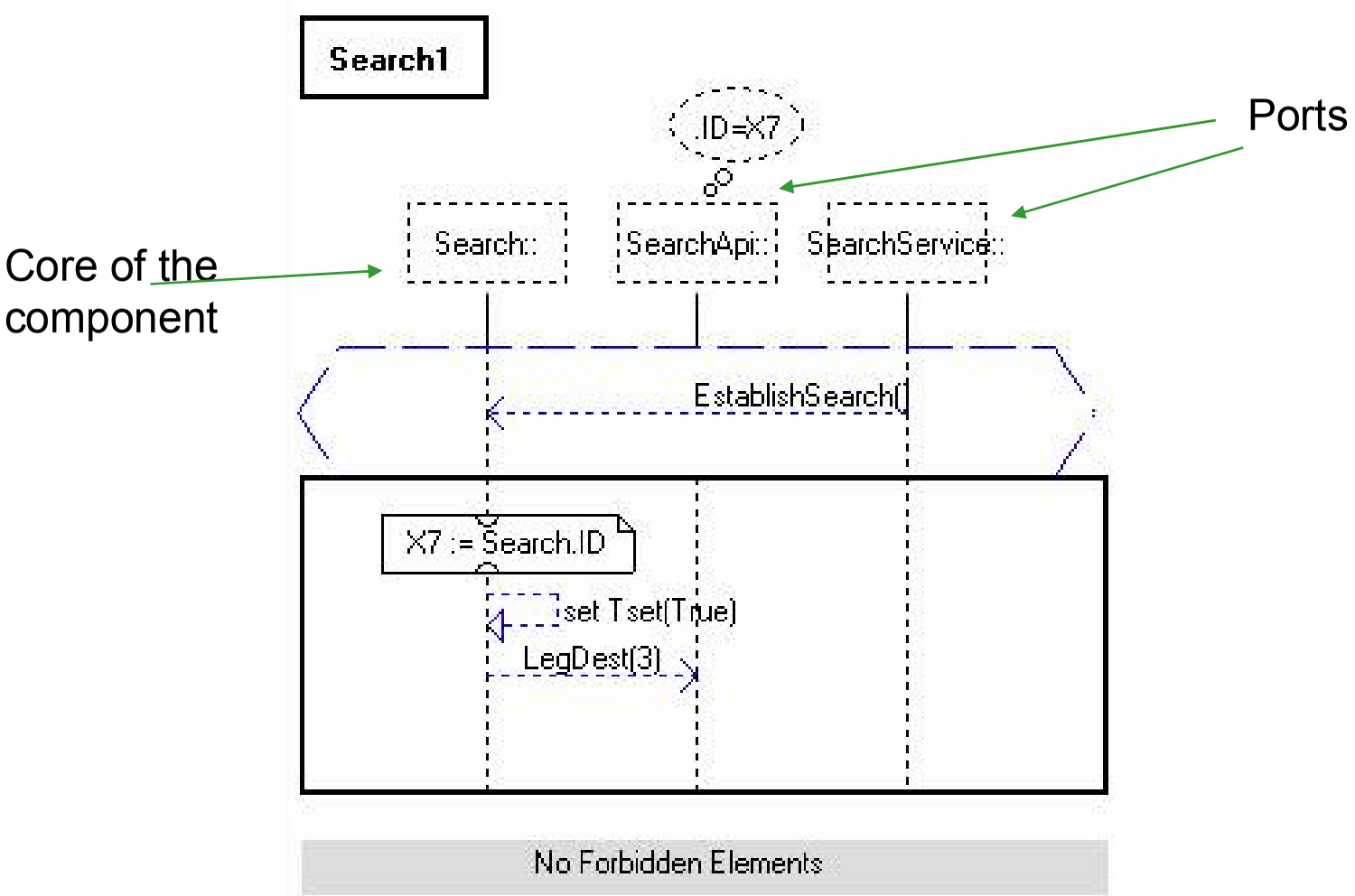




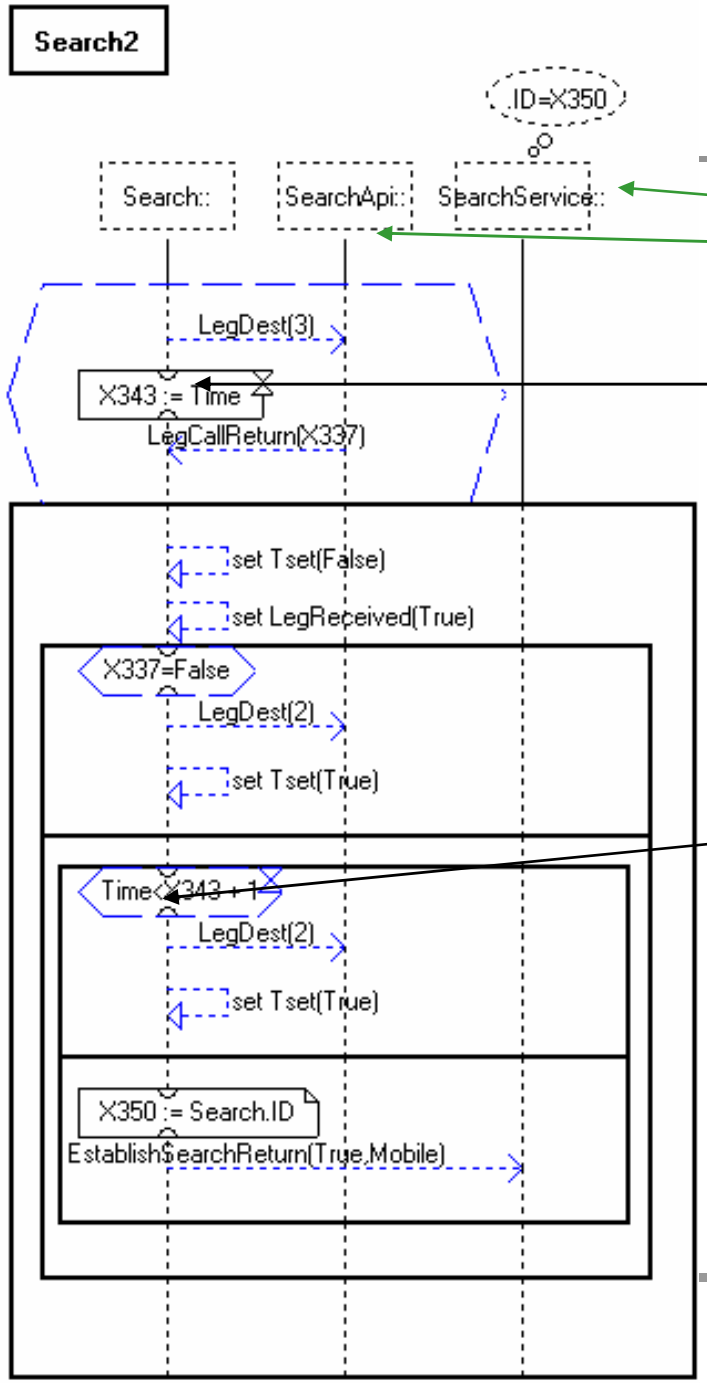
Described by a set of LSCs

Independently of any embedded system

Search On List



Search On List



ports

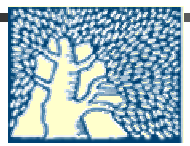
T is recorded, just after the sending of LegDest

The *main chart* is executed On reception of LegCallReturn

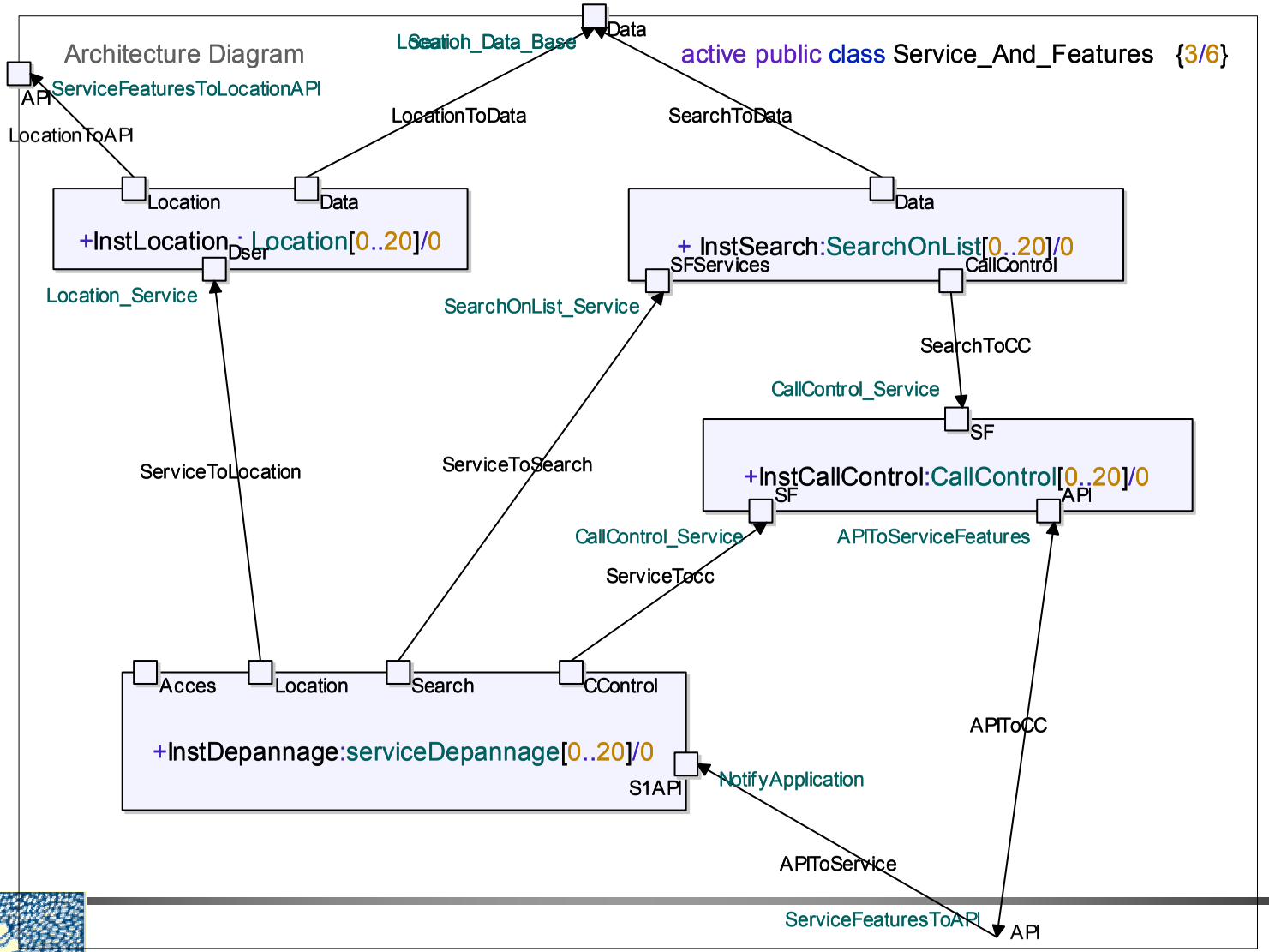
If time evolution is Under 1, then try another Destination party

Composite Modeling with LSC

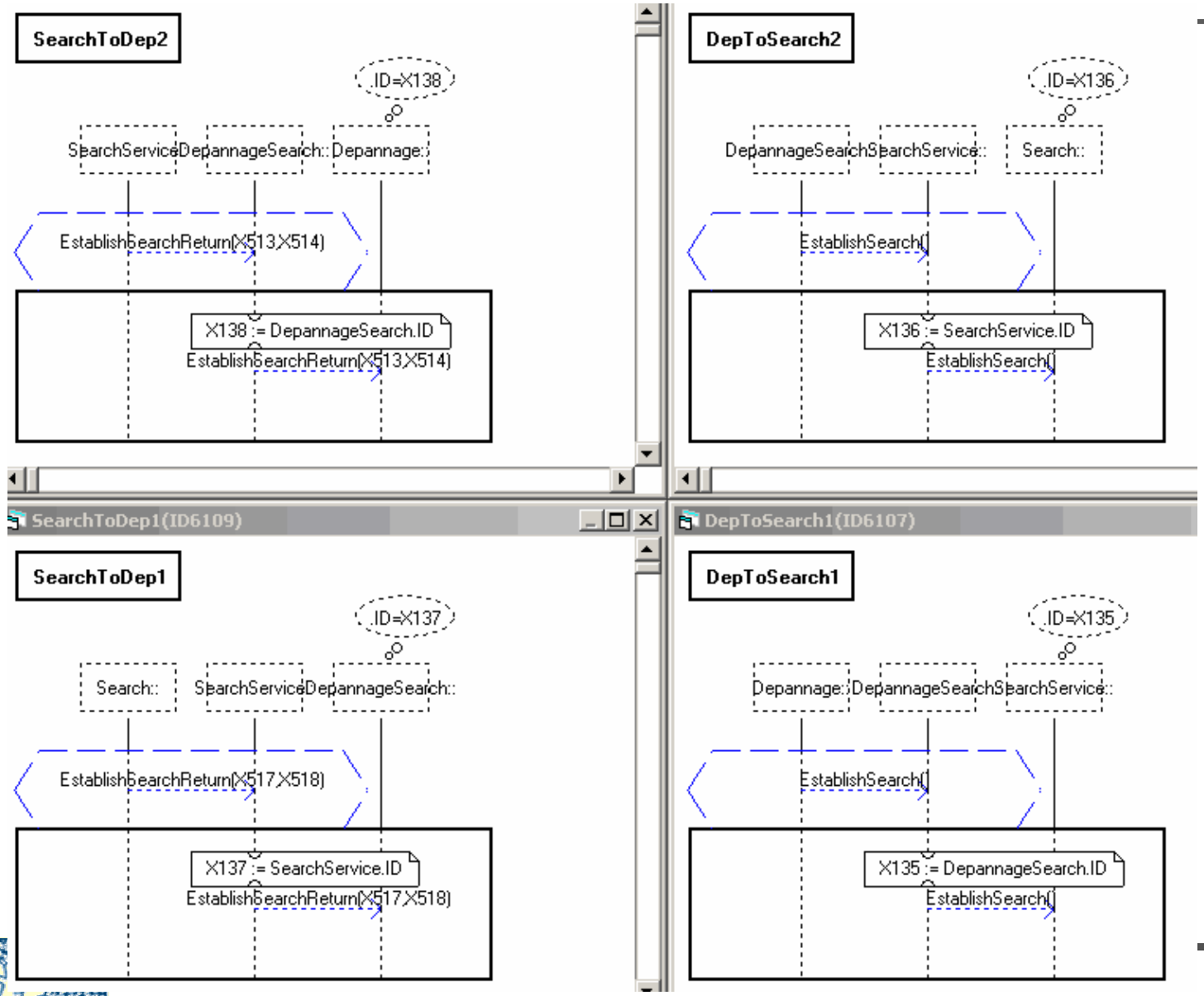
- **Based on UML2 architectural diagram**
  - Express Requirement (Existential LSC) from the system (composite) point of view
  - Static description of embedded components and connectors
  - Express the dynamic behaviour (assumptions) on connectors (Universal LSCs),
    - Time constraints, Delays , Message losses on protocols and communications (with probabilities)
  - Express the environment potential behaviours (Universal LSCs)
- **Great use of symbolic instances**
- **Remark: we did not develop graphical user interface**

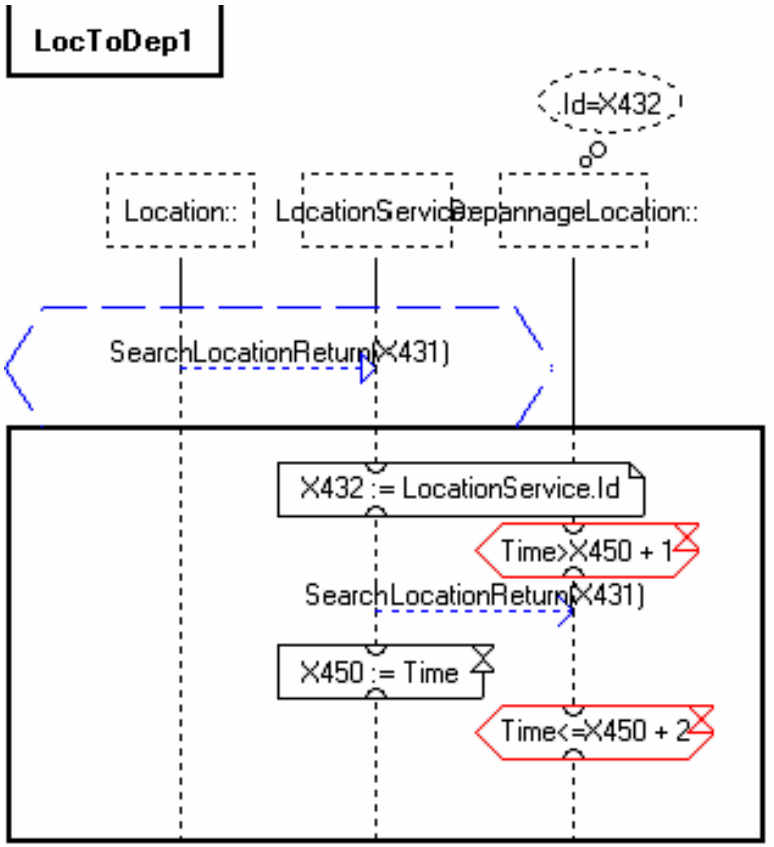
The Composite: an architectural view



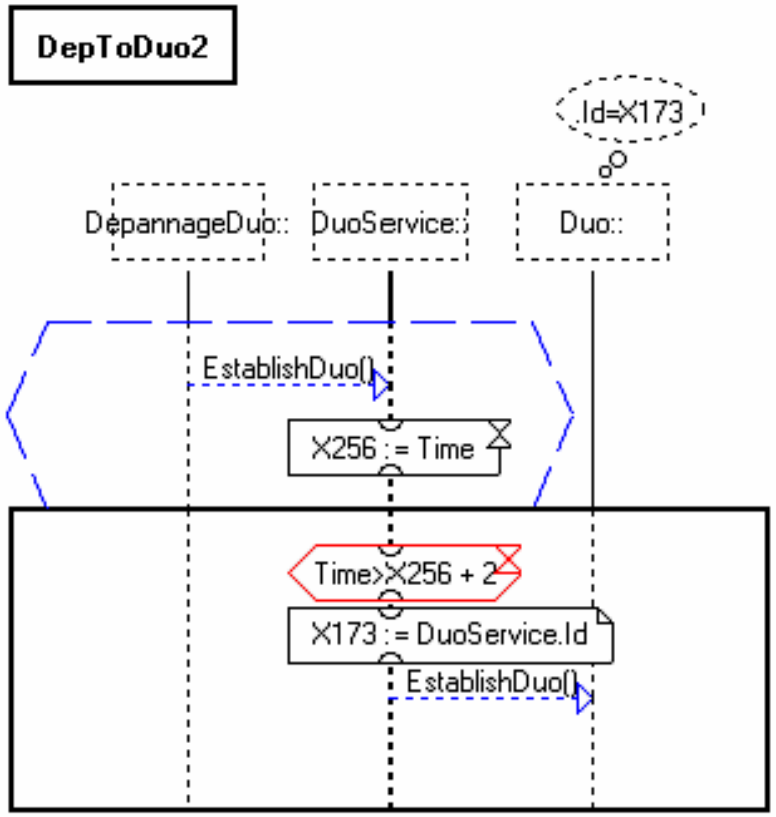
Simple Connections



Connections with Delay



On the connector (by signal)

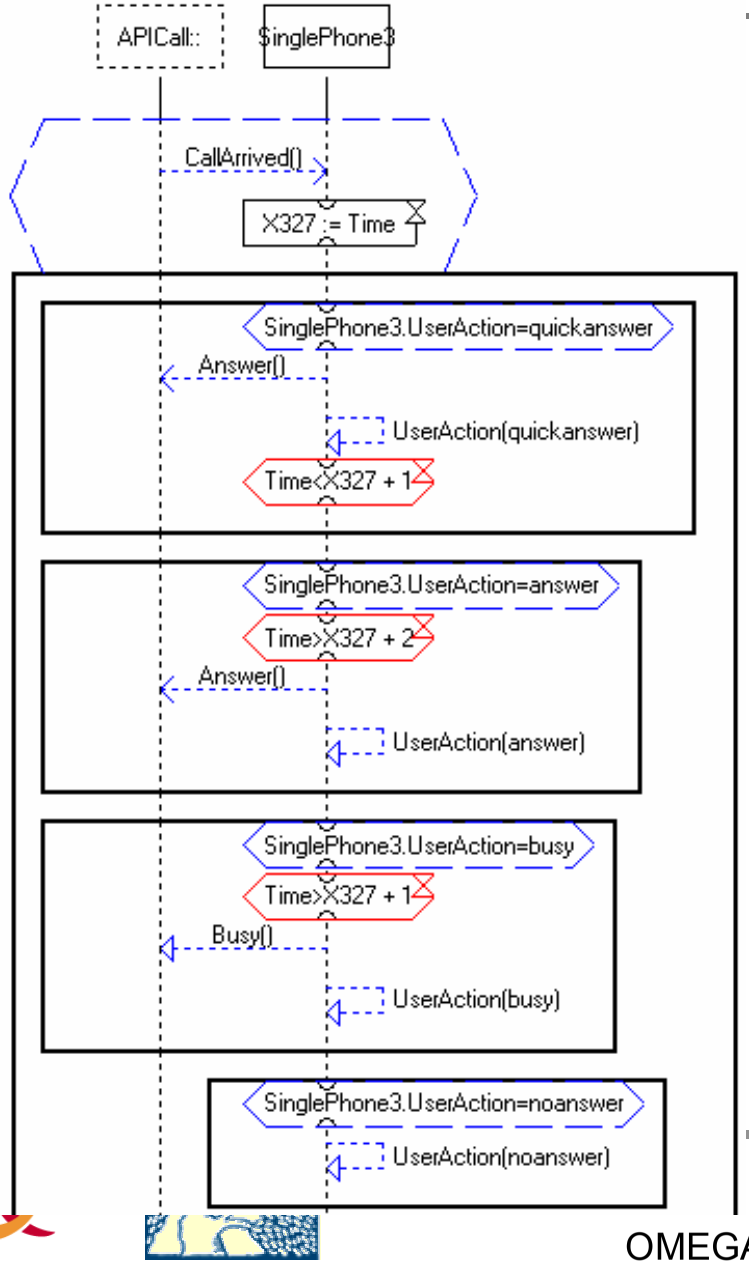


On a port/interface

The delays could depend on the signal, the parameters, the history, etc

We may introduce signal loss and loss rates

The Environment: GSM user



Answer before T + 1

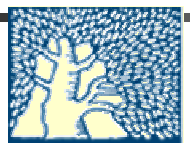
Answer after T + 2

Busy after T + 1

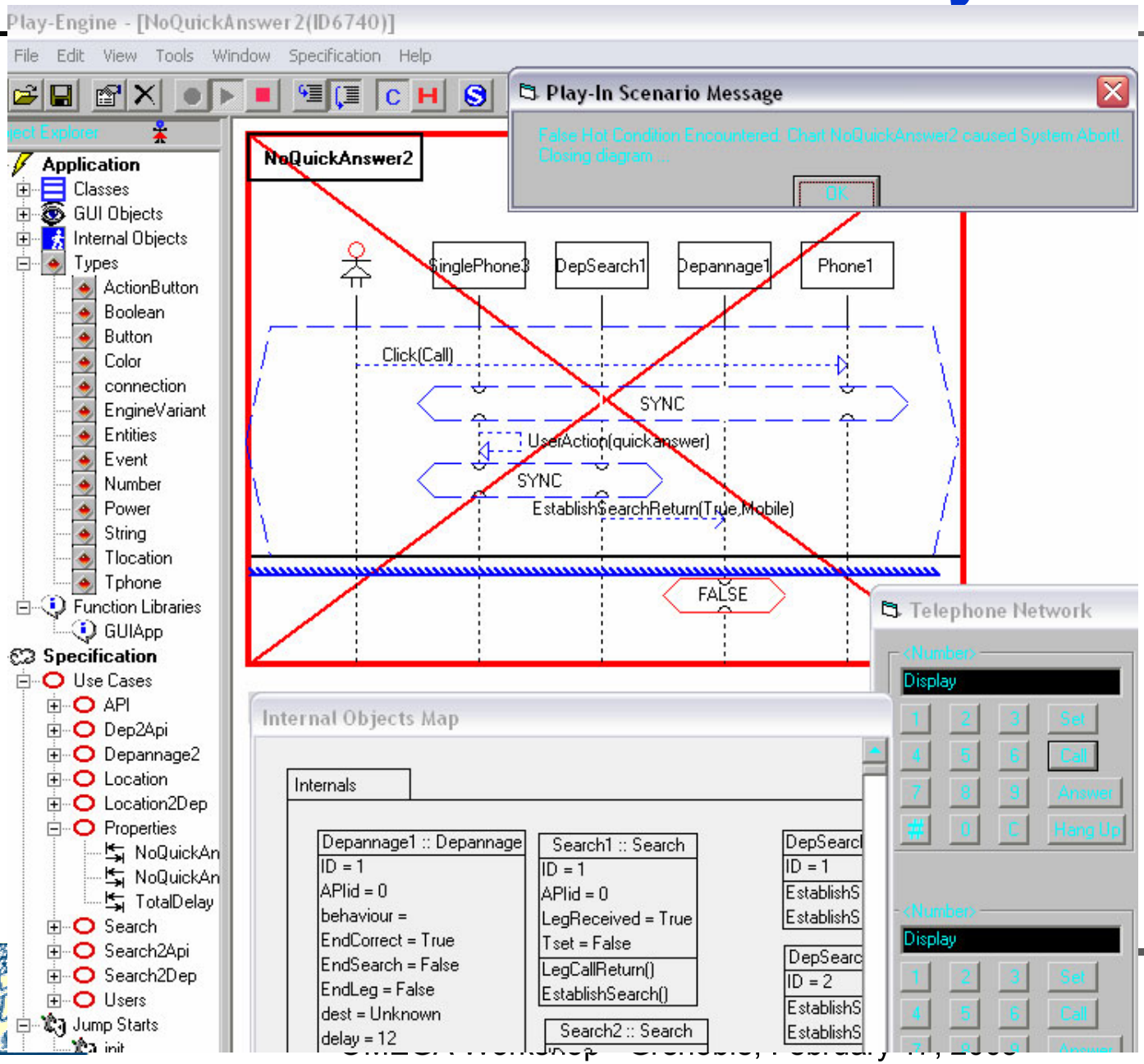
Animation of LSC Model

- **Animation for a better understanding of the model execution**
  - Executing different scenarios/configurations
  - Recording the traces
  - Observing the existential LSCs

- **On the use of LSCs and the Play-In tool**
  - LSC is well-suited for the expression of requirements and dynamic assumptions on different parts of the model (components, connectors, system)
    - A graphical language accessible to non-specialist in formal theory
    - Great expressivity
    - Great flexibility

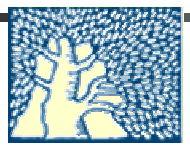



Play-Out scenario



Formal Verification with smart Play-Out

- **Principe of the Play-Out Engine**
  - To find **one** execution that satisfies a existential LSC (the property)
- **Principe of formal verification**
  - Check that, for **all** executions, a requirement is respected (not violated)
- **Principe of the verification method**
  - Express the requirement by a property (an existential LSC) that violates it
  - Run the Play-Out engine
  - If the property is satisfied, then the requirement is violated (for at least one execution path)
  - If the property is not satisfied, the requirement is verified for all execution paths

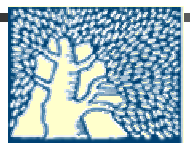



Formal Verification with smart Play-Out

Time requirements

- **We mainly want to verify requirements such as:**
  - D1 <Time_Duration < D2
  - Time_Duration is the end-to-end execution delay

- **Example: Time_Duration < D2**
  - We express the property by a existential LSC with a condition
  - Time_Duration ≥ D2
  - Running the Play-Out Engine, the property is not satisfied



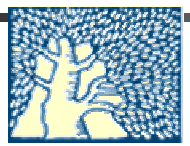
Formal Verification with smart Play-Out

- **Restrictions on the smart Play-Out**
  - No symbolic instances
  - Multiple parameters in signal
- **State-explosion problem**

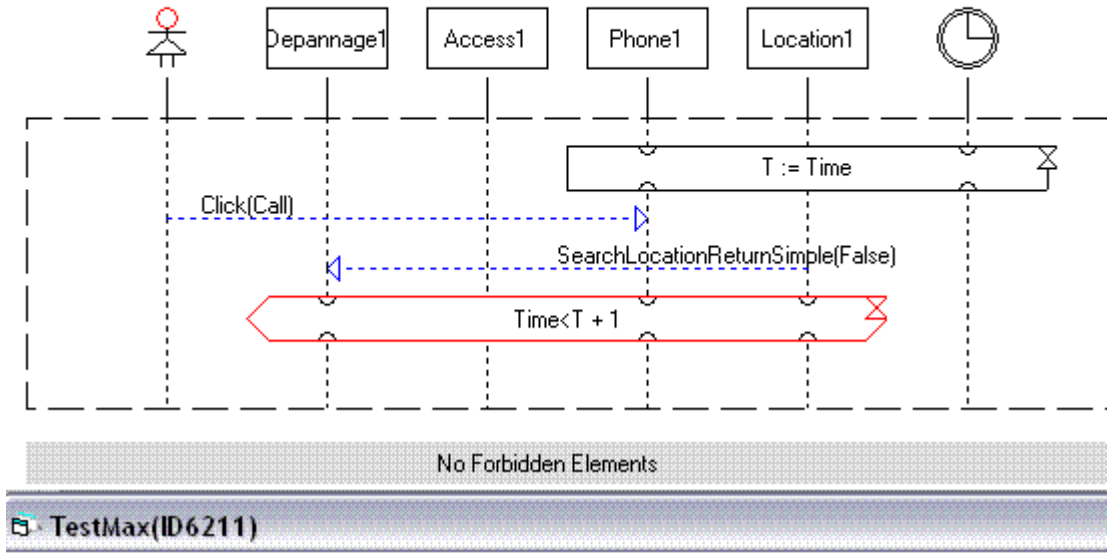
➢ **Needs to make several models**
  - ➢ Focusing on specific parts of the model (more complex/critical)
  - ➢ Reducing non determinism
    - ➢ Use of configuration
  - ➢ Feedback on the complete model

❖ **Very good compromise between formal techniques and readability**

✓ **Three examples**

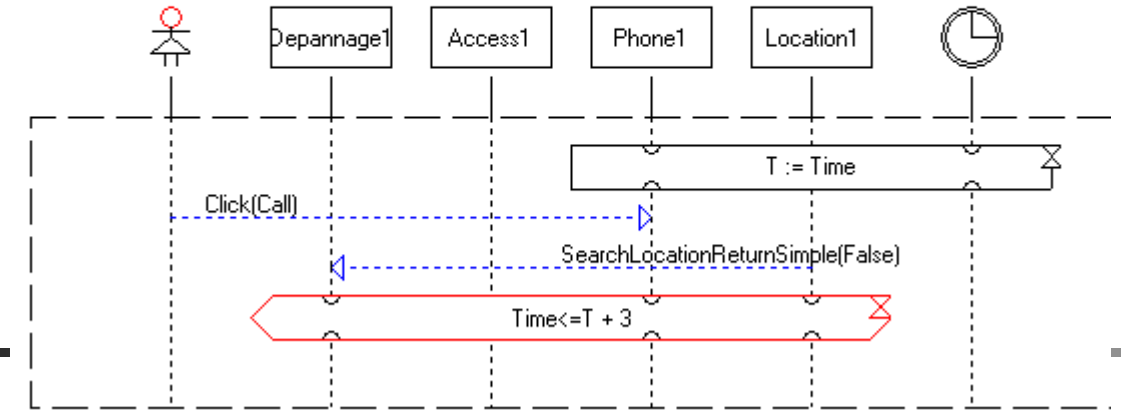



1 Existential LSC



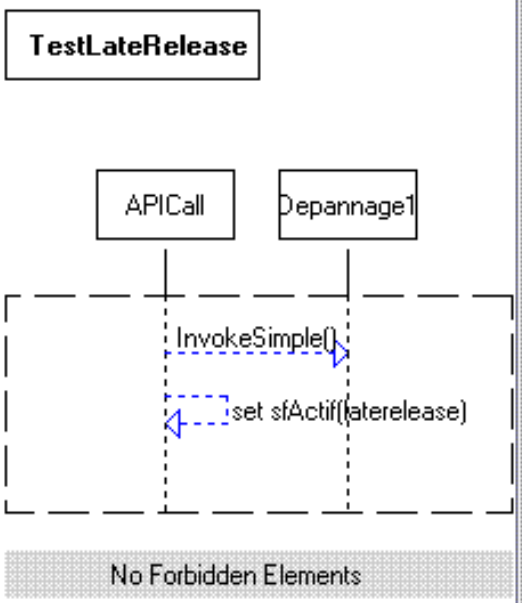
Not satisfied

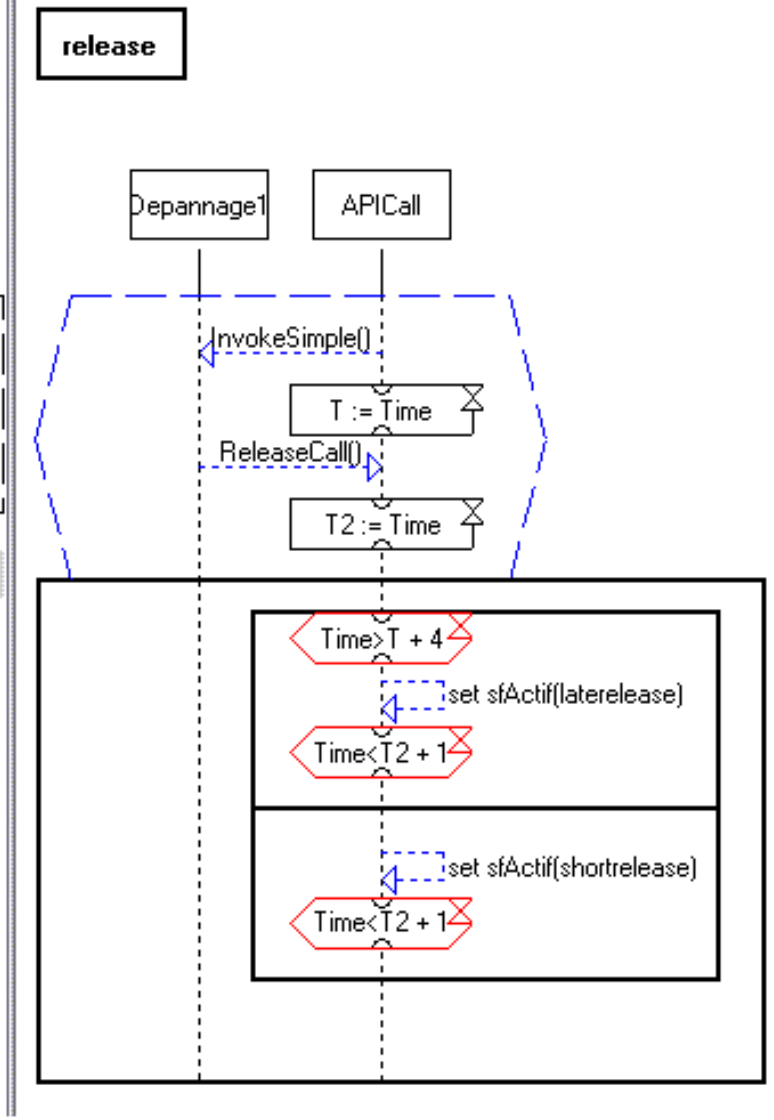
For all execution Time_Duration will be more (or equal) than 1



Satisfied

2 Another Time Requirements

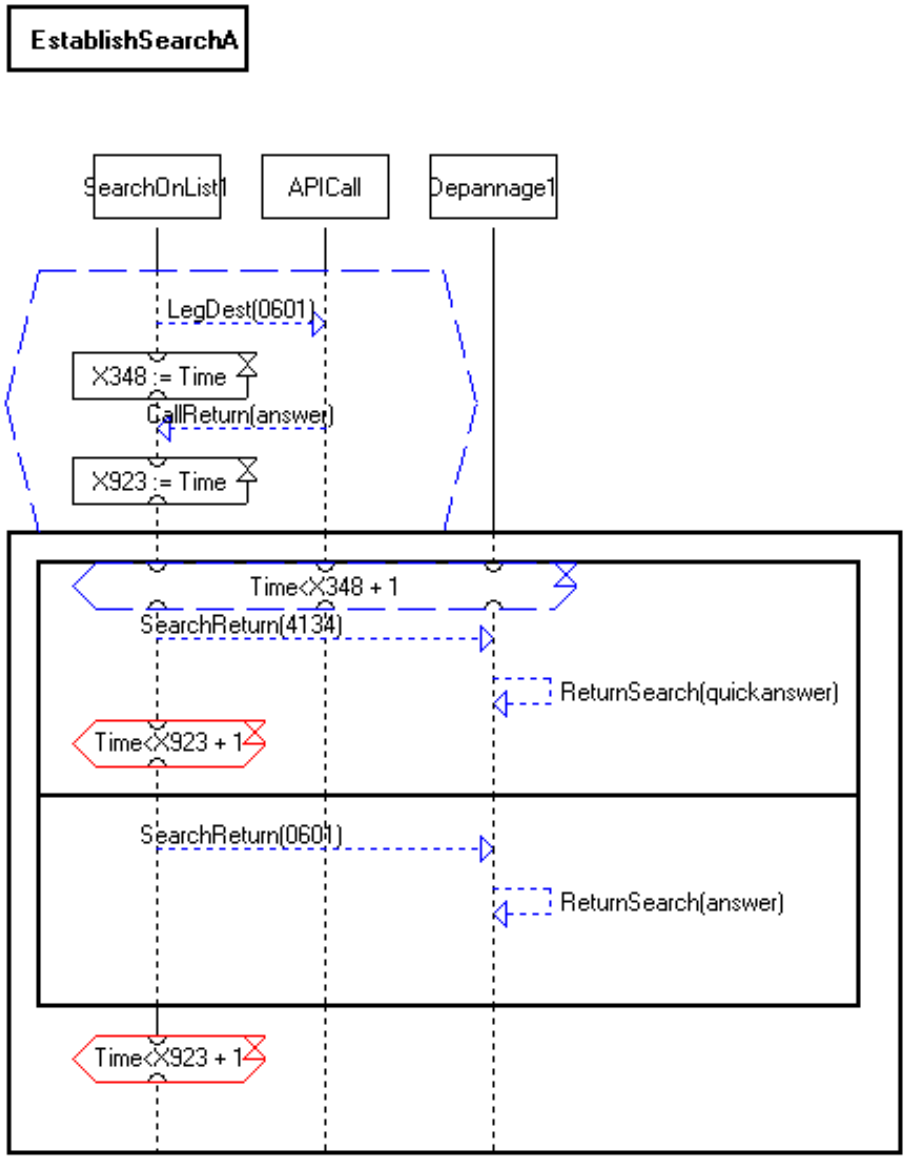




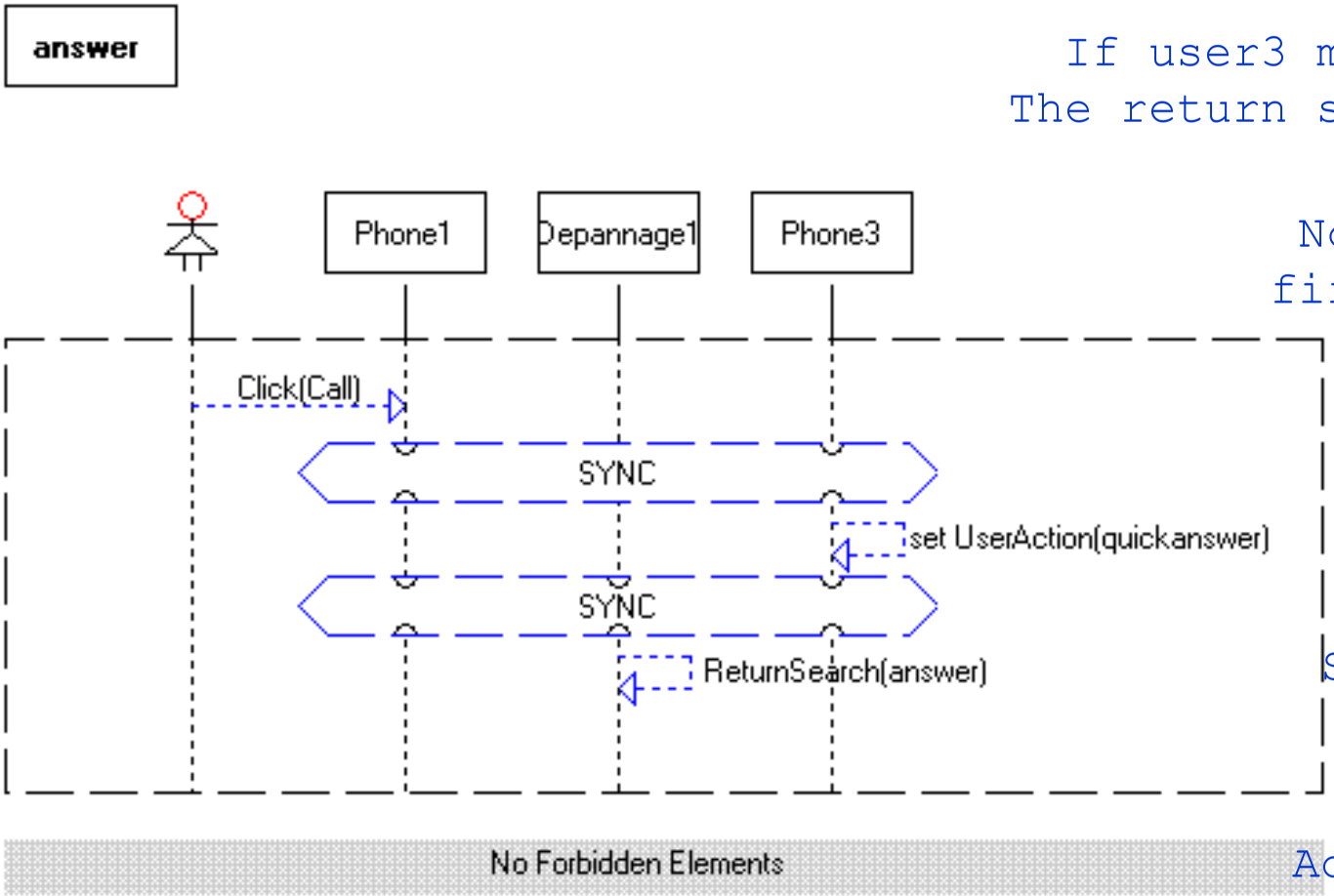
Not satisfied

Always, the end-to-end delay will be Less than 4

3 Search Component



3 Search Property



If user3 makes *quickanswer*,
The return should not be *answer*

No satisfied with first configuration

Satisfied with the second configuration:

Adding a new feature (Forward)