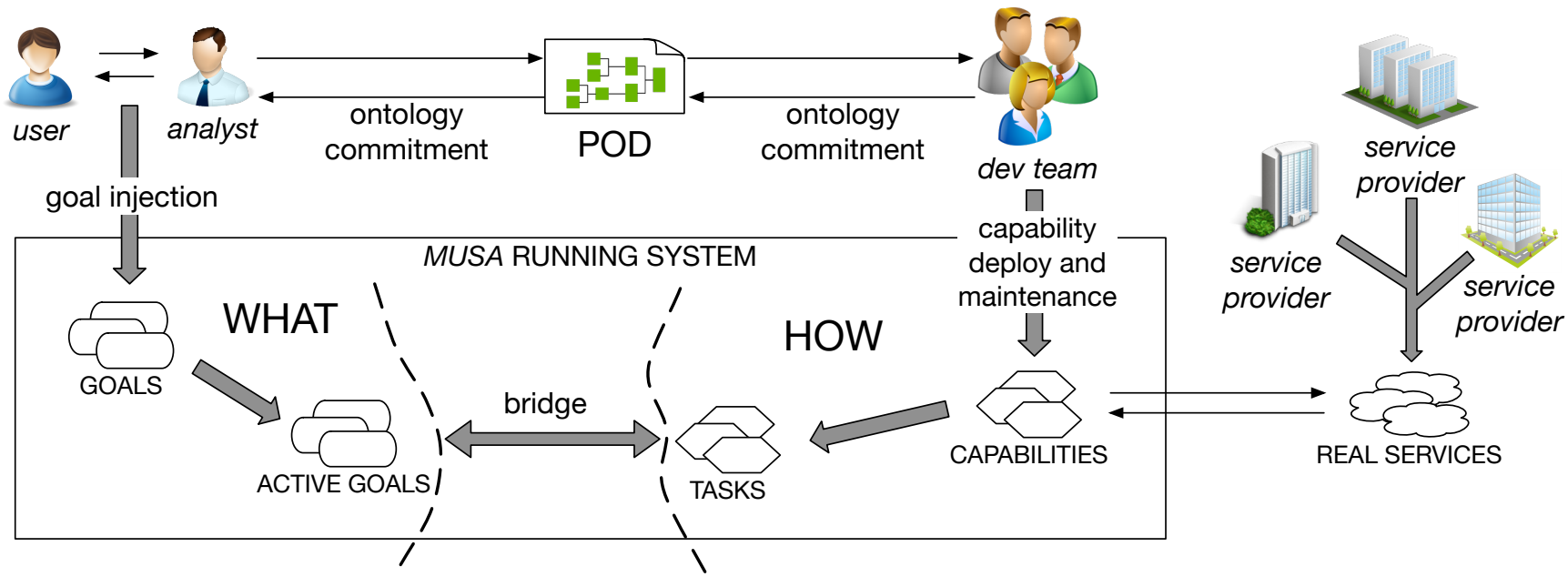
SEAMS 2015, Florence, May 18-19

From Means-End Analysis to Proactive Means-End Reasoning

Luca Sabatucci and Massimo Cossentino

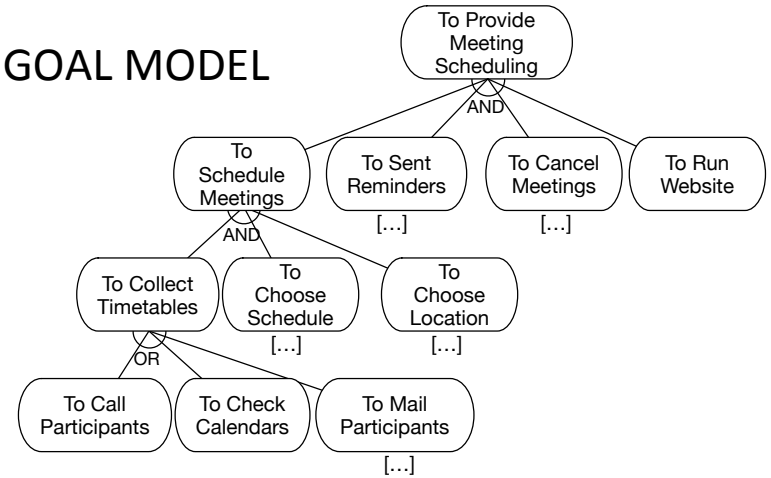
The Vision

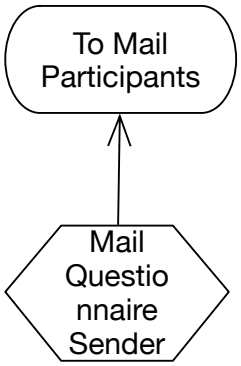

user
analyst
ontology commitment
POD
ontology commitment
dev team
service provider
service provider
service provider
service provider
goal injection
capability deploy and maintenance
MUSA RUNNING SYSTEM
WHAT
HOW
GOALS
ACTIVE GOALS
bridge
TASKS
CAPABILITIES
REAL SERVICES

Goal Oriented Requirements

- A goal is a ***state of affair*** that an actor wants to achieve

GOAL MODEL



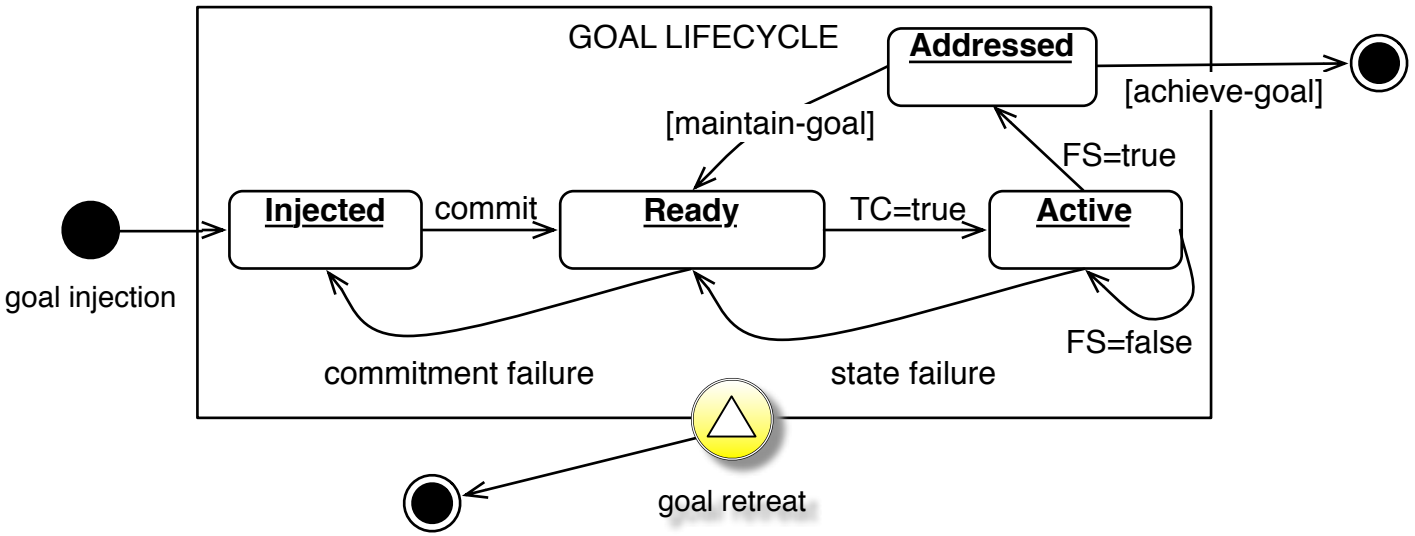


MEANS-END ANALYSIS

The State of the World

- A state of the world (W^t) is a dynamic object that describes the current "state of affair"
  - or better: what the system knows about
- We implement W^t by employing a set of semantically coherent first order logic facts.
- W^t describes a closed-world in which everything is not explicitly declared is assumed to be false.

Operative Implementation of Goal



- Goal's TC is the Condition that must hold in W^t in order the agent can actively pursue that goal.
- Goal's FS is the Condition that must hold in W^t in order the goal can be marked as addressed.
- GOALSpec is a language conceived to inject goal specifications in a human-friendly format

USER-GOAL_01

WHEN schedule(Usr,Meeting) THE system SHALL PRODUCE canceled(Meeting) OR confirmed(Meeting)

USER-GOAL_02

WHEN pending(Meeting) AND meeting_datetime(DT) AND attendee(Meeting,A) THE system SHALL PRODUCE notified(A,Meeting,DT)

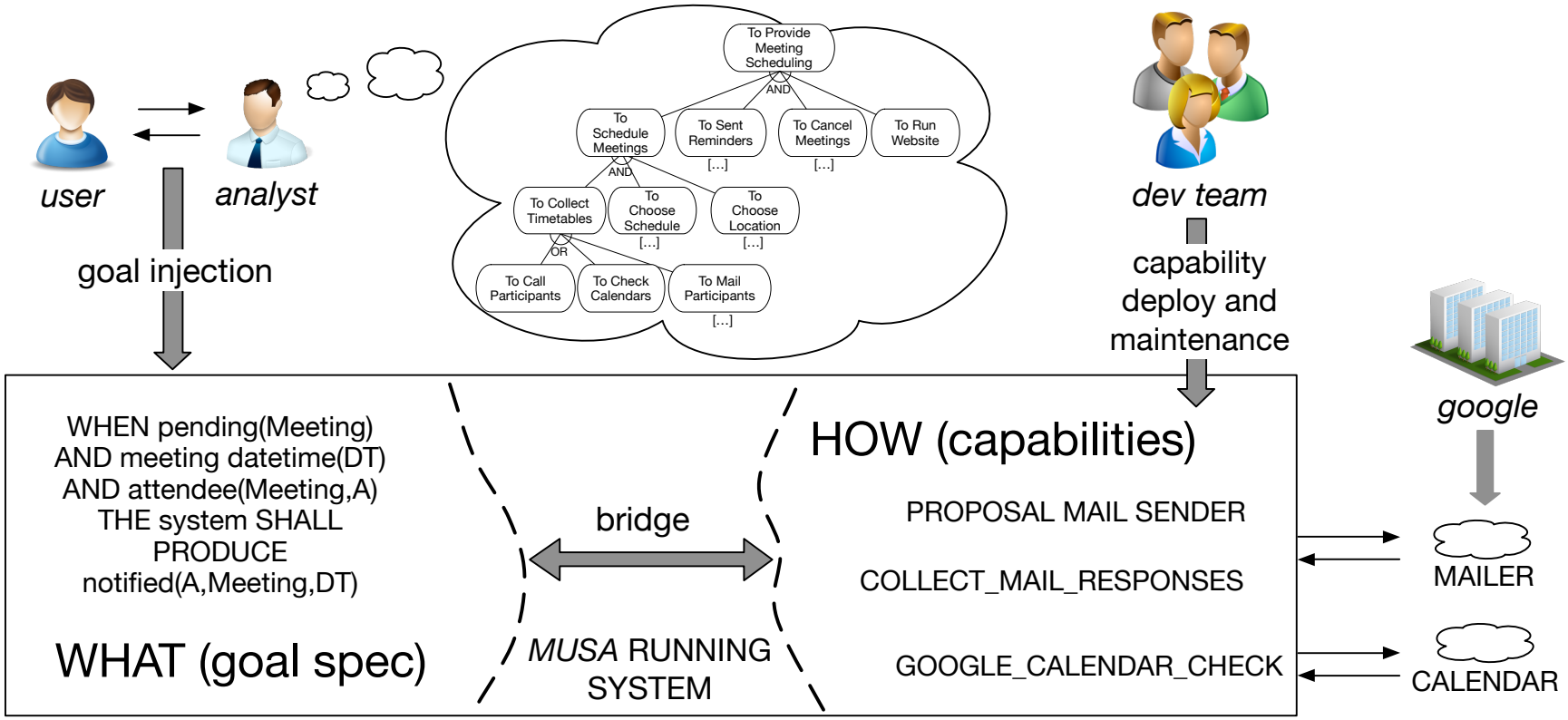
AI-Style CAPABILITIES

- The system owns a set of capabilities, i.e. atomic and self-contained actions
- The effect of a capability is an endogenous evolution of W^t
- The system is aware of its capabilities
- and it is aware of 'when' and 'how' to use a capability in order to address a desired result

Name	PROPOSAL_MAIL_SENDER
InputParams	QUESTION : TEXT, RESPONSEID: STRING USERMAIL : STRING
OutputParams	NONE
Constraints	$format(UserMail,$ $RFC\_5322\_Address\_Specification)$
Pre-Condition	$email(Usr, UserMail)$
Post-Condition	$notified(Question, Usr)$
Evolution	$evo = \{add(notified(Msg, Usr)),$ $add(mailed(UserMail, Question))$ $add(questioned(Usr, ResponseId))\}$

ABSTRACT DESCRIPTION OF A CAPABILITY

Bridging WHAT and HOW



The PROACTIVE MEANS-END REASONING
is the problem of
finding the minimal set of capabilities (called PMR Solution) that can fully address a goal model, given the current W^t.

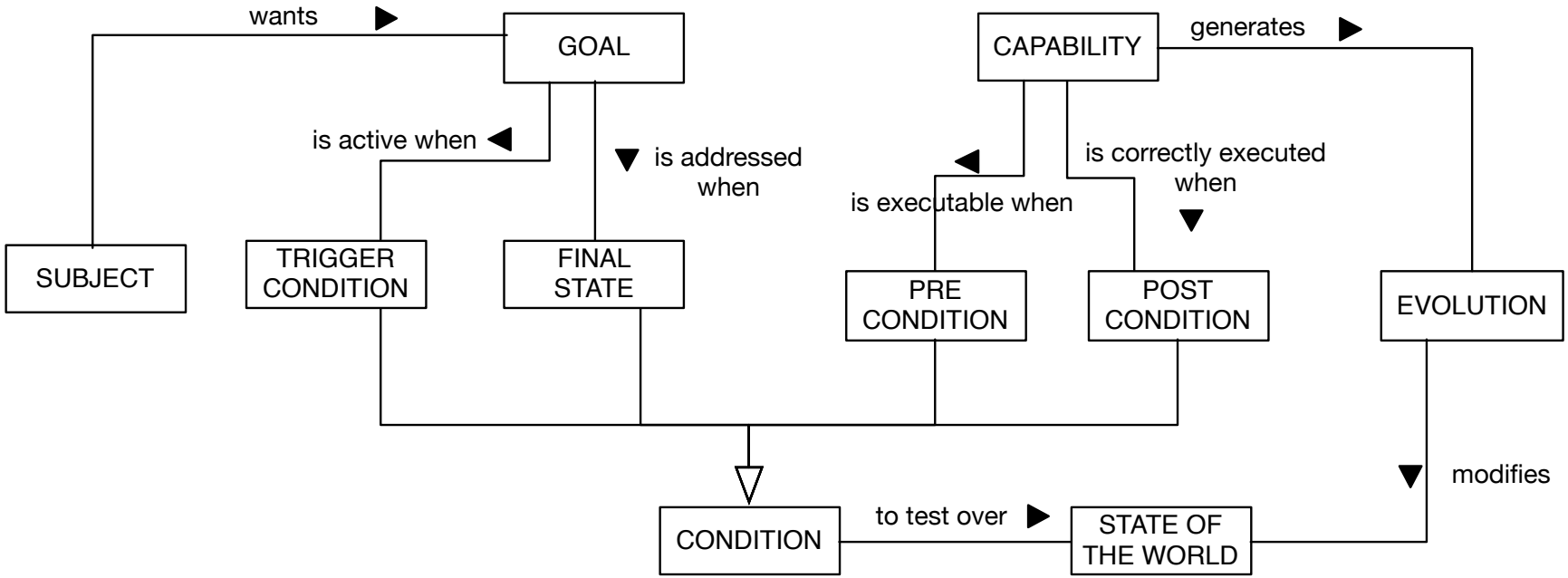
The PMR Solution

- The Proactive Means-End Reasoning is different from
  - A scheduling problem: it does not require an exact timing of the activities
  - A planning problem: it does not require to create a plan for executing the activities
- The system will contextually evaluate which capability to use, when, and how.
  - The same capability in the PMR_Solution will eventually used 0..n times

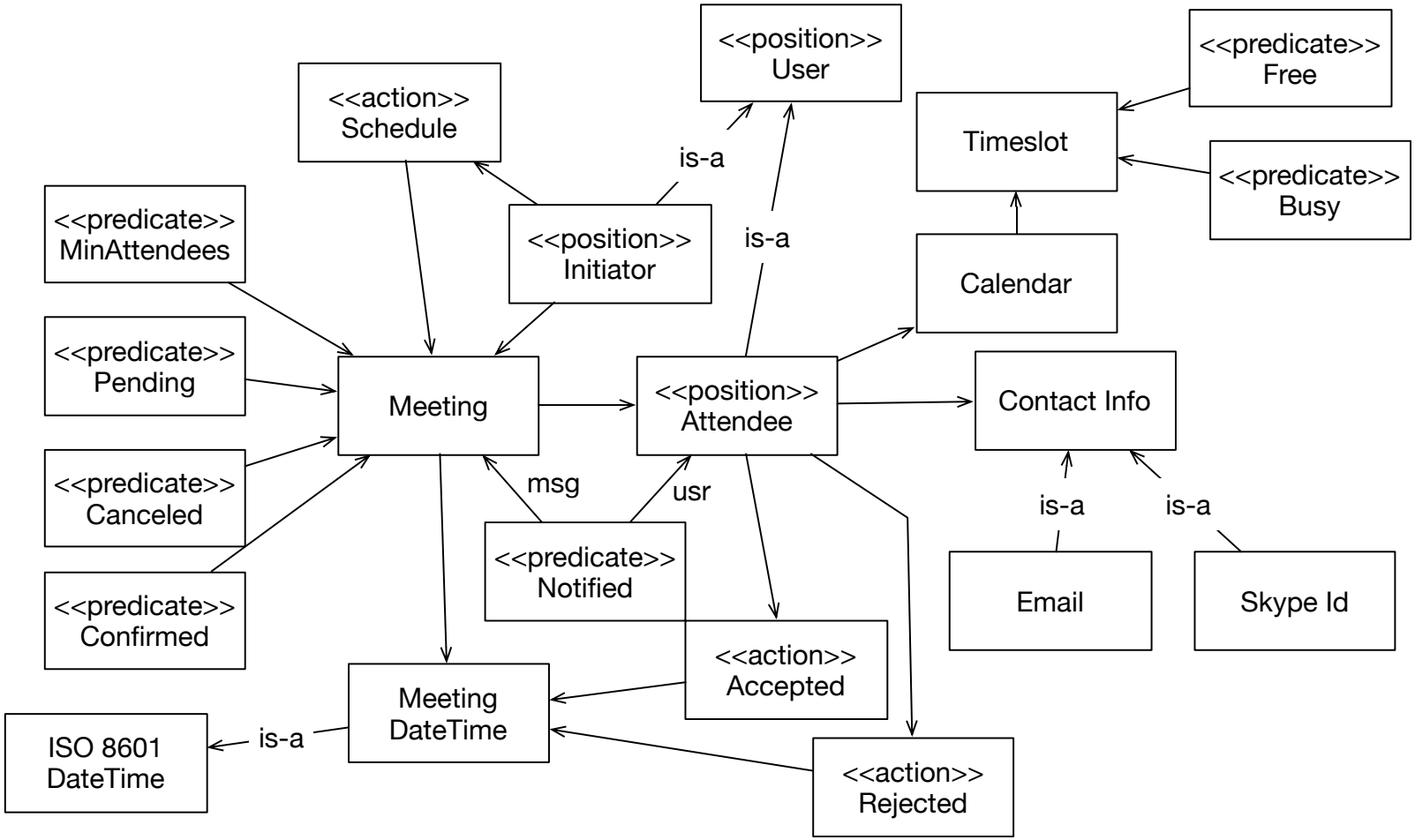
The proposed algorithm

- It is based on the ability to discover if a capability can be used for addressing a goal (or contributing to)
- The principle is that of matching Goal's TC/FS and Capability's Pre/Post/Evolution
- This is possible if goals and capabilities share
  - The same formalism
  - The same background ontology

The State of World as Common Formalism



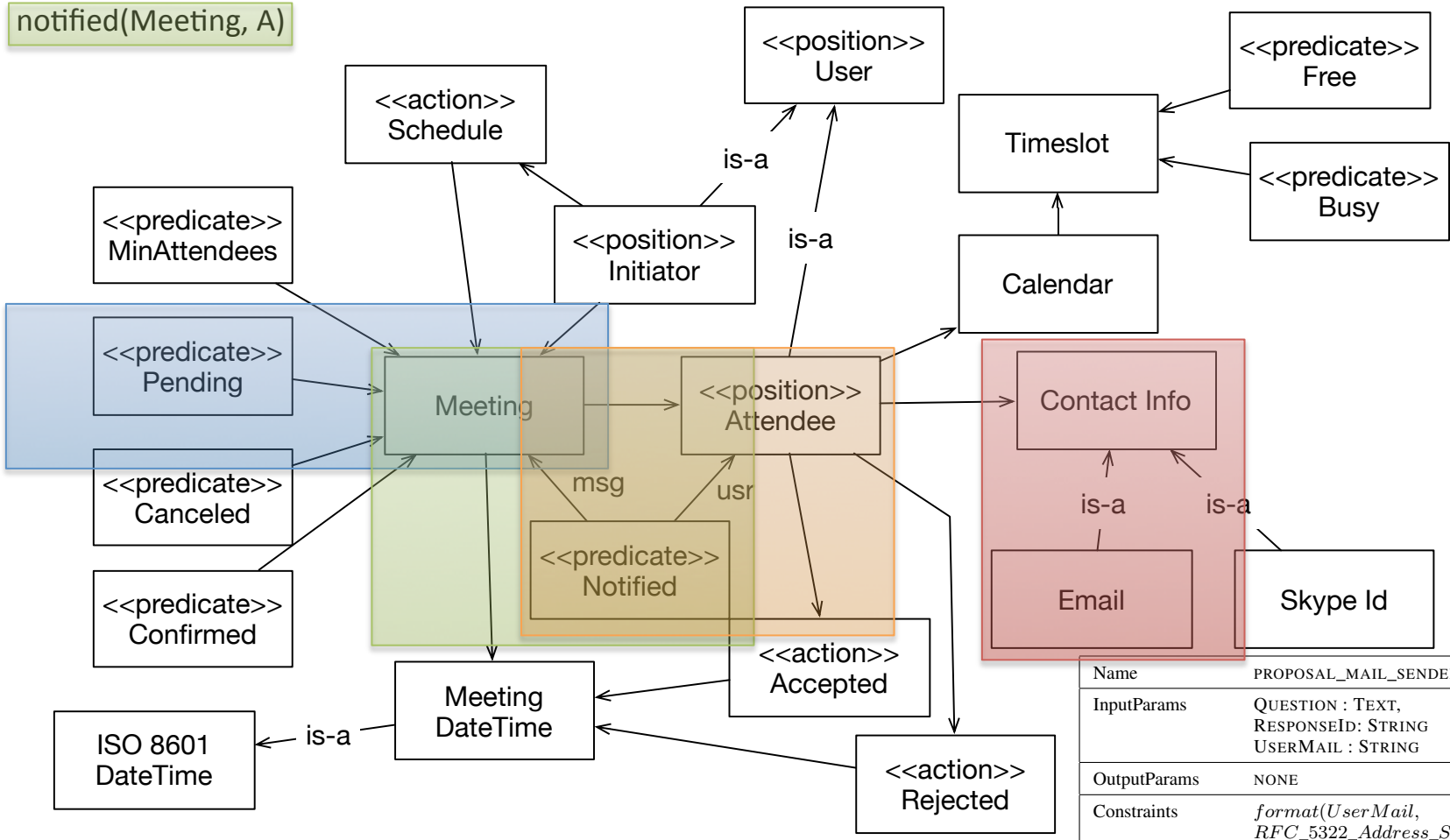
The Ontology as Common Background



Common Background (II)

USER-GOAL_01

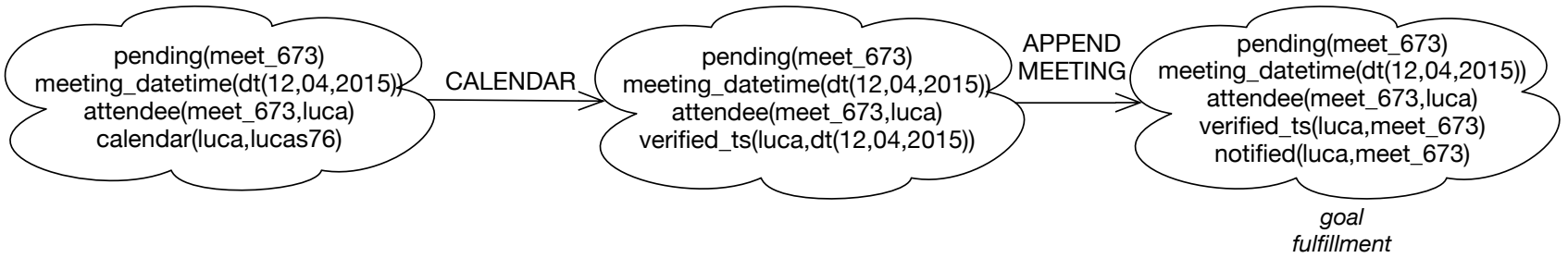
WHEN pending(Meeting) AND meeting datetime(DT) AND attendee(Meeting,A) THE system SHALL PRODUCE notified(Meeting, A)



Name	PROPOSAL_MAIL_SENDER
InputParams	QUESTION : TEXT, RESPONSEID: STRING USERMAIL : STRING
OutputParams	NONE
Constraints	$format(UserMail, RFC\_5322\_Address\_Specification)$
Pre-Condition	$email(Usr, UserMail)$
Post-Condition	$notified(Question, Usr)$
Evolution	$evo = \{add(notified(Msg, Usr)),$ $add(mailed(UserMail, Question))$ $add(questioned(Usr, ResponseId))\}$

Planning-Like Space Exploration

USER-GOAL_01
WHEN pending(Meeting) AND meeting datetime(DT) AND attendee(Meeting,A) THE system SHALL PRODUCE notified(A,Meeting)



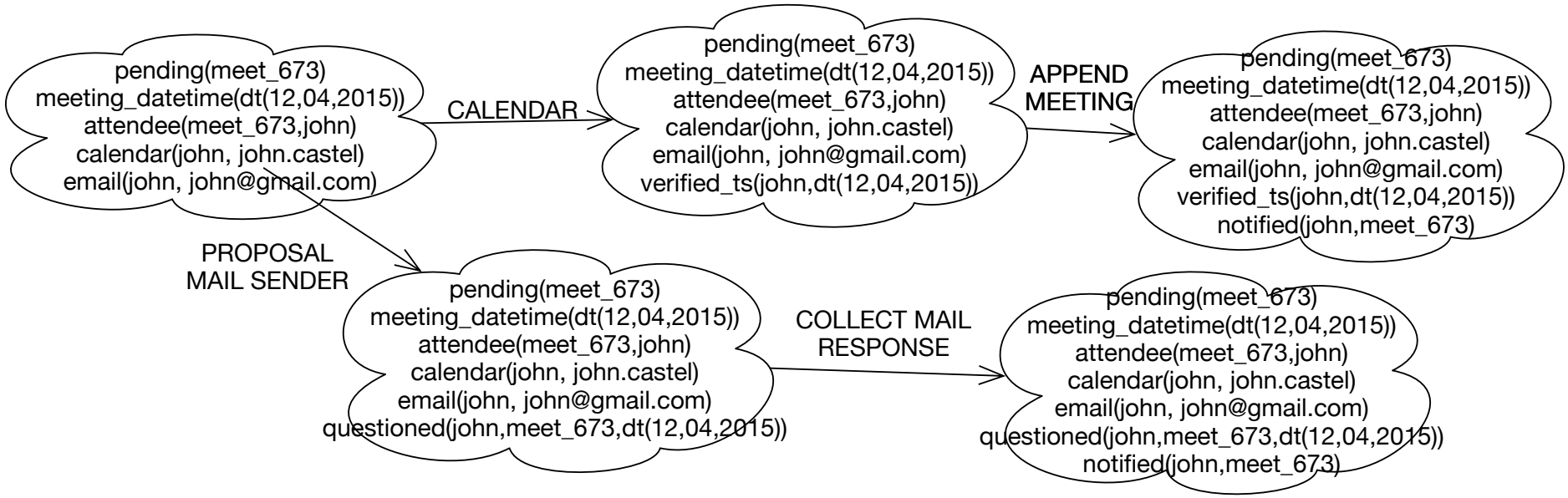
Name	**Calendar_Timeslot_Check**
Pre-condition	calendar(Usr,UserAccount)
Post-condition	free(Usr,TimeSlot) OR busy(Usr,TimeSlot)
Evolution	evo={ add(verified_ts(Usr,TimeSlot)) }

Name	**Append_Meeting**
Pre-condition	free(Usr,TimeSlot)
Post-condition	busy(Usr,TimeSlot)
Evolution	evo={ add(notified(Usr,Meeting)) }

Space Exploration (II)

USER-GOAL_01

WHEN pending(Meeting) AND meeting datetime(DT) AND attendee(Meeting,A) THE system SHALL PRODUCE notified(A,Meeting)

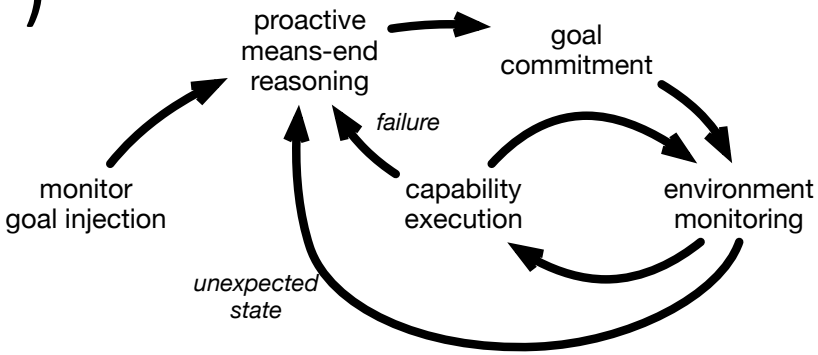


Name	**Proposal Mail Sender**
Pre-condition	email(Usr,MailAddress)
Post-condition	questioned(Usr,Meeting)
Evolution	evo={ add(questioned(Usr,Meeting)) }

Name	**Collect Mail Response**
Pre-condition	email(Usr,MailAddress)
Post-condition	accepted(Usr,Meeting) OR rejected(Usr,Meeting)
Evolution	evo={ add(notified(Usr,Meeting)) }

Final Remarks – Self Adaptation

- Self Adaptation is the result of a loop in which the Proactive Means-End Reasoning is executed every time (with different W^I)
  - New goal-model is injected
  - An existing goal changes
  - A capability fails:
    - software failure and exceptions
    - the generated W is different from the expected one
    - the connected resource is no more available
  - New capability is injected



Future Works

- The planning algorithm is inefficient
  - In some circumstances it requires an exponential time to complete.
  - We are planning to explore many strategies for improving it
    - SAT solvers, optimized planning and case base reasoning
- Scalability is limited
  - We are studying a better integration with a Cloud architecture (Open-Stack)
- To date the use of a static ontology enables the agent's
  - it is also a limit when capabilities/goals evolve one independently from the others.
  - In order to enable distributed development-teams, we are integrating linguistic techniques for dealing with
    - conceptual ambiguities and linguistics flaws, similarities and synonyms.

https://github.com/icar-aose/MUSA

Questions?

sabatucci@pa.icar.cnr.it