From AUML to ICs

Preliminary considerations about automatic translation of AUML diagrams into Social Integrity Constraints

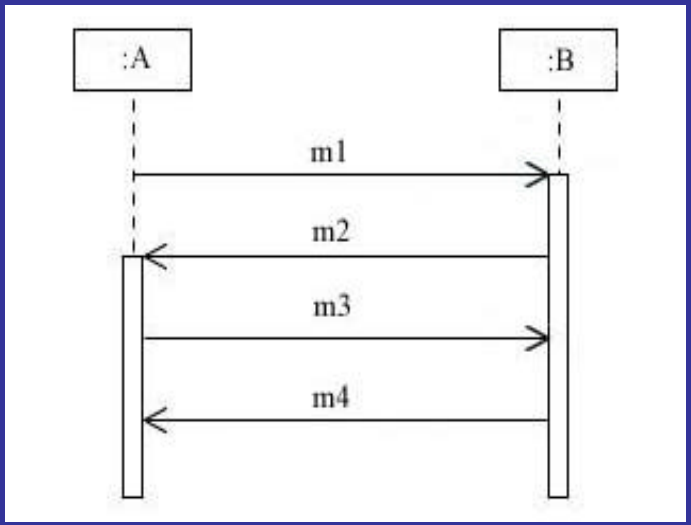
Goal

- Define an algorithm for automatic translation of AUML diagrams into ICs

- Inspiration drawn from the paper: M. Baldoni, C. Baroglio, A. Martelli, V. Patti, and C. Schifanella. Verifying protocol conformance for logic-based communicating agents. In J. Leite and P. Torroni, editors, *Proc. of Fifth International Workshop on Computational Logic in Multi-Agent Systems, CLIMA V*, pages 82-97, Lisbon, Portugal, September 2004

Idea

- There is a direct relation between the operator *message arrow* and the concept of "*social relevant event*" in the SOCS framework



$$H(m1, T_1) \rightarrow E(m2, T_2) \wedge T_2 > T_1.$$

$$H(m1, T_1) \wedge H(m2, T_2) \wedge T_2 > T_1 \rightarrow E(m3, T_3) \wedge T_3 > T_2.$$

$$H(m1, T_1) \wedge H(m2, T_2) \wedge H(m3, T_3) \wedge T_2 > T_1 \wedge T_3 > T_2 \rightarrow E(m4, T_4) \wedge T_4 > T_3$$

Freedom degrees…

- An expectation about an event (a message) is generated if and only if ***all*** the precedent events happened correctly. No expectations are generated otherwise.

- Hence agent B can freely utter the message *m2*. This is not forbidden, and it does not generate future expectations.

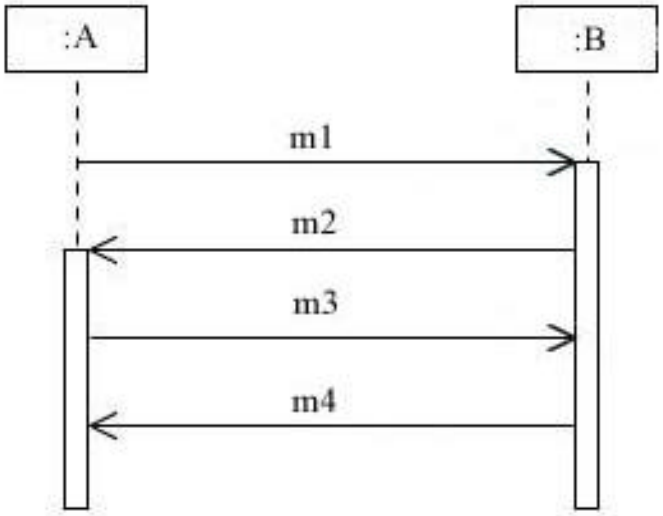
Lowering the freedom (1)

It is possible to restrict such freedom (? more compliant with AUML), in two ways:

1. By adding a meta-constraint specifying that “everything that is not expected to happen is not permitted”. If something happens, and it is not expected, then a violation can be detected.

Lowering the freedom (2)

2. By inserting backward constraints, i.e. constraints about past events:



...

$$H(m1,\ T_1) \wedge H(m2,\ T_2) \wedge T_2 > T_1 \rightarrow E(m3,\ T_3) \wedge T_3 > T_2.$$

$$H(m2,\ T_2) \rightarrow E(m1,\ T_1) \wedge T_2 > T_1.$$

...

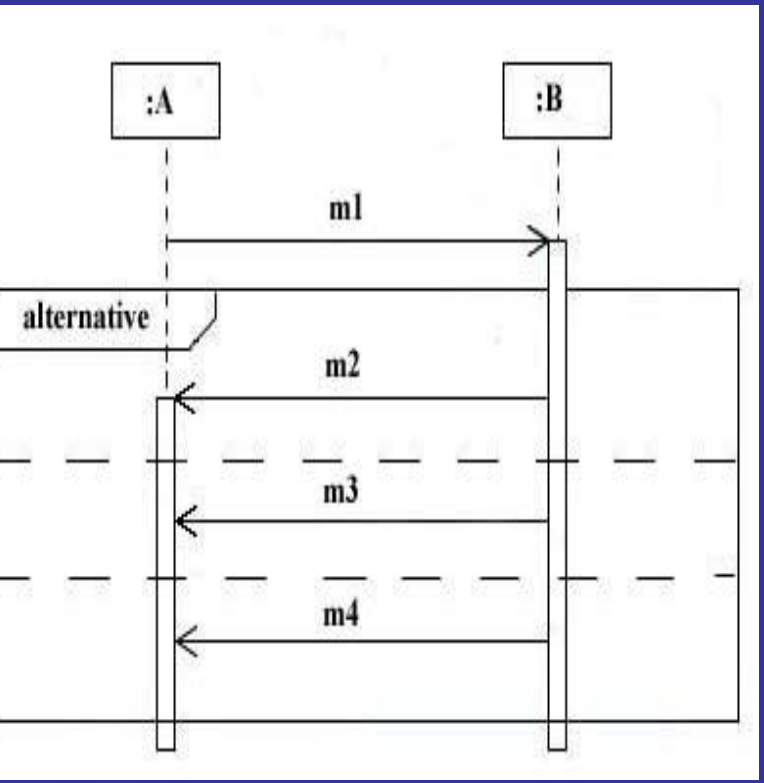
AUML Elements

- Until now, we have implemented some AUML elements:
  - Alternative Interaction Operator
  - Loop Interaction Operator (depending on the loop condition)
  - Stop operator (Interaction Termination)
  - Protocol combination (implemented via recursion)

Alternative Operator

- The splitting phase is easily implemented by having some expectations in OR
- Each alternative path is implemented as a single protocol (independent of other alternative paths)
- The resulting protocol is given by the union of the ICs generated for each alternative path

Alternative Operator (example)



H(m1, T_1)?

E(m2, T_2) ∧ EN(m3, T) ∧ EN(m4, T_4)

∨

E(m3, T_3) ∧ EN(m2, T_2) ∧ EN(m4, T_4)

∨

E(m4, T_4) ∧ EN(m2, T_2) ∧ EN(m3, T_3)

Problems with AUML (1)

- It seems that nothing has been done since the working draft dated 2-7-2003.
- Some elements do not have a semantic. In fact sometimes the semantic is expressed with the acronym "**TBD**" (Google says: "To Be Done")

Problems with AUML (2)

- It is not clear what can/could/must be specified about the content of the messages.
- Quite often it is necessary to express constraints about the content of a message. AUML does not address this issue.
- However, it is not forbidden (?) to express such constraints.

Problems with AUML (3)

- Due to the loose specifications of the standard, it could possible to combine several AUML elements in “funny” ways.
- The meaning of the resulting diagrams can be very ambigous…

Tools for AUML

- A further problem is given by the absence of proper tools for defining AUML diagrams.
- A possible solution could be the “re-use” of UML2 tools. But there is a certain delay also in the adoption of the UML2 formalism.
- There is no standard for the low-level representation of UML2 diagrams (Rose propose a proprietary standard *de facto*, while OMG pushes for a standard *de jure*)

What has been done until now

- A very rough and preliminary version of the algorithm has been defined. It tackles only a few elements (alternative, loop, termination, message, protocol combination).
- The algorithm has been implemented as a Java class that parses an XML tree. The definition of the XML structure, in a first implementation, has been “invented”.
- A preliminary study of all the tool available nowdays for AUML/UML2 has been conducted

Future steps…

To choose a tool and to integrate our algorithm with such a tool. Probably (but not for sure) we will focus on Ingenias, an open source AUML tool implemented by the Computer Science department of the University of Madrid

Main algorithm

- Given:
  - ICsSet: The set of the Social Integrity Constraints generated by the algorithm (initially empty)
  - HapSet: The set of happened events (initially empty)
  - CurrentOperator: the next operator in the sequence diagram (initially set to the first operator)

Main algorithm

1. for each specific operator, generates a new proper set of constraints, called ICs_{New}
2. $ICs = ICs \cup ICs_{New}$
3. HapSet