Semantics of Service Discovery and Binding

José Fiadeiro

(with Laura Bocchi and Antónia Lopes)

University *of* Leicester

Social complexity

- A major source of complexity in software-intensive systems is *social*:
  - Complex and dynamic/evolving interactions among heterogeneous parties (software components, devices, people)
  - Components are not necessarily 'big'
  - The major concern is in interconnecting the parties so that desired behaviour can emerge from their interactions, not so much in building software components

Services vs Components

- In CBD, software components are "taken out of a box" and **plugged** into a system (possibly with the addition of some "glue" code) to provide a "service" (see article by Broy et al, TOSEM February 2007)
- In SOC, each time a service is invoked, a different provider may be chosen to negotiate terms and conditions, and then the service is finally **bound** (see article by Elfatatry, CACM August 2007)

A bank will use components for calculating interests, charging commissions, etc, that it will use in different products (savings, loans, …)

The same bank is likely to rely on external courier services that are procured according to the delivery address, speed, cost, …

(social complexity)

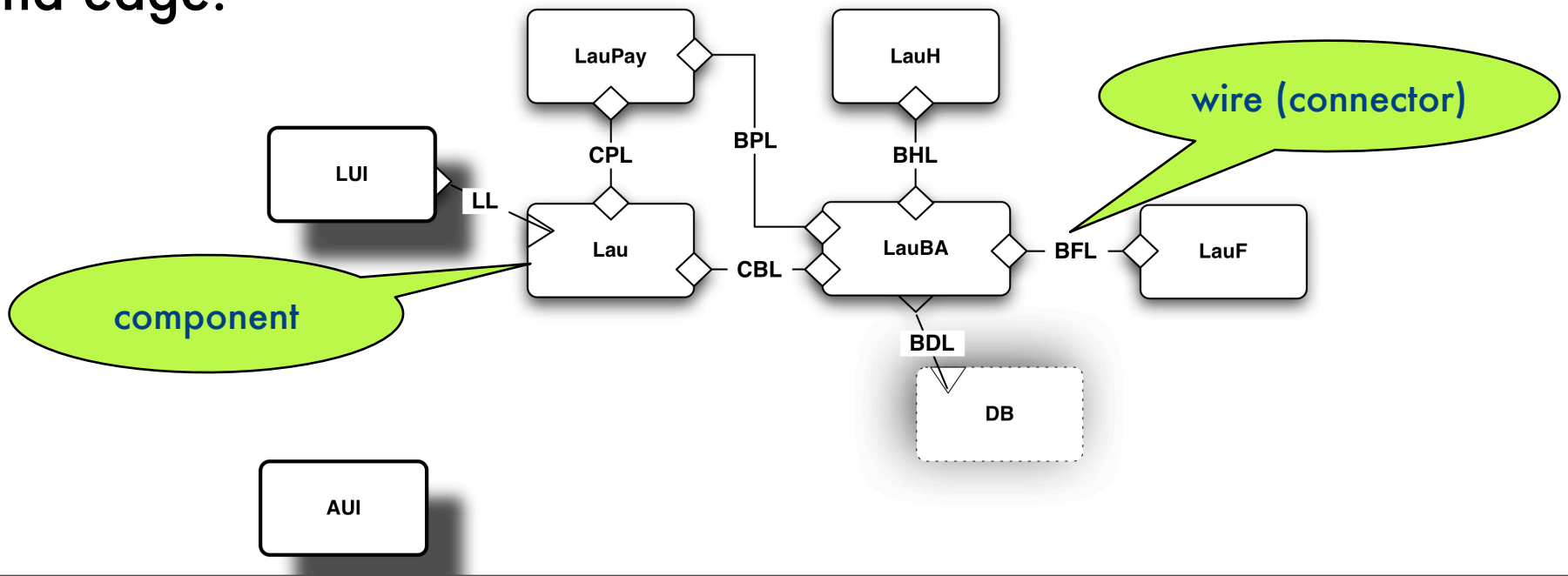
Service-oriented approach

- Services add a *social* layer of abstraction over a component infrastructure in sense that they structure the process of interconnection *(programmed interconnections)*.
  - Services should be published at a level of abstraction that corresponds to a real-world activity or recognisable business function (which is where social complexity can be understood)
  - Systems should be socially-reflective

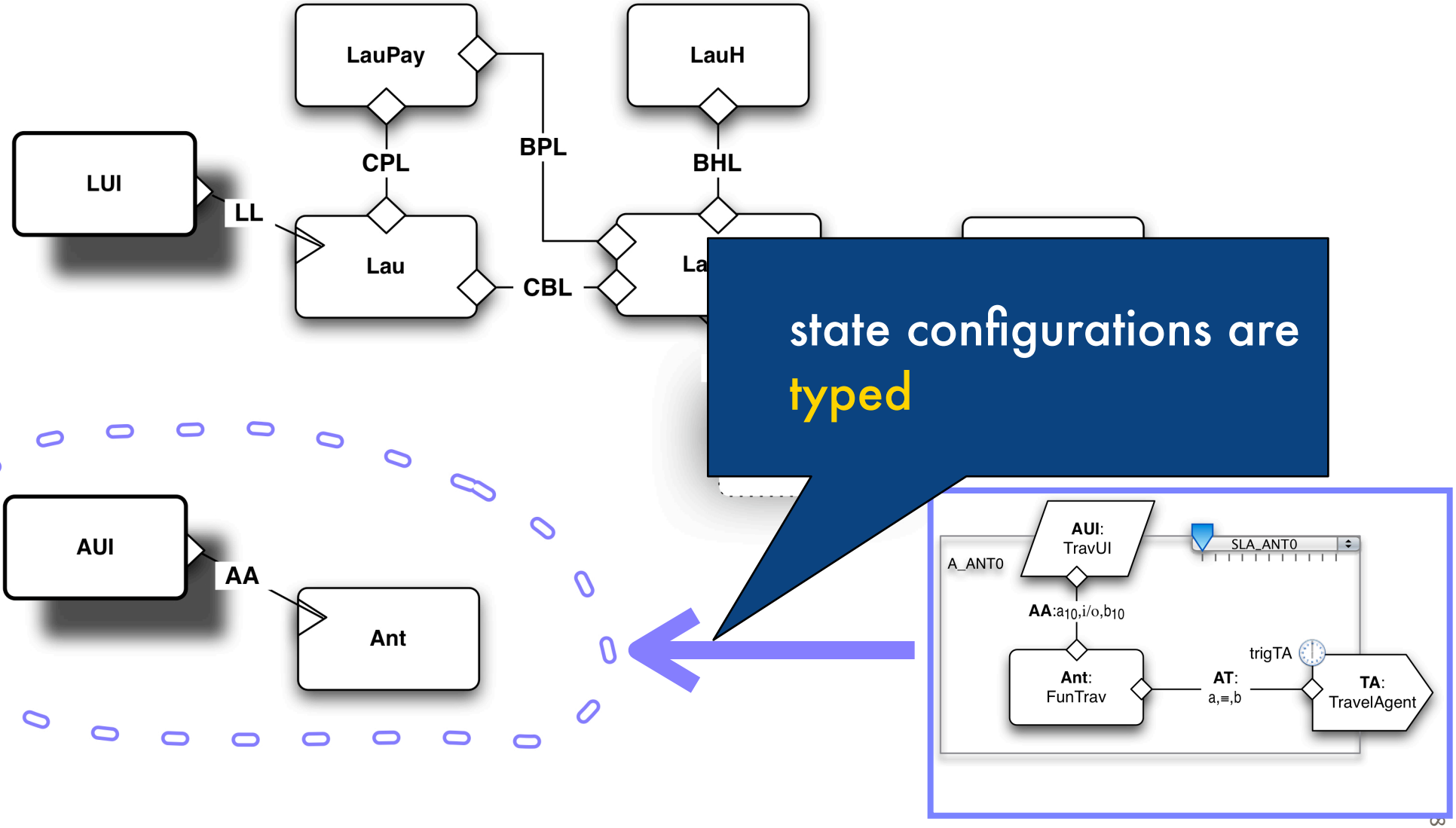
state configurations

A state configuration SF consists of:

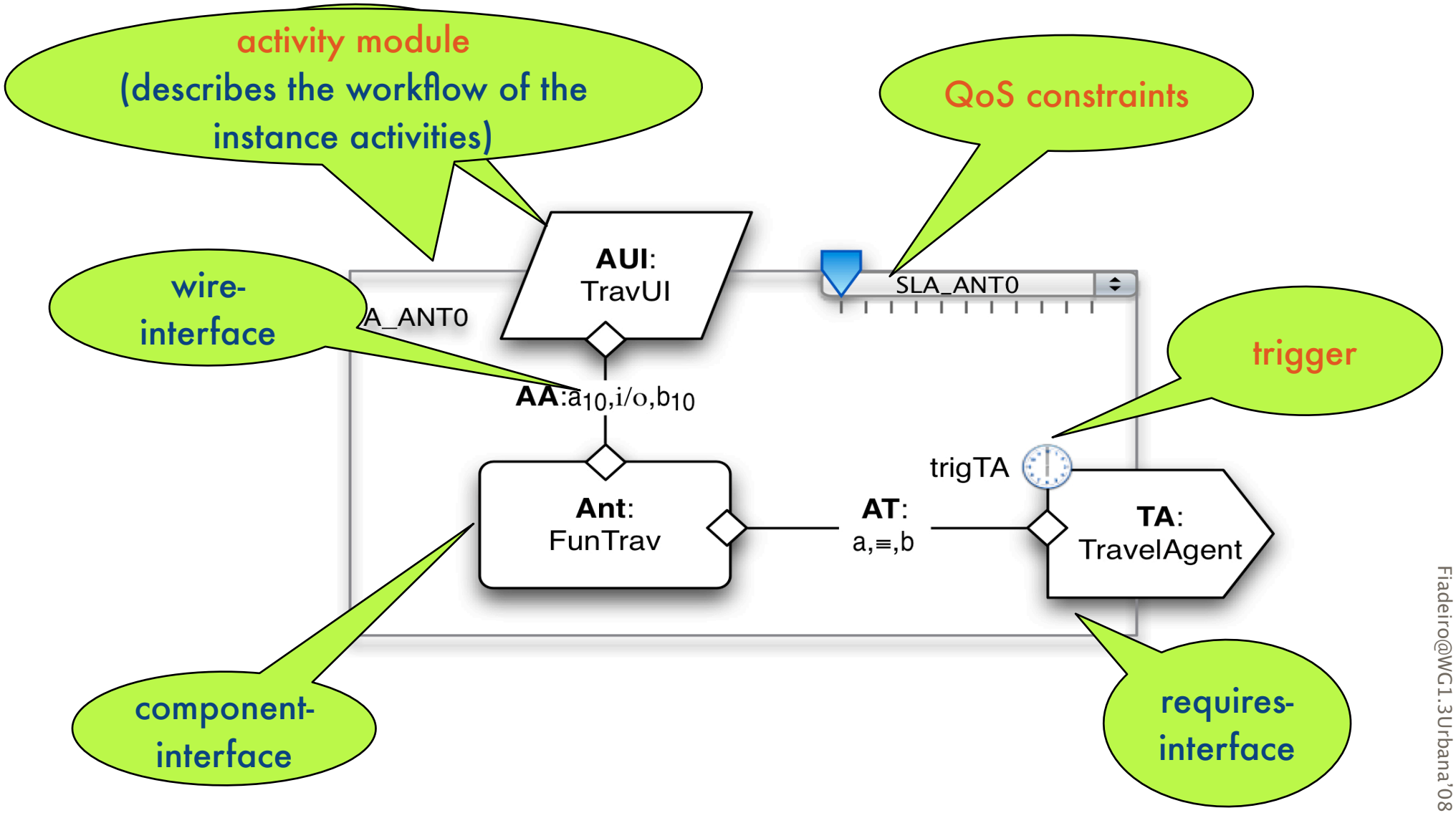
- A simple graph, i.e. a set nodes(SF) and a set edges(SF); nodes are components, and edges are wires.
- A labelling function that assigns a state to every node and edge.



social reflection



a business activity *type*



modules

A module M consists of:

- A labelled graph.
  - Some distinguished subsets of nodes(M):
    - requires(M) labelled with business protocols
    - uses(M) labelled with layer protocols
    - components(M) labelled with business roles
    - serves(M) (ø or 1) labelled with a layer protocol
    - provides(M) (ø or 1) labelled with a business protocol
  - Edges (wires) are labelled with connectors
- An internal configuration policy
- An external configuration policy

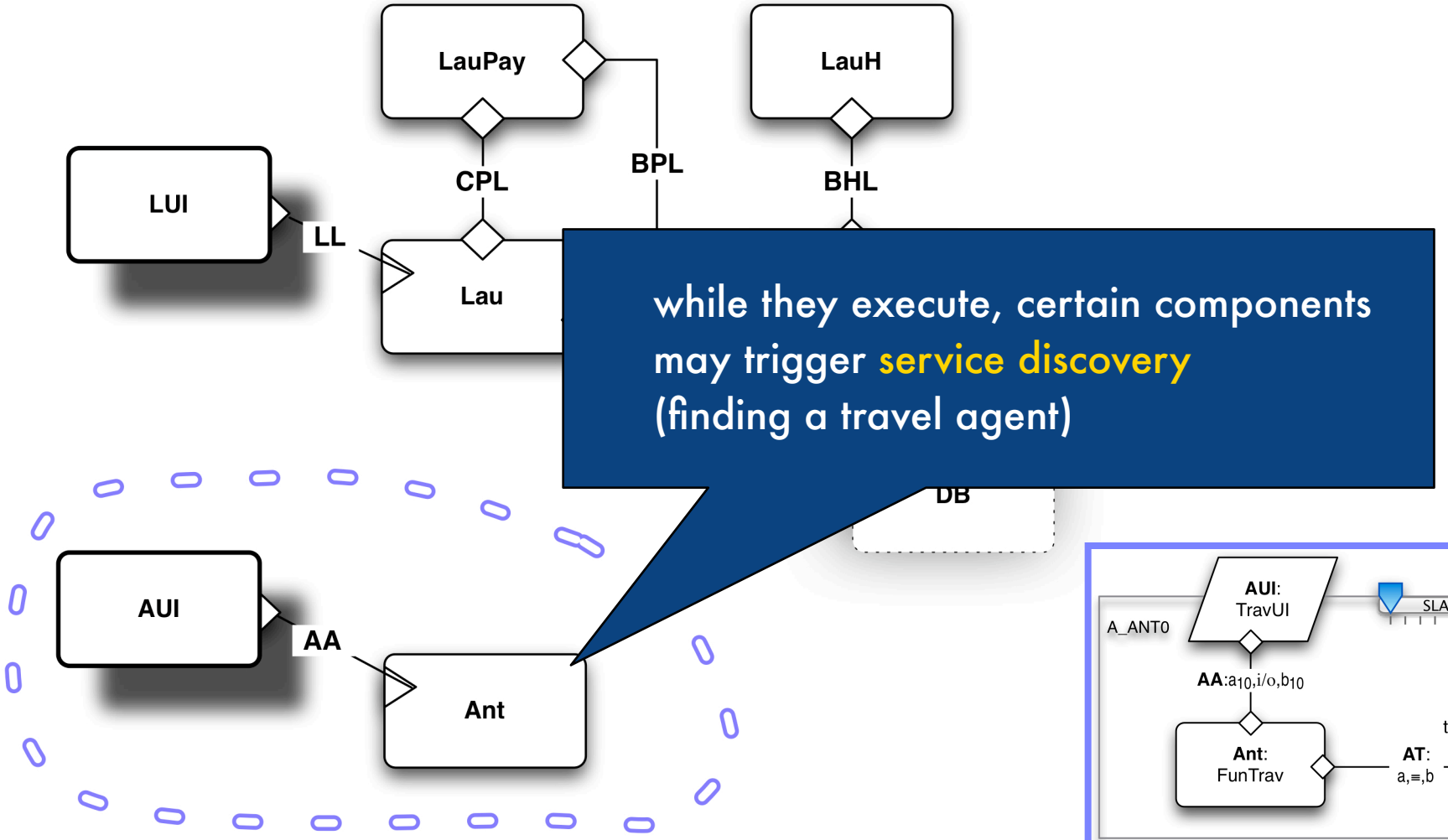
business configurations

A business configuration consists of:

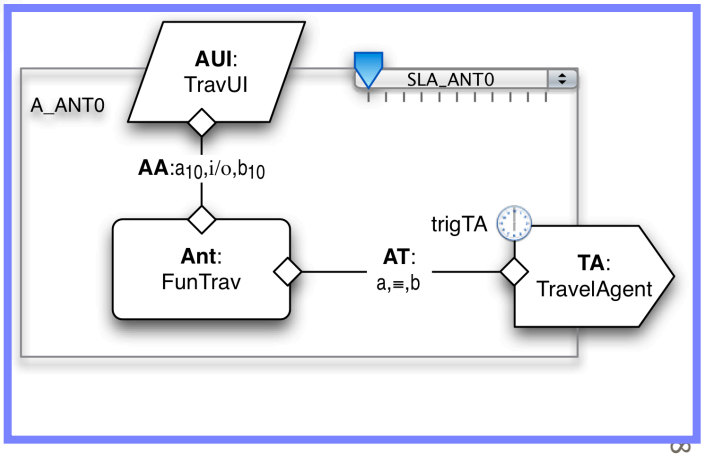
- A set of activity names (chosen from a domain).
- A state configuration SF.
- A mapping $\mathscr{B}$ that assigns a module $\mathscr{B}(a)$ to every activity a – the workflow performed by a in SF.
- For every activity a, a homomorphism $\mathscr{B}(a)$ of graphs between the body of $\mathscr{B}(a)$ and SF.

(This homomorphism makes configurations reflective.)

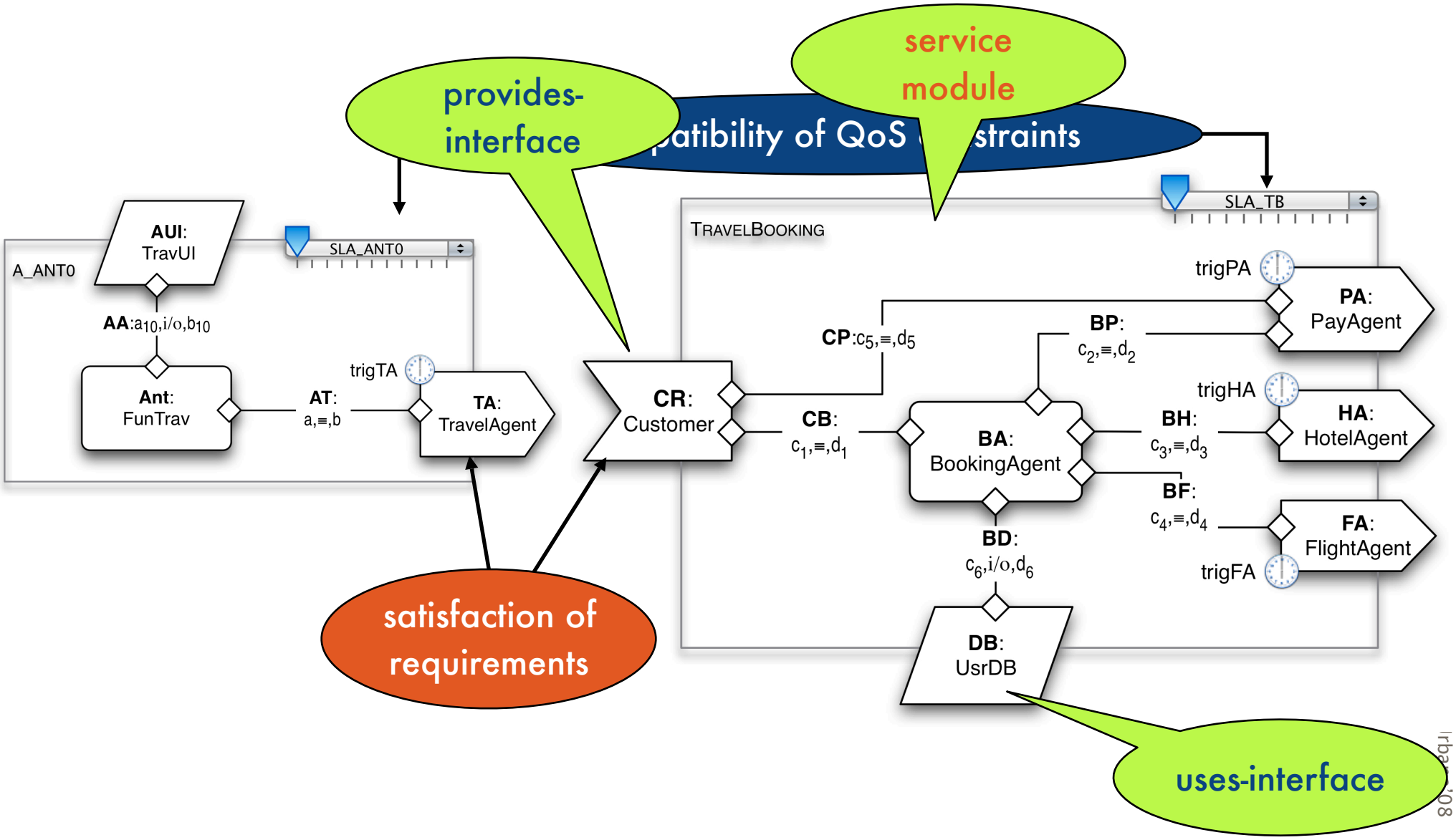
a business activity instance



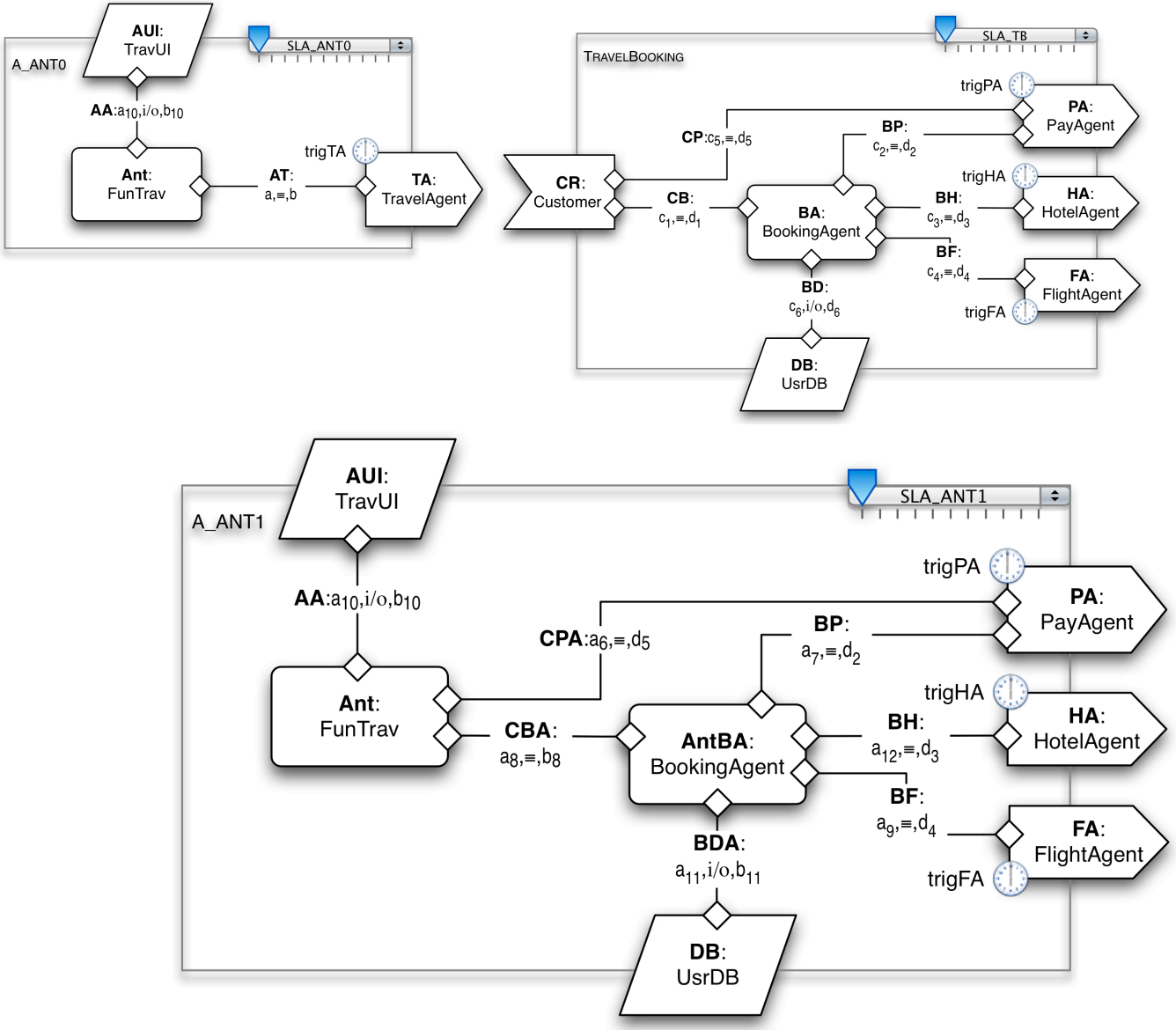
while they execute, certain components may trigger service discovery (finding a travel agent)



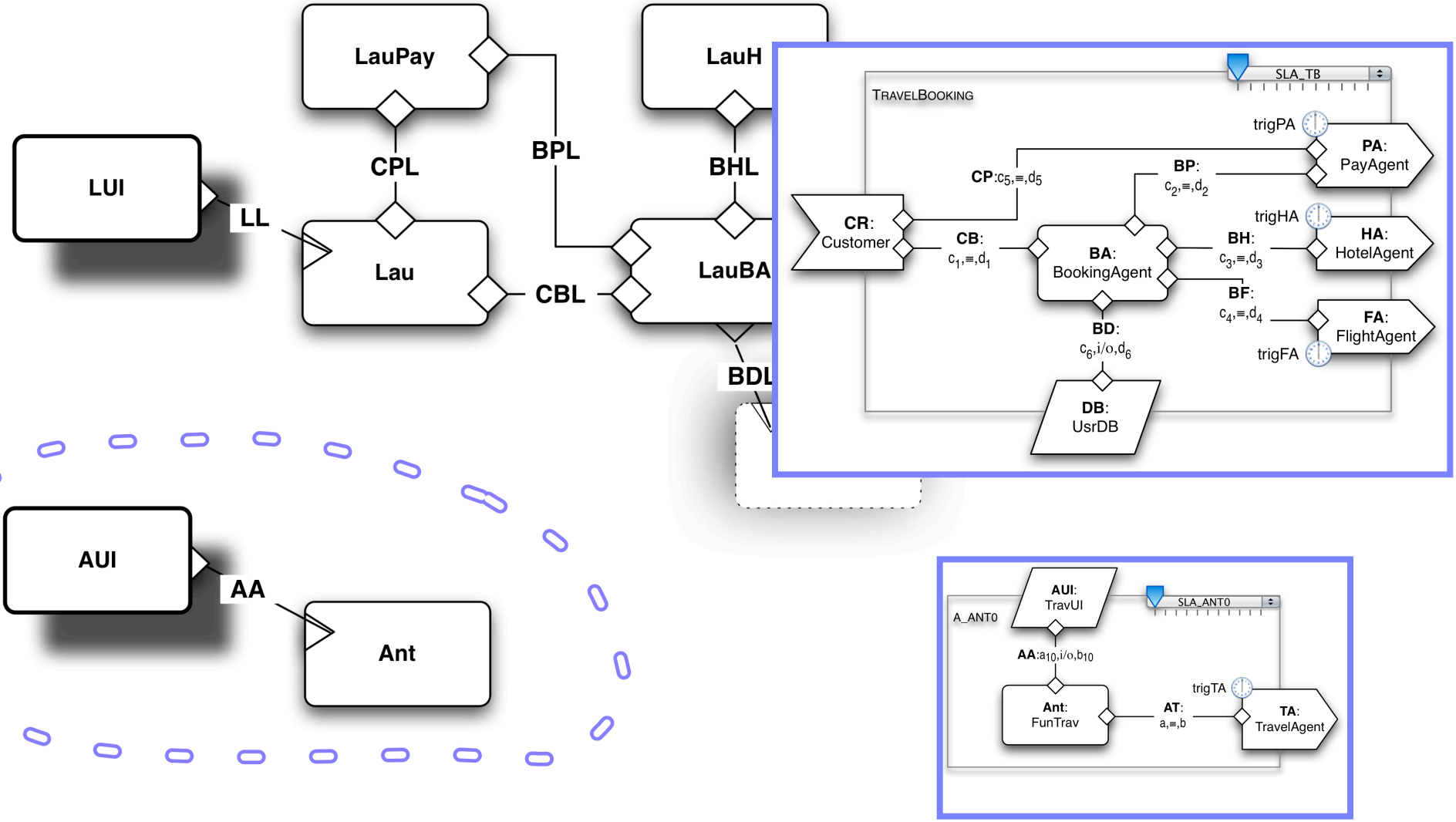
discovery and selection



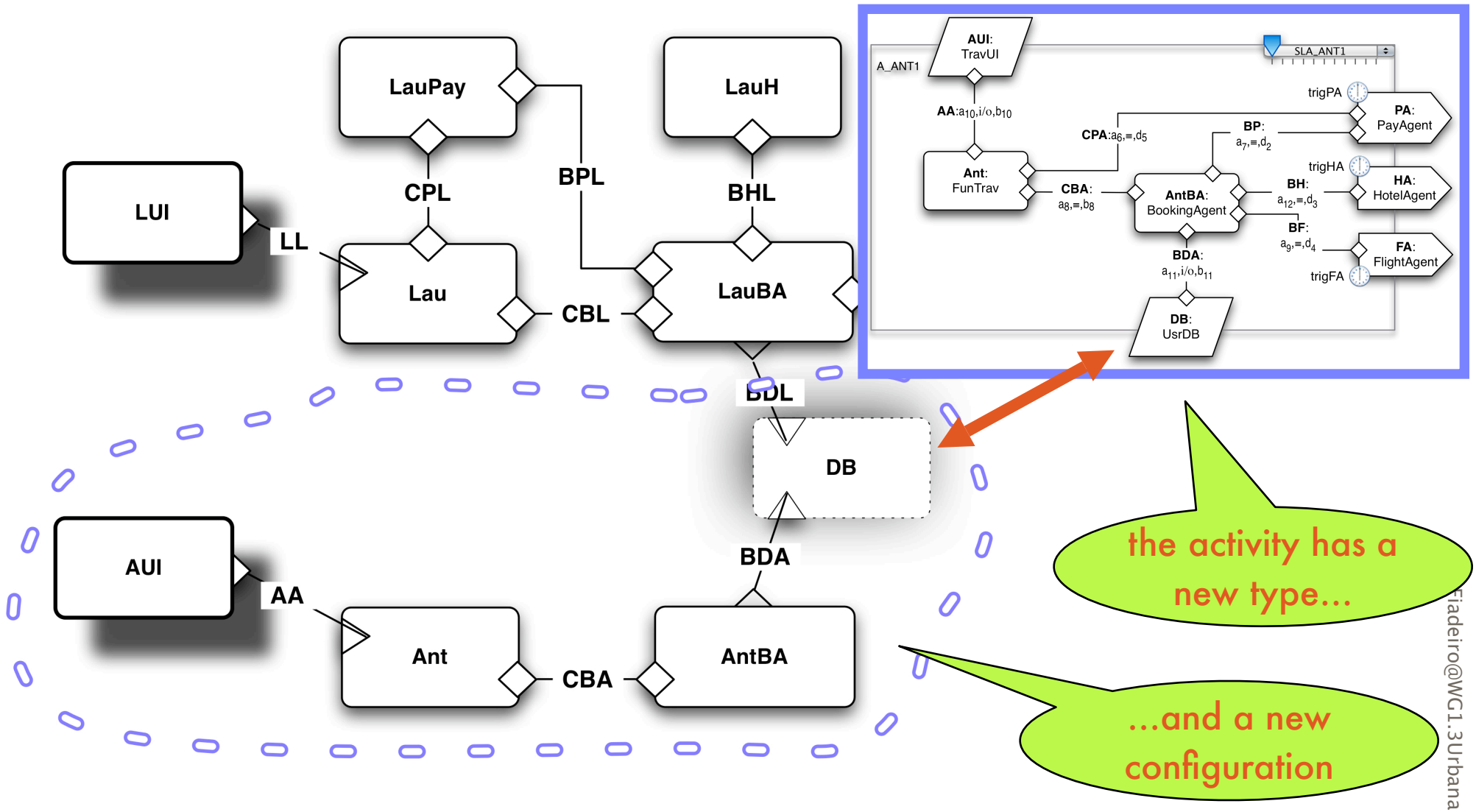
binding/composition



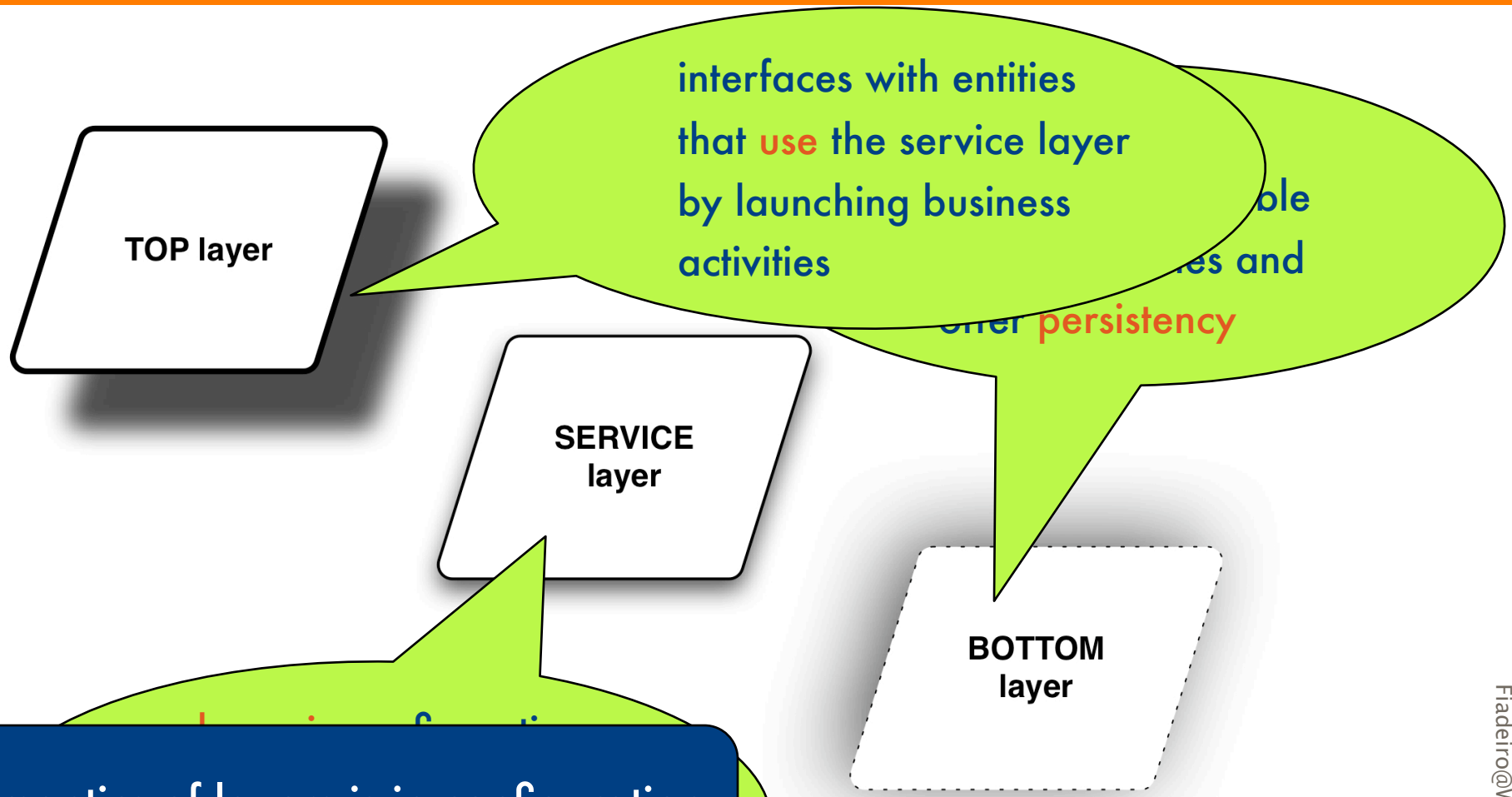
instantiation



instantiation

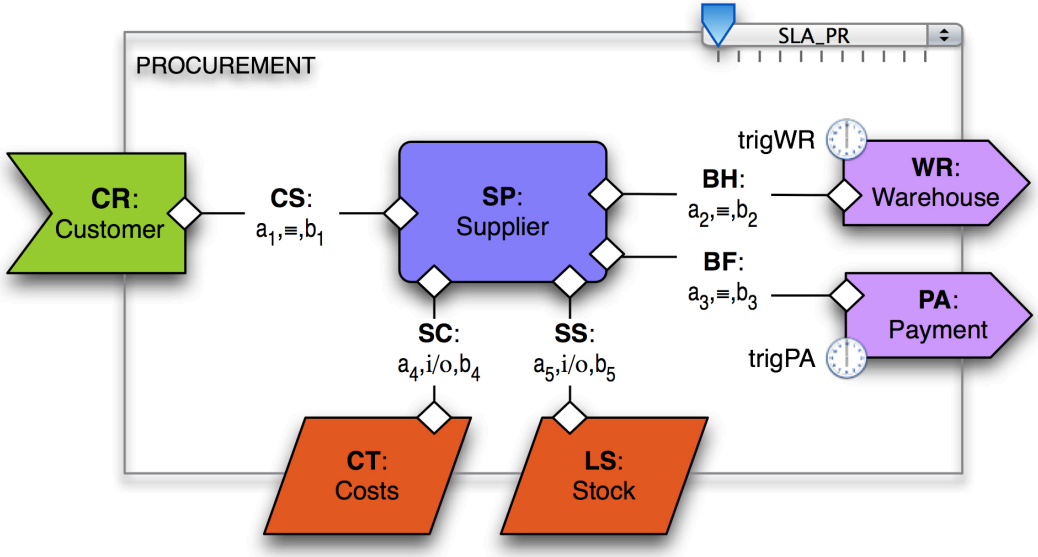


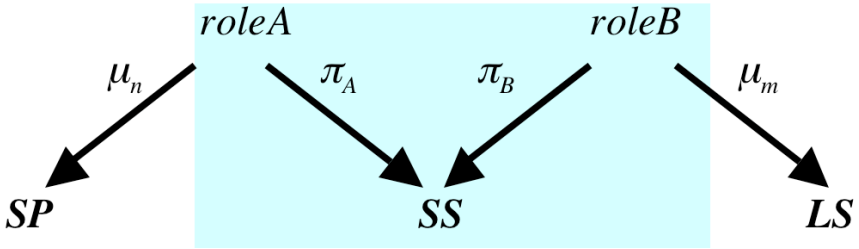
three layers

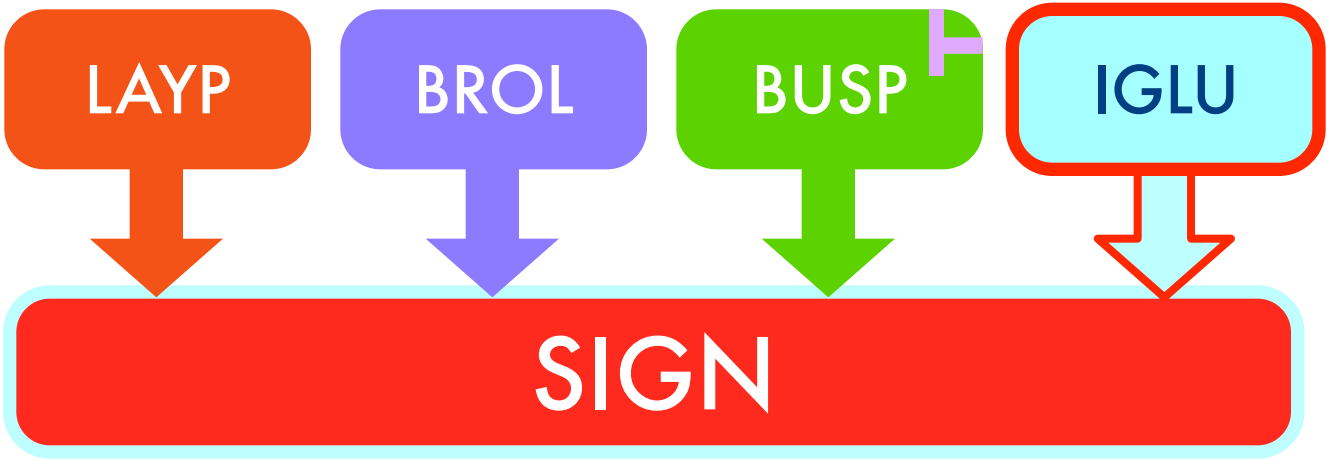


The semantics of layers is in configuration management. The SRML architectural was formalised in ADR by Alberto Lafuente.

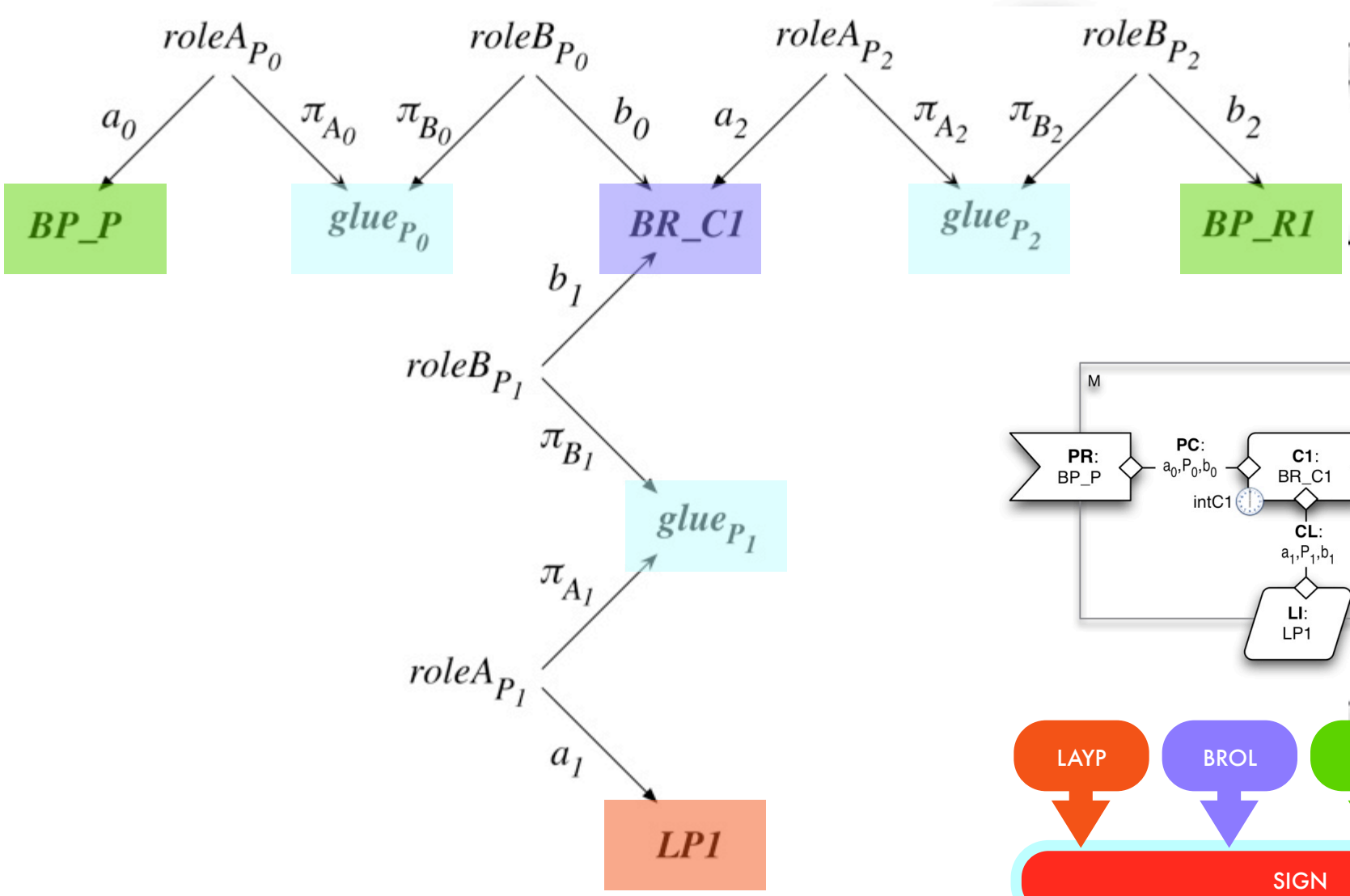
The formal domains



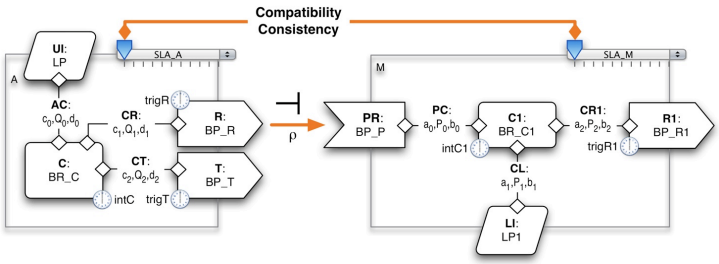


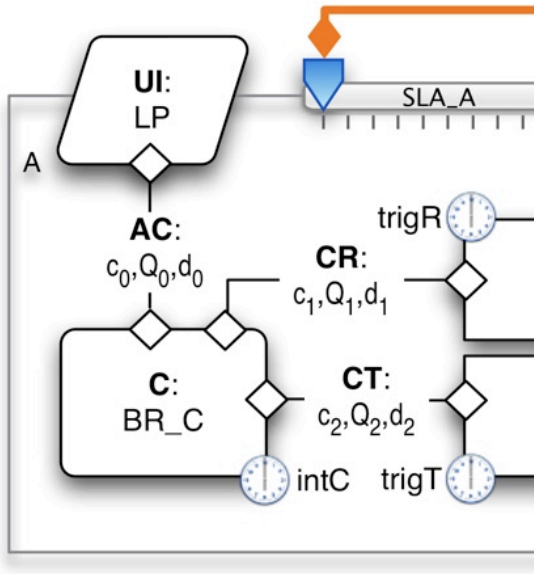


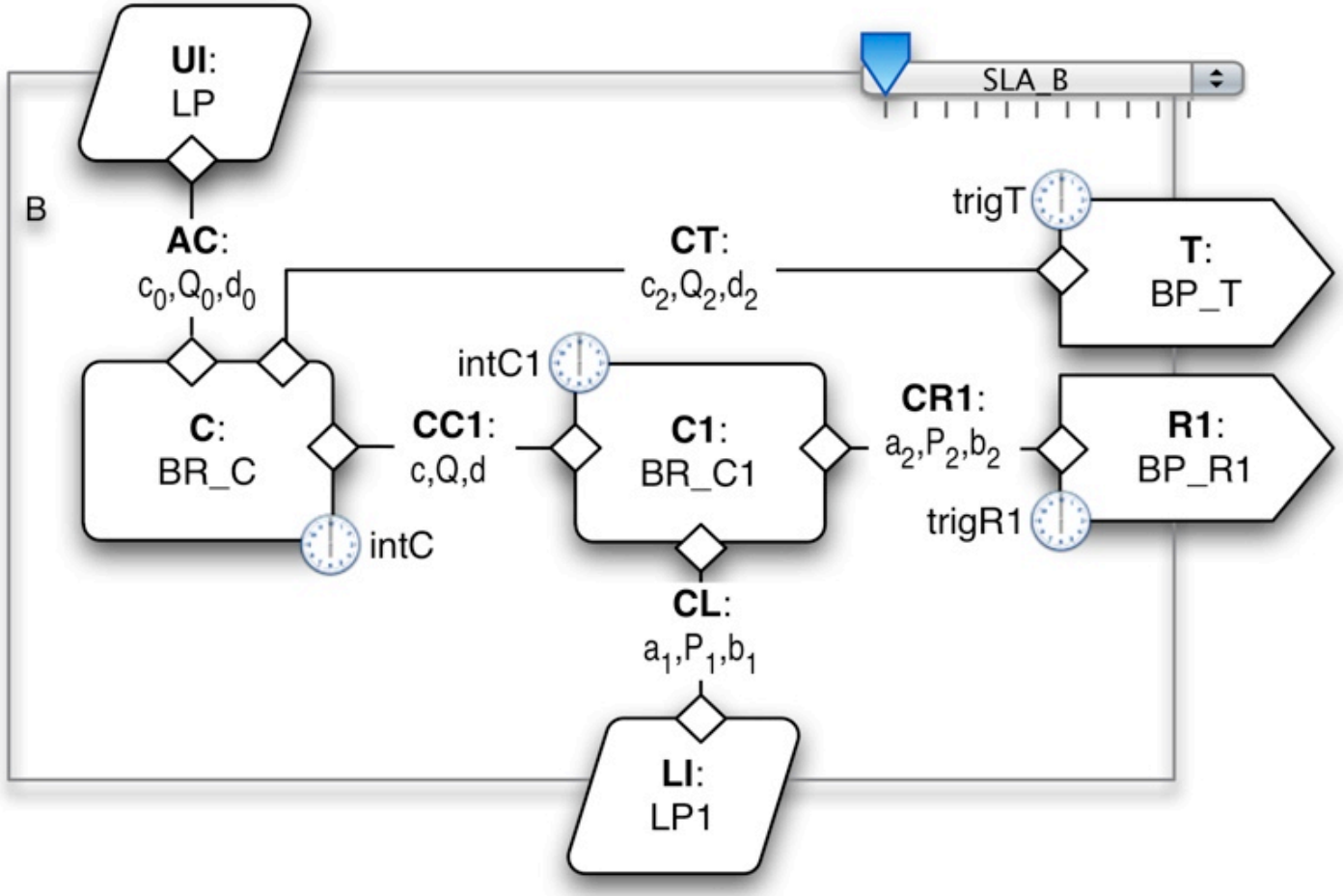
Modules as labelled graphs



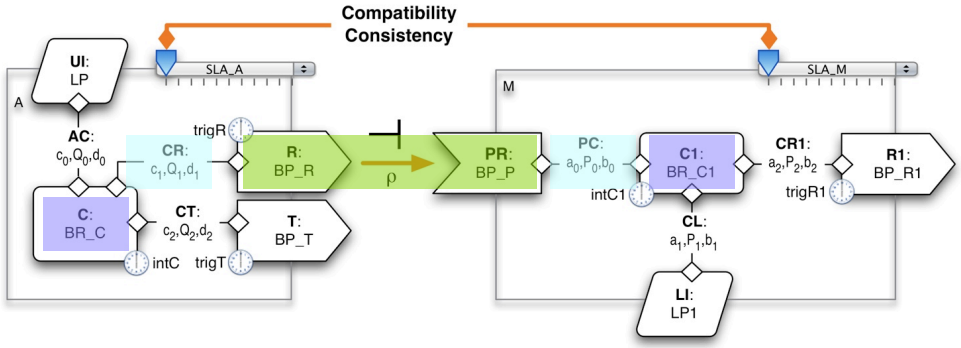
Algebraic Semantics of Composition

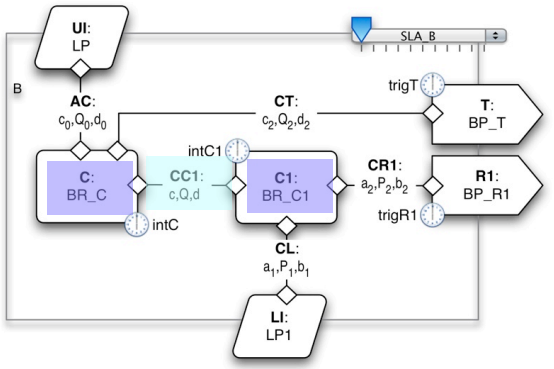


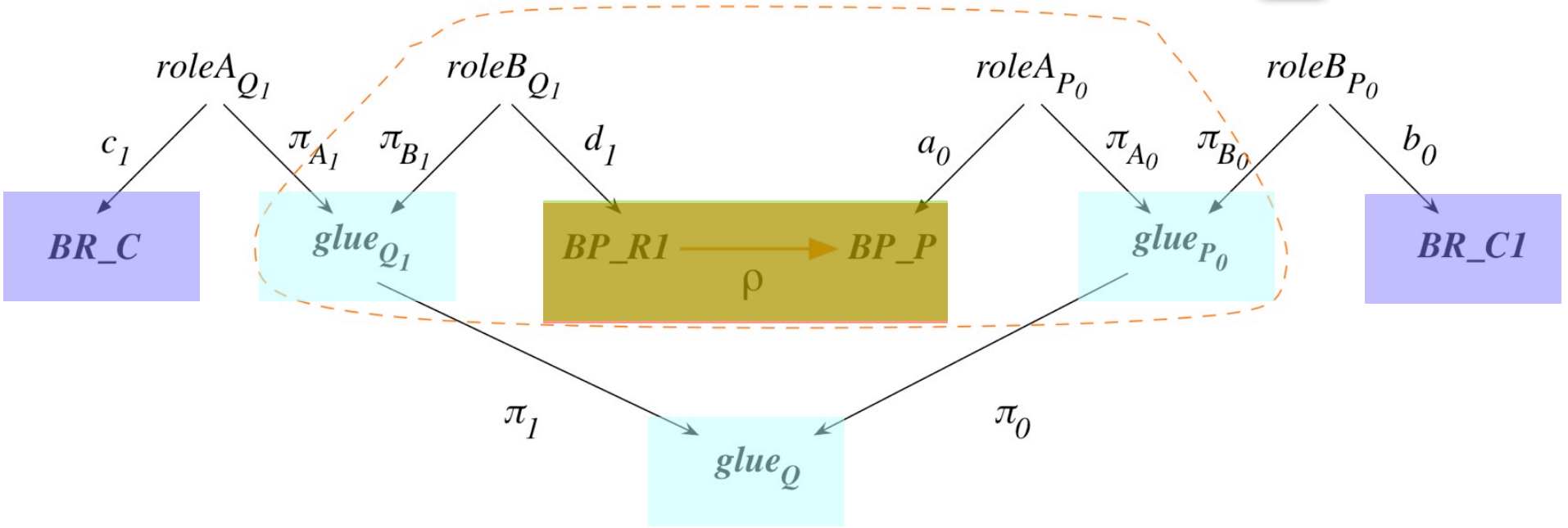




Composition



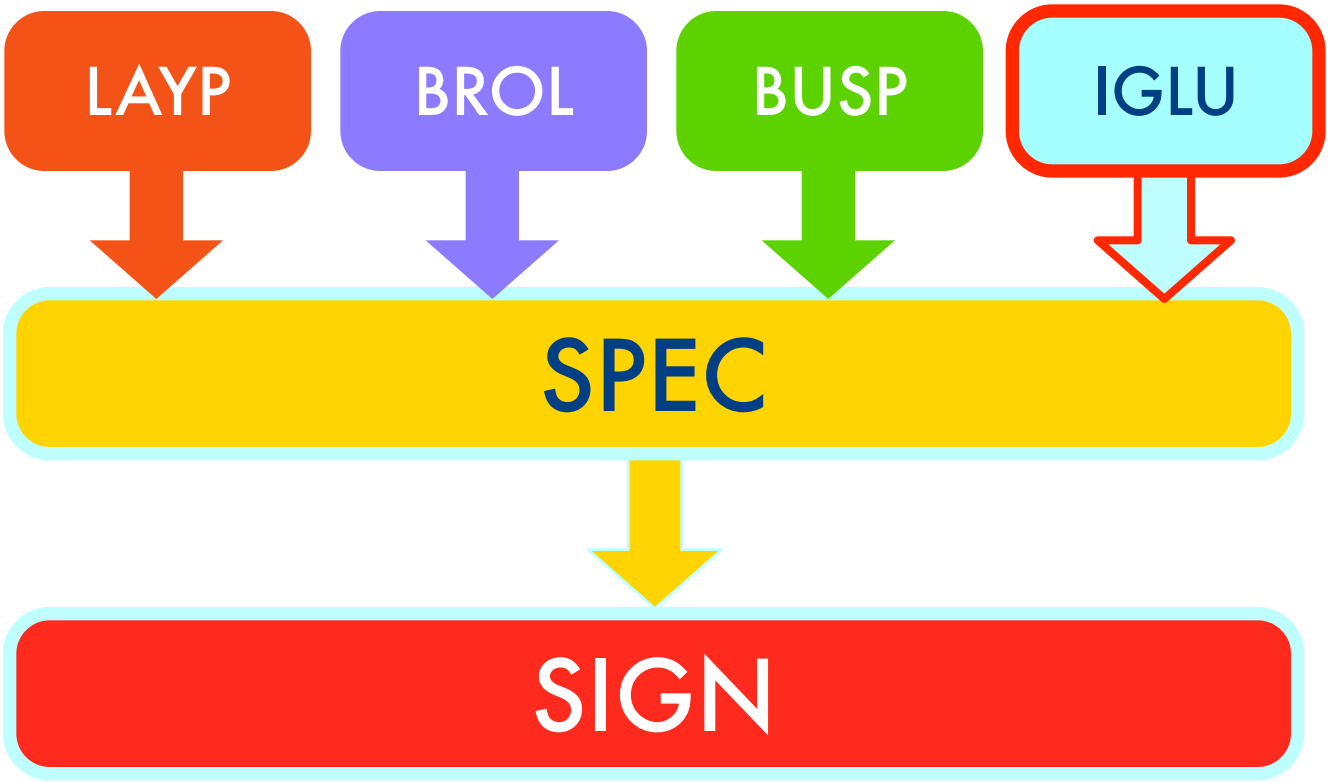




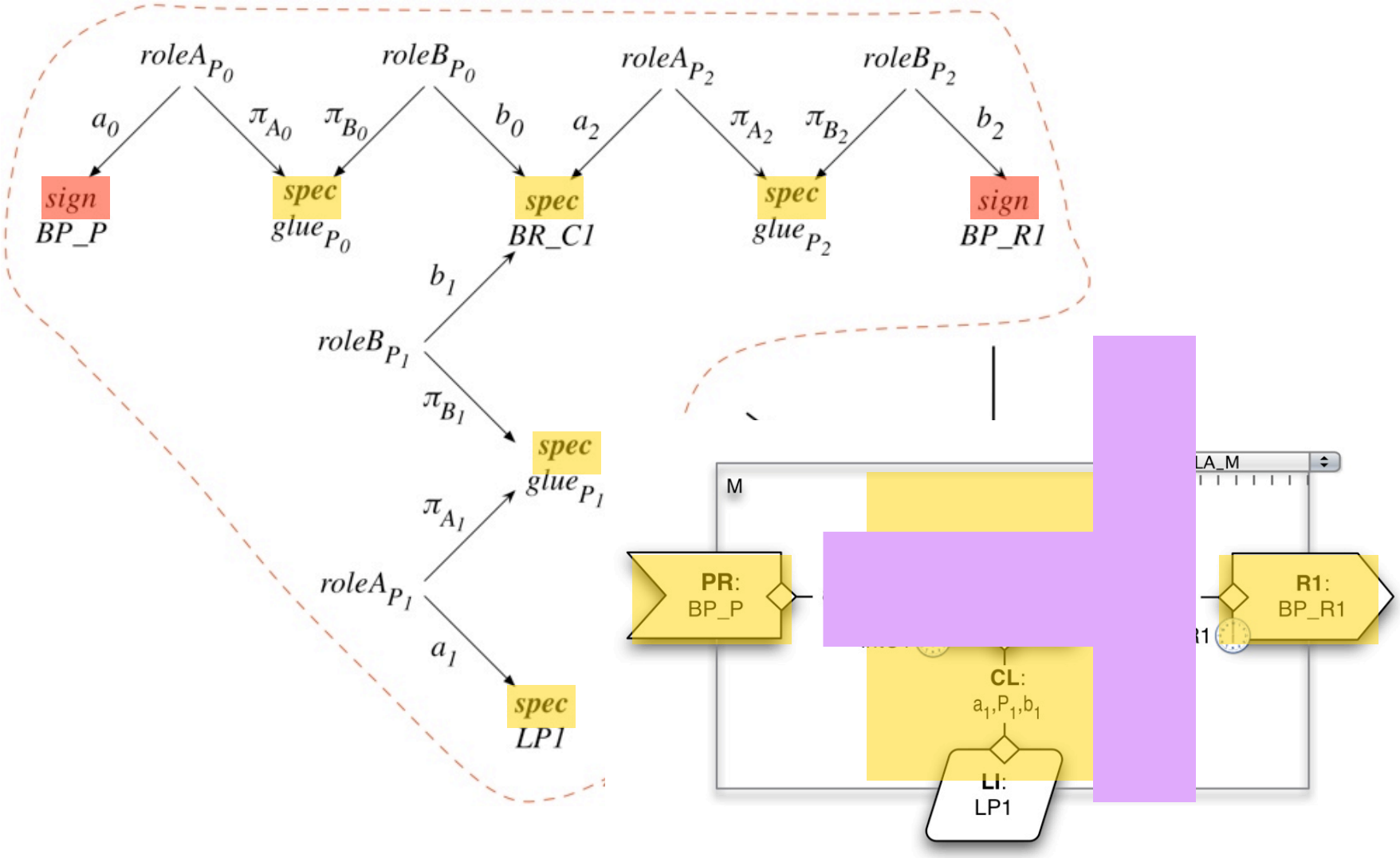
structured co-spans over SIGN

Correctness by entailment

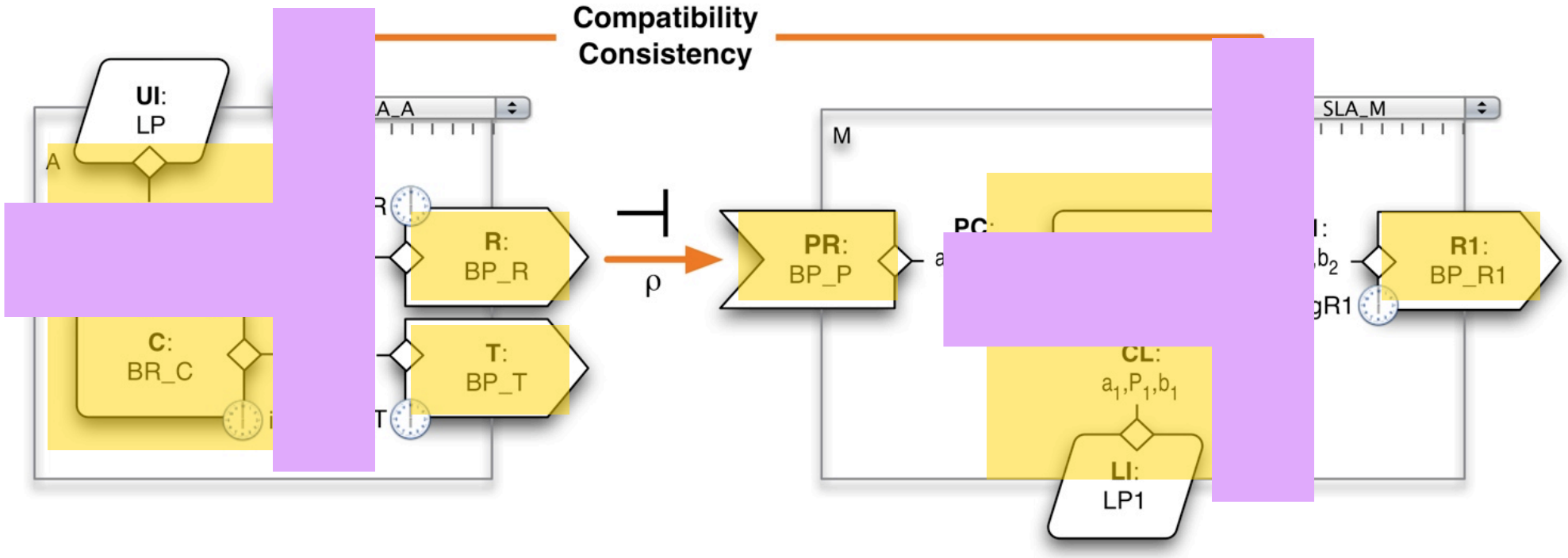
An entailment system
<**SIGN**,**gram**:SIGN→**SET**, ⊢>



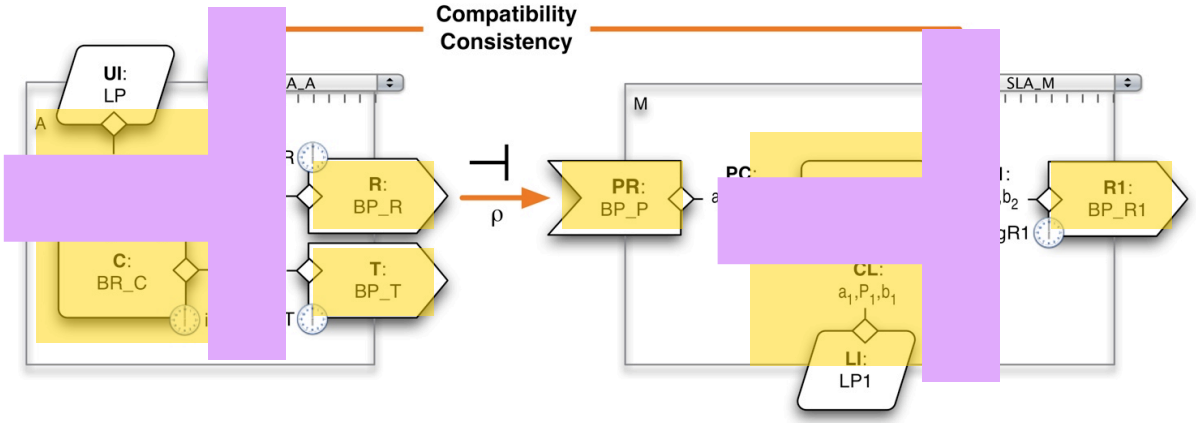
Correctness by entailment

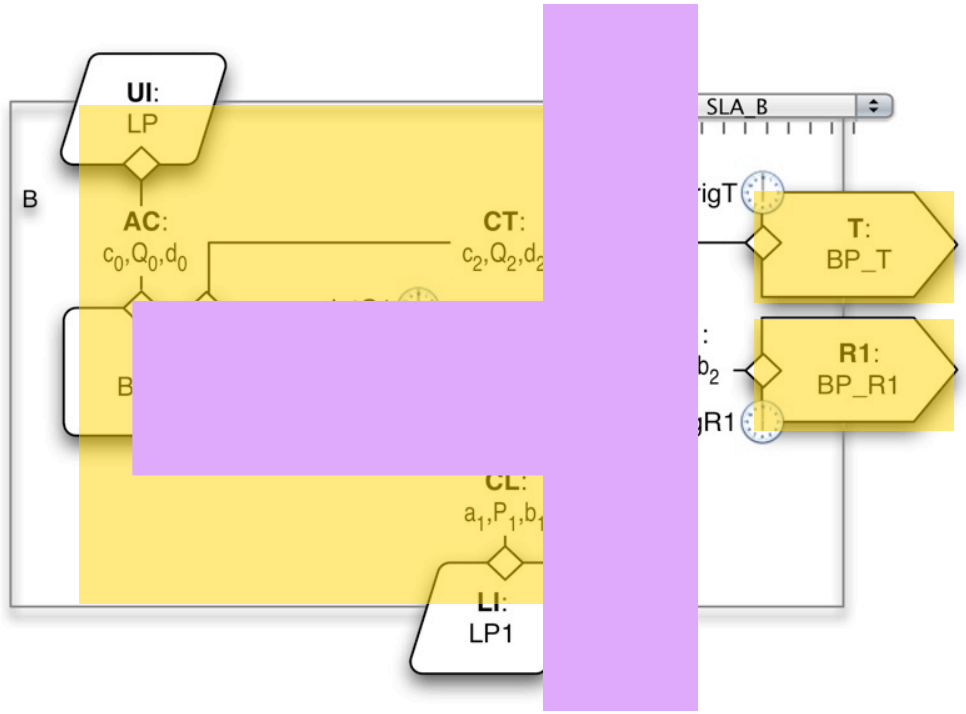


Composition



Composition by resolution





Questions

- Model-theoretic correctness
- Model-checking correctness
- Distributed heterogeneous institutional framework

What else?

- Already available:
  - Semantics of SLA over c-semirings
  - Mapping of BPEL to SRML
- On-going research:
  - Logic for specification and verification
  - Analysis through model checking
  - Operational semantics over service calculi
  - UML-notations
  - Relationship with SCA