Electronic Institutions

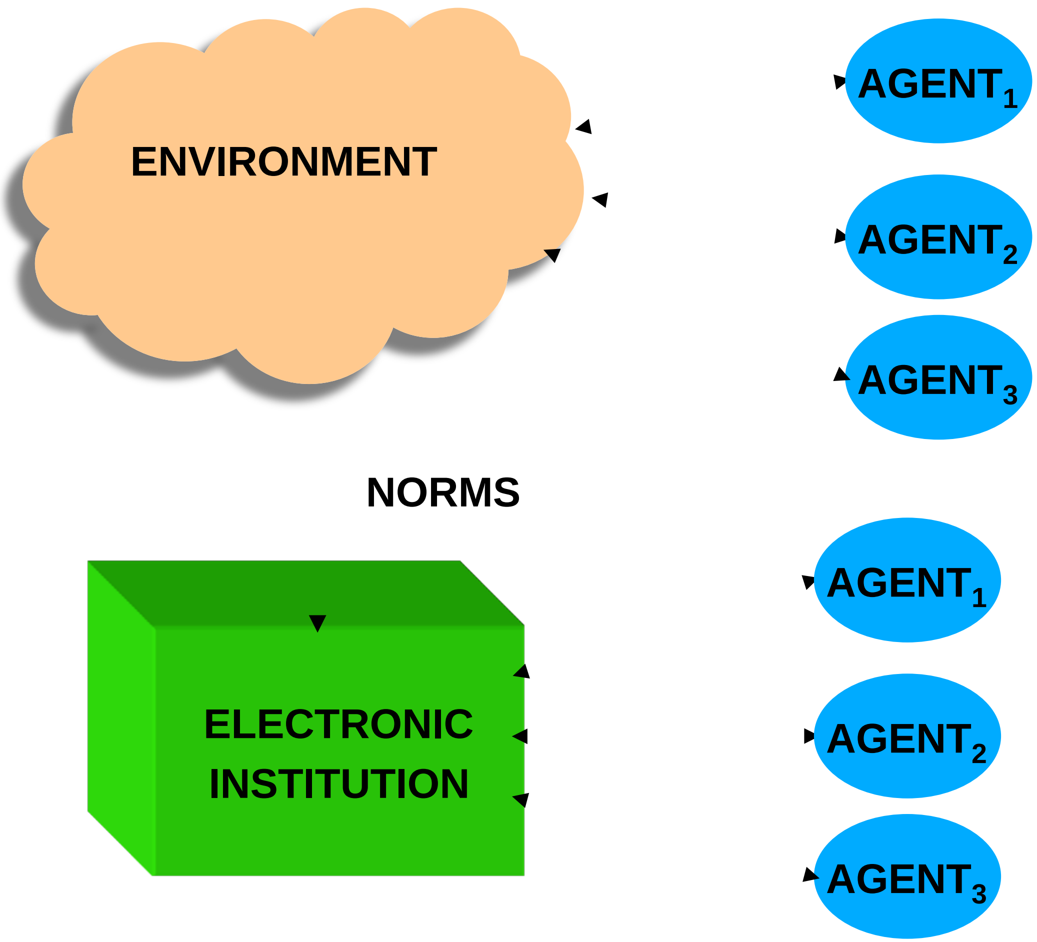
Marc Esteva

Artificial Intelligence Research Institute (IIIA), CSIC, Spain

I. Introduction
II. Specifying electronic institutions
III. Running electronic institutions
IV. Electronic Institutions Development Environment
V. Conclusions

- *Research issue*: design and development of **methodologies** and **software tools** to support the **design, verification, deployment, and analysis of open multi-agent systems.**
- **Open multi-agent systems** are populated by *heterogeneous*, *self-interested* agents, developed by different people, using different languages and architectures. Participants *change* over time and are unknown in advance.
- With the expansion of the Internet open multi agent systems represent the most important area of application of multi agent systems.

- Institutions have proved to successfully regulate human societies for a long time:
  - created to achieve particular goals while complying norms.
  - responsible for defining the rules of the game (norms), to enforce them and assess penalties in case of violation.
- Examples: auction houses, parliaments, stock exchange markets,….

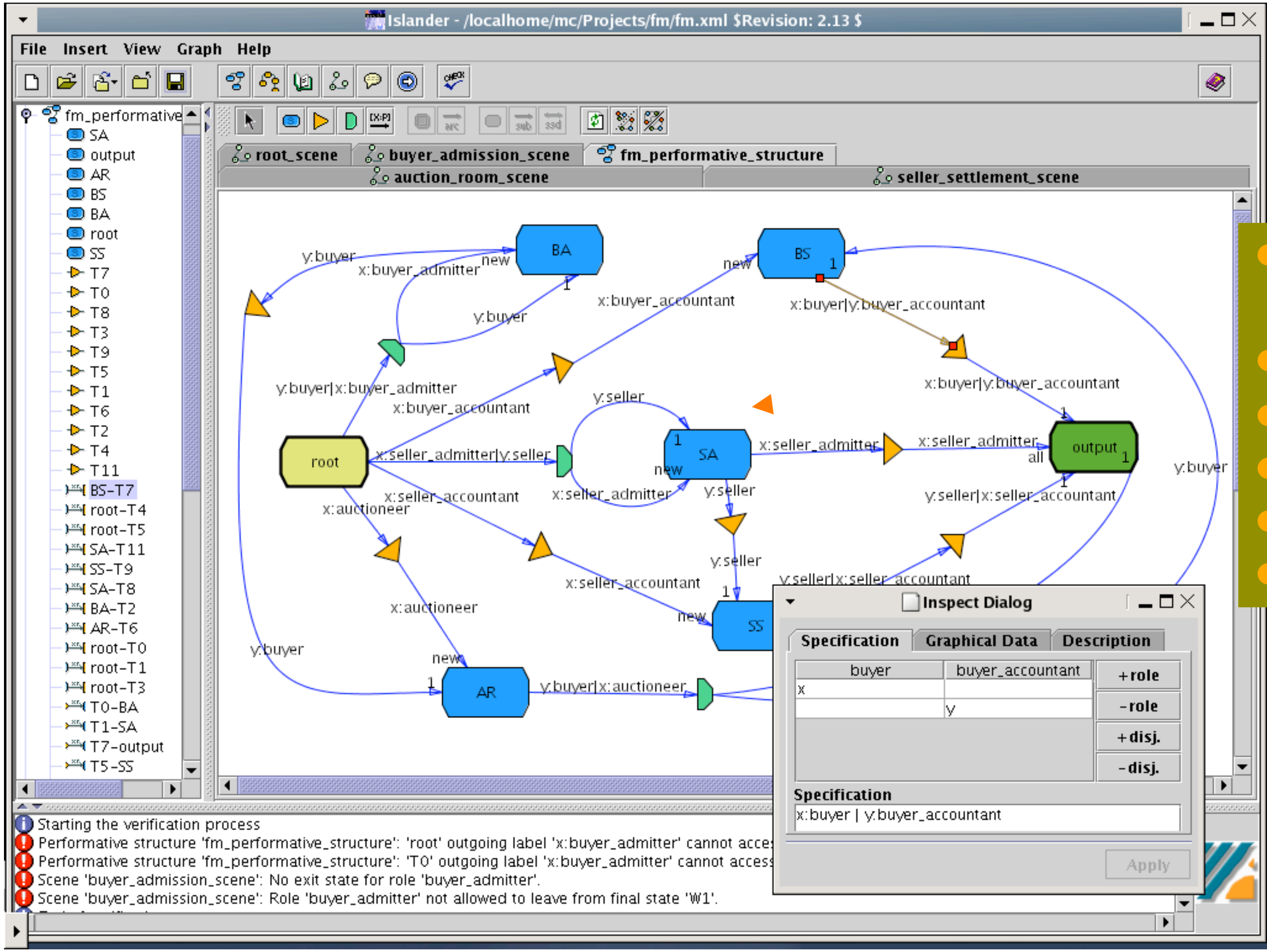


Institutions in the sense proposed by North "… set of artificial constraints that articulate agent interactions".

- **Electronic institutions development** can be divided into two basic steps:
  - **Formal specification** of institutional rules.
  - **Execution** via an infrastructure that **mediates** agents' interactions while **enforcing** the institutional rules.
- The **formal specification** focuses on **macro-level** (rules) aspects of agents, not in their micro-level (players) aspects.
- The **infrastructure** is required to be of **general purpose** (can *interpret* any formal specification).

I. Introduction
II. Specifying electronic institutions
III. Running electronic institutions
IV. Electronic Institutions Development Environment
V. Conclusions

Electronic Institution Specification with ISLANDER



- Common Ontology and language
- Agent Roles
- Network of protocols
- Multi-agent protocols
- Norms
- Information Models

Dialogical Framework Components

- Common ontology
- Valid communication language expressions
  - List of illocutionary particles
  - Content language
- Roles that agents can play

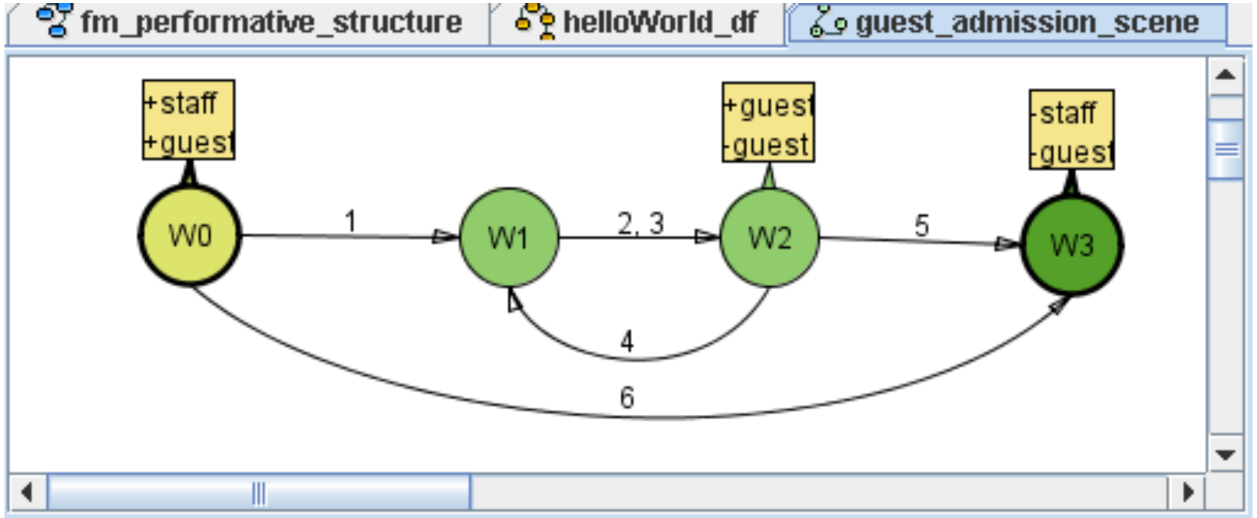
- CL expressions are formulae of the form $(i\ (\alpha_i\ r_i)\ \beta\ \gamma\ \tau)$ where:
  - *i* is an illocutionary particle (e.g. request, inform);
  - α_i can be either an agent variable or an agent identifier;
  - r_i can be either a role variable or a role identifier;
  - β represents the addressee(s) of the message and can be:
    - $(\alpha_k\ r_k)$ the message is addressed to a single agent.
    - r_k the message is addressed to all the agents playing role r_k.
    - "all" the message is addressed to all the agents in the scene.
  - γ is an expression in the content language.
  - τ can be either a time variable or a time-stamp

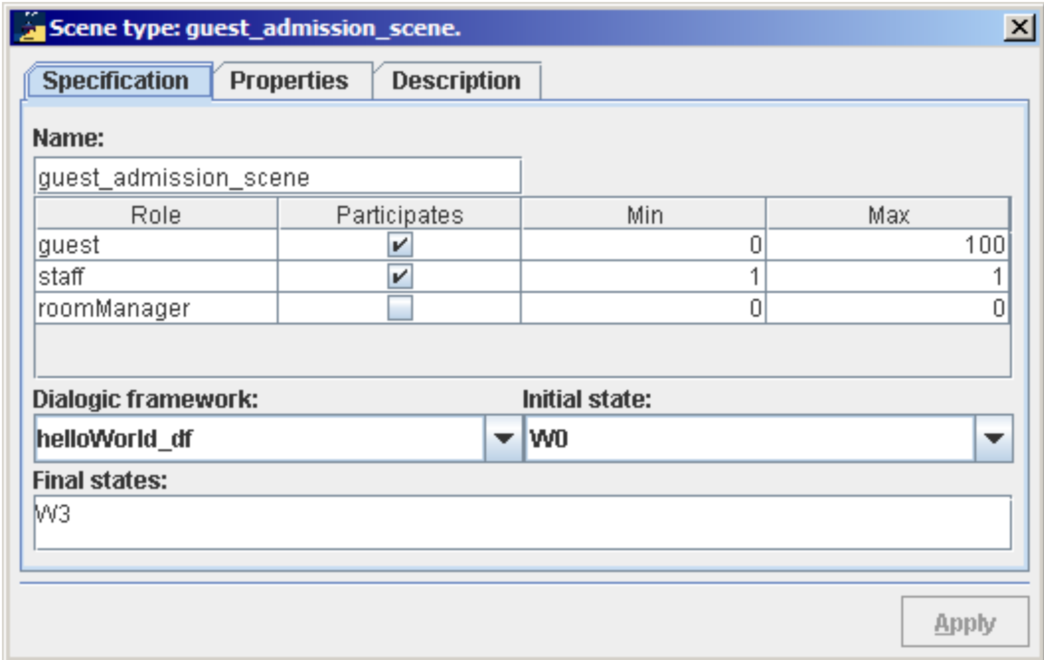
- *(request (?x guest) (!y staff) login(?user,?password)*

- Each role defines a pattern of behaviour within the institution (actions associated to roles).
- Agents can play multiple roles at the same time
- Agents can change their roles.
- Two types of roles:
  - Internal: played by the staff agents to which the institution delegates its services and tasks.
  - External: played by external agents.
- Role relationships:
  - Static incompatibility (ssd)
  - Dynamic incompatibility (dsd)
  - Hierarchy (sub)
- Information model per role: a set of attributes that define the information that the institution keeps per each role.

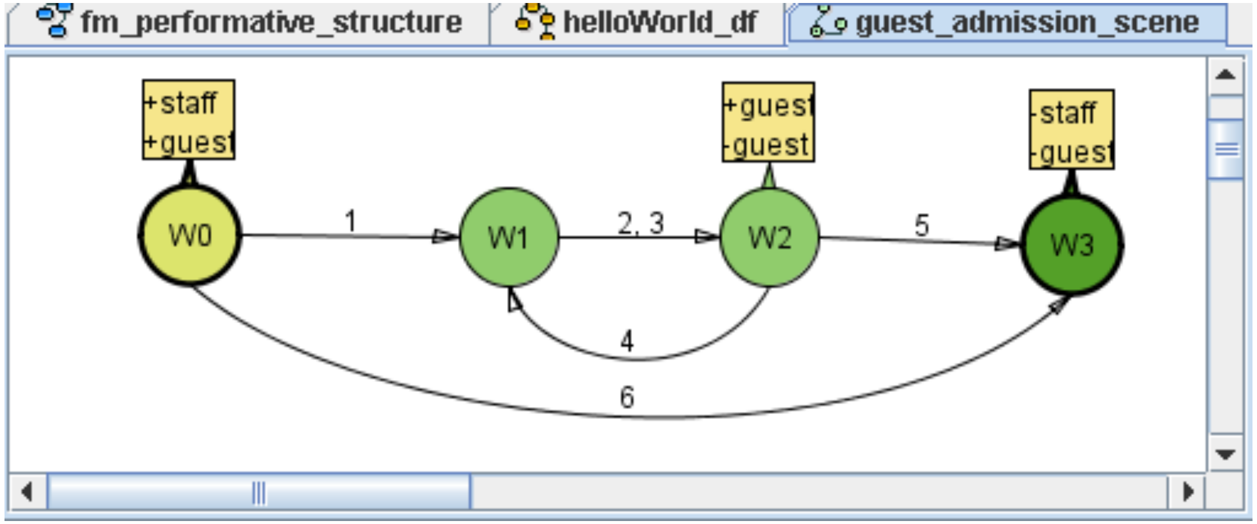
- Specification level
  - A scene is a pattern of multi-agent interaction.
  - Scene protocol specified by a finite state automata where the nodes represent the different states and oriented arcs are labelled with *illocution schemes* or *timeouts*.
- Execution level
  - Agents may join or leave scenes.
  - Each scene keeps the *context* of its multi-agent interaction.
  - A scene can be multiply executed and played by different groups of agents.

Guest admission scene





Guest admission scene. Illocutions



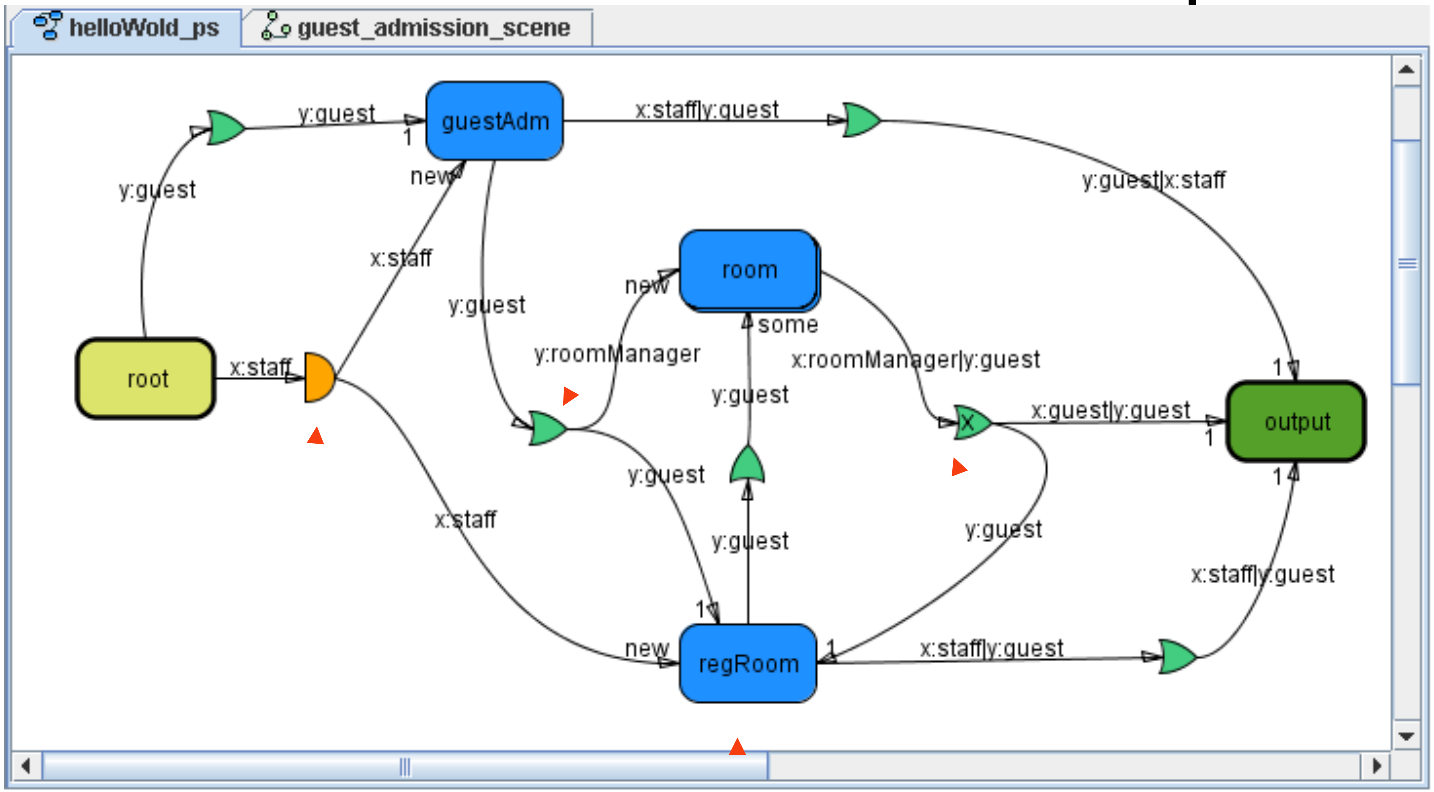
1. (request (?x guest) (?y staff) login(?user ?email)))
2. (inform (!y staff) (!x guest) accept()))
3. (failure (!y staff) (!x guest) deny(?code)))
4. (request (?x guest) (!y staff) login(?user ?email)))
5. (inform (!y staff) (all guest) close()))
6. (inform (?y staff) (all guest) close()))

Scene Constraints

- Constraints capture how past actions in a scene affect its future evolution:
  - restricting the valid values for a variable
  - restricting the paths that a conversation can follow
- Examples:
  - A buyer can only submit a single bid at auction time.
  - A buyer must submit a bid greater than the last one.
  - An auctioneer can not declare a winner if two buyers have submitted a bid at the higher value.
  - An agent can not repeat an offer during a negotiation process.

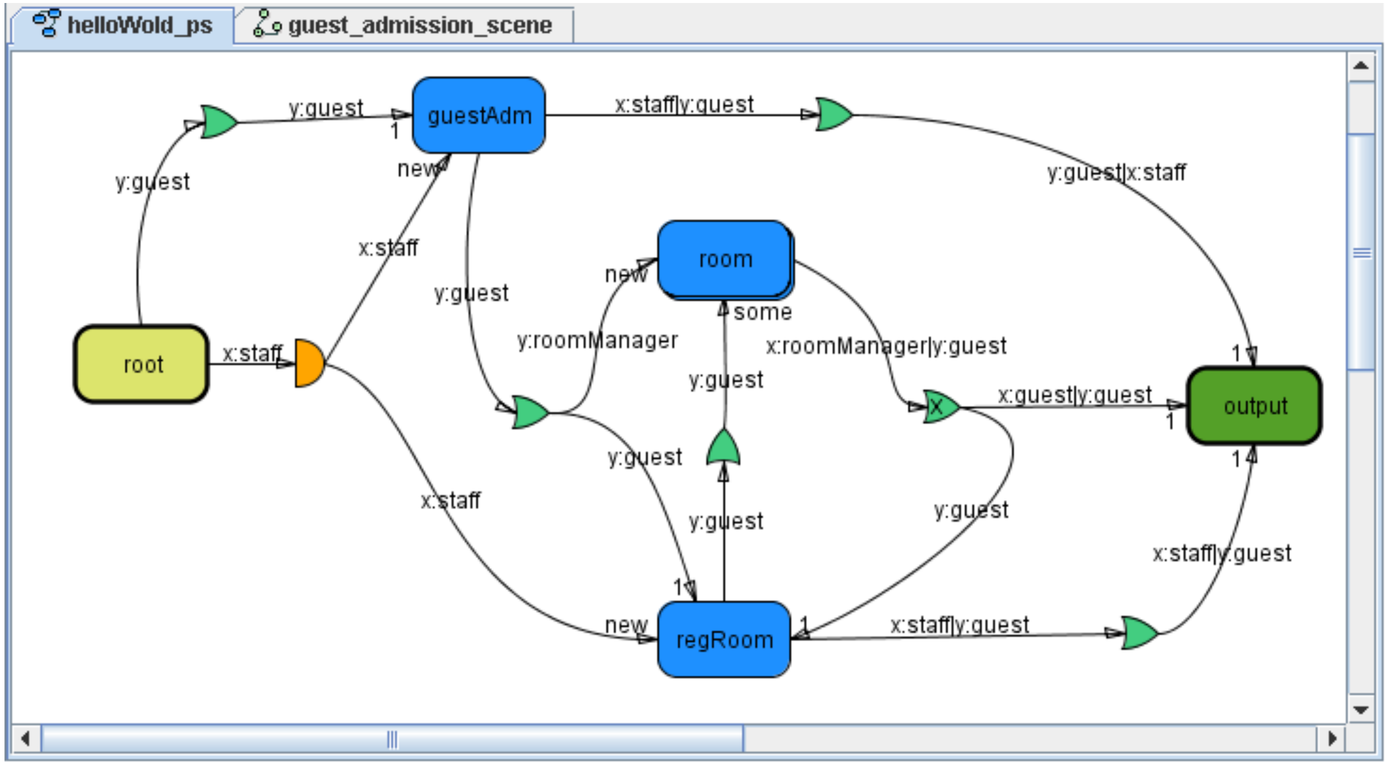
- Complex activities can be specified by establishing relationships among scenes that define:
  - causal dependency (e.g. buyers must go through the admission scene before joining an auction room)
  - synchronisation points (e.g. synchronise a buyer and a seller before starting a negotiation scene)
  - parallelisation mechanisms (e.g. buyers can go to multiple auction rooms)
  - choice points (e.g. a buyer leaving an admission scene can choose which auction scene to join)
  - the role flow policy

Or transition: **choice point**



Activity XOr transition: **exclusive choice point**

And transition: **synchronisation and parallelisation point**



Arcs connecting transitions to scenes determine whether agents join *one*, *some* or *all* current executions of the target scene(s) or whether *new* executions are started.

- Norms define the consequences of agents actions within the institution.
- Such consequences are captured as obligations.
  - Obl(x,ι,s): meaning that agent x is obliged to utter ι in scene s.
- Actions expressed as pairs of scene and illocution schema.

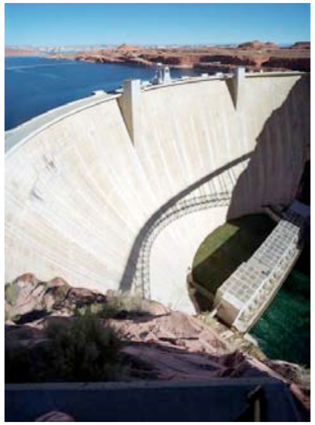
Buyers' Payment

$$done(inform(?x : a, all : b, sold(?good\_id, ?buyer\_id, ?price)), auction\_room) \Rightarrow$$

$$obliged(!x, pay(!buyer\_id : b, !y : bac, sale(!good\_id, !buyer\_id, !price)), buyer\_settlements)$$

The electricity market

Producers



Network

Consumers

The scenario:

- Periodic auctions of transmission rights in form of **tickets** valid for the injection or extraction of energy over half an hour periods.
- Double auction.
- Offer is greater than the demand.

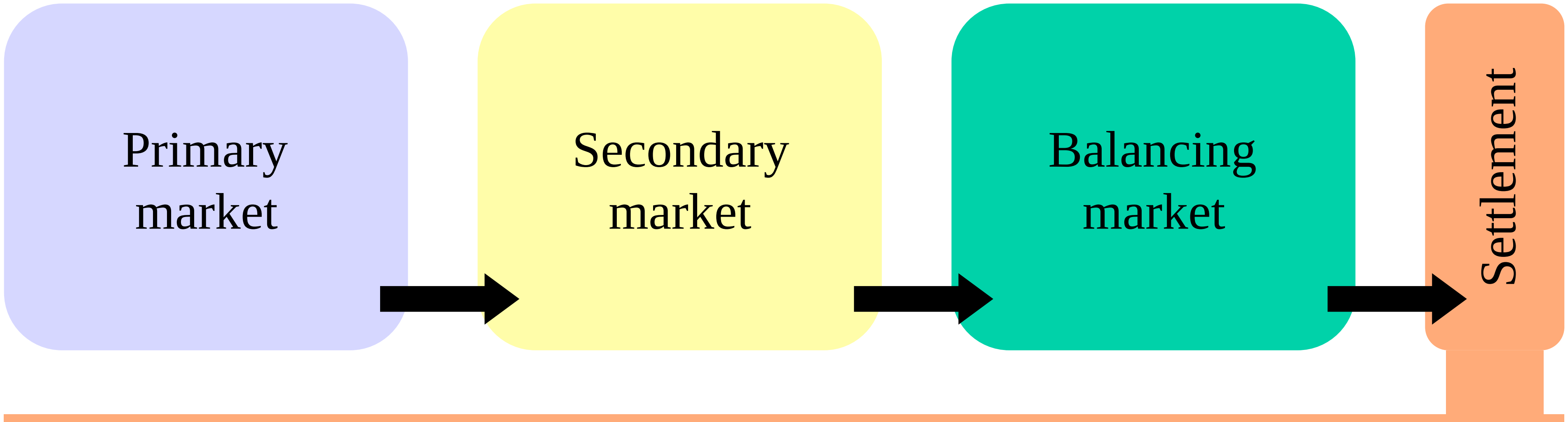
The scenario:

- Secondary market for the trading of transmision **tickets**.
- Lasts until the “gate closure”.
- Negotiation process.

The scenario:

- Allows the SO to maintain the voltage level and dynamic security.
- The SO can identify shortfalls or excesses of energy that will arise in the ticket window.
- The SO can: (i) dispatch additional generation,
  (ii) back-off scheduled generation.

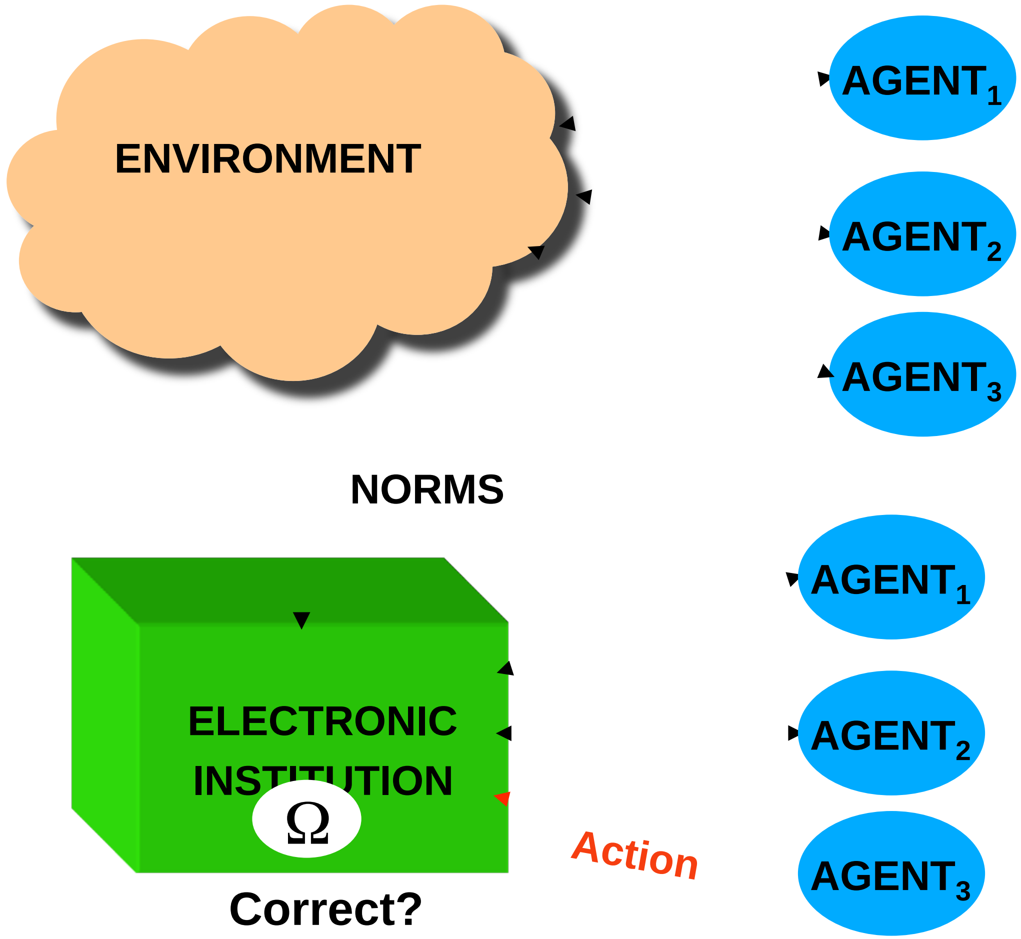
The scenario:



• Consumers pay producers for the power consumed.

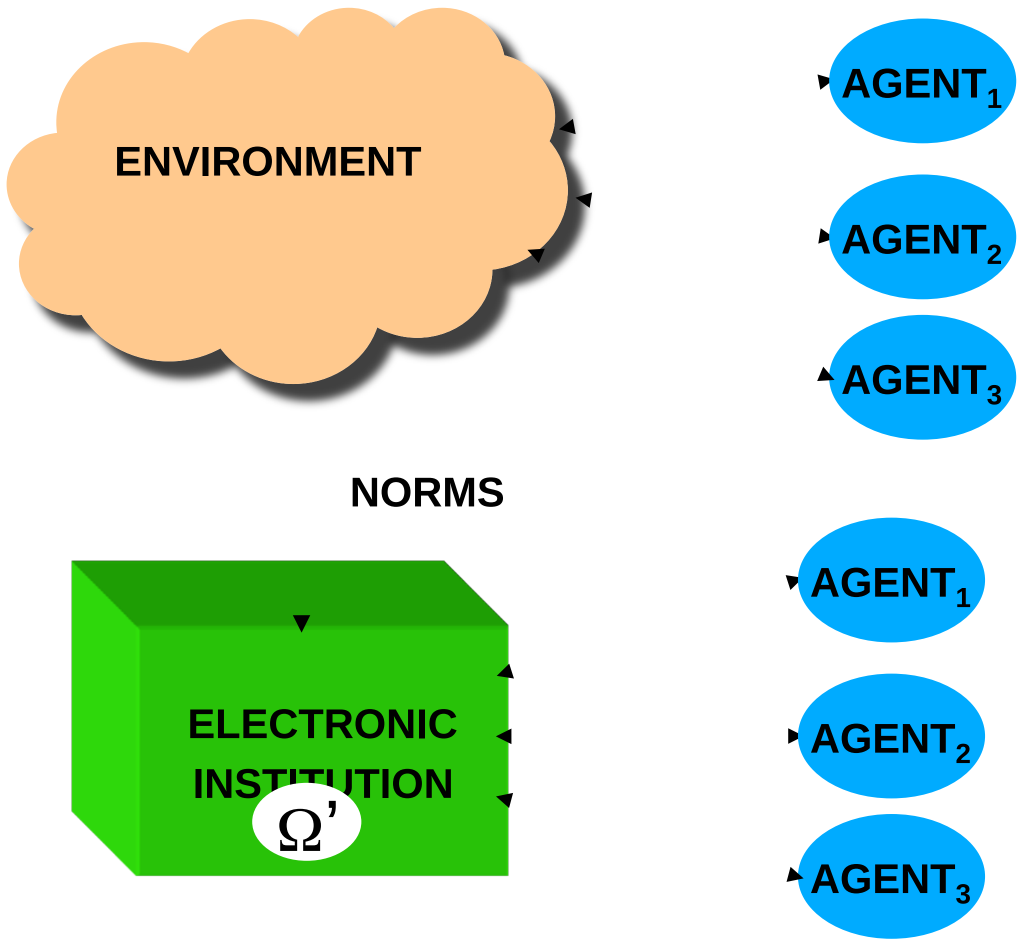
I. Introduction
II. Specifying electronic institutions
III. Running electronic institutions
IV. Electronic Institutions Development Environment
V. Virtual Institutions
VI. Conclusions

Electronic institution execution



EXECUTION STATE: Ω

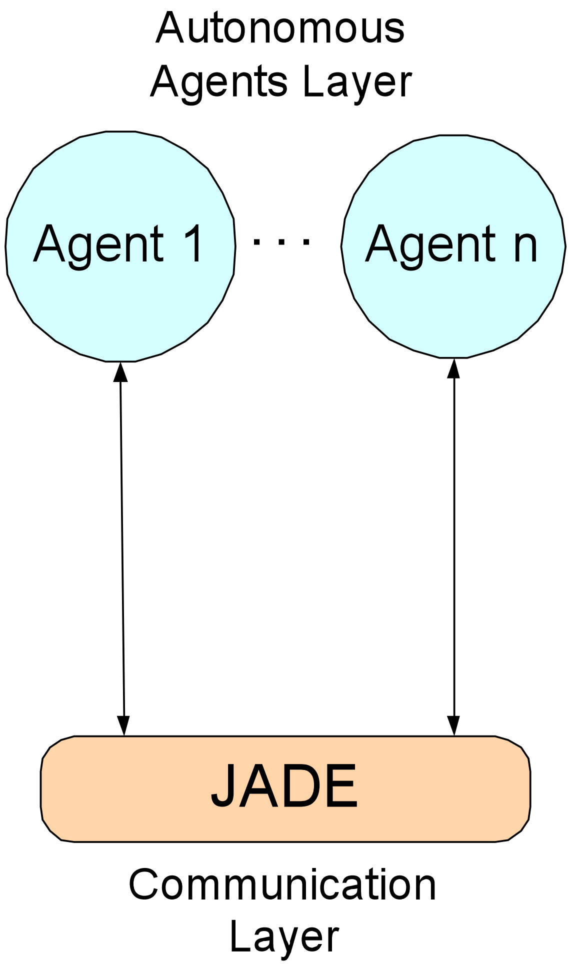
Electronic institution execution



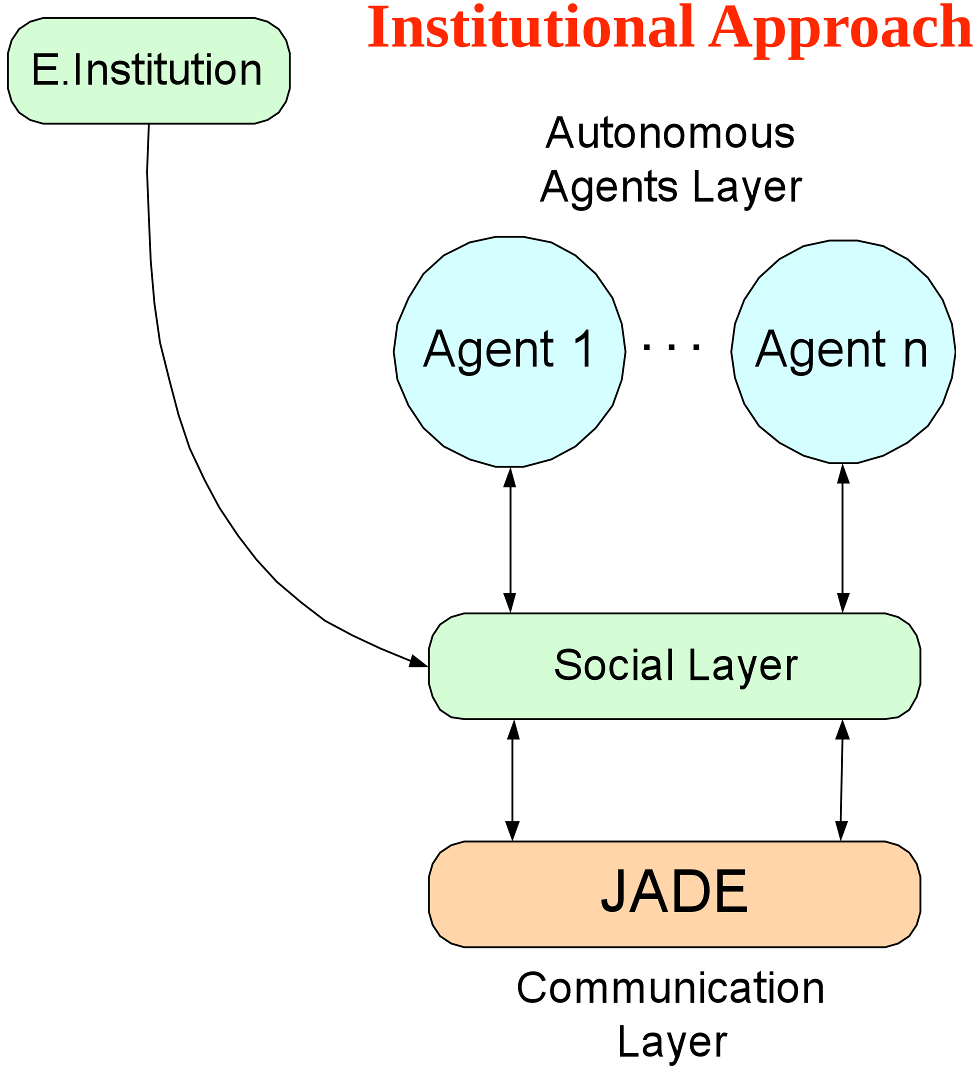
EXECUTION STATE: $\Omega \rightarrow \Omega'$

Electronic Institution Infrastructure

Traditional Approach



Institutional Approach

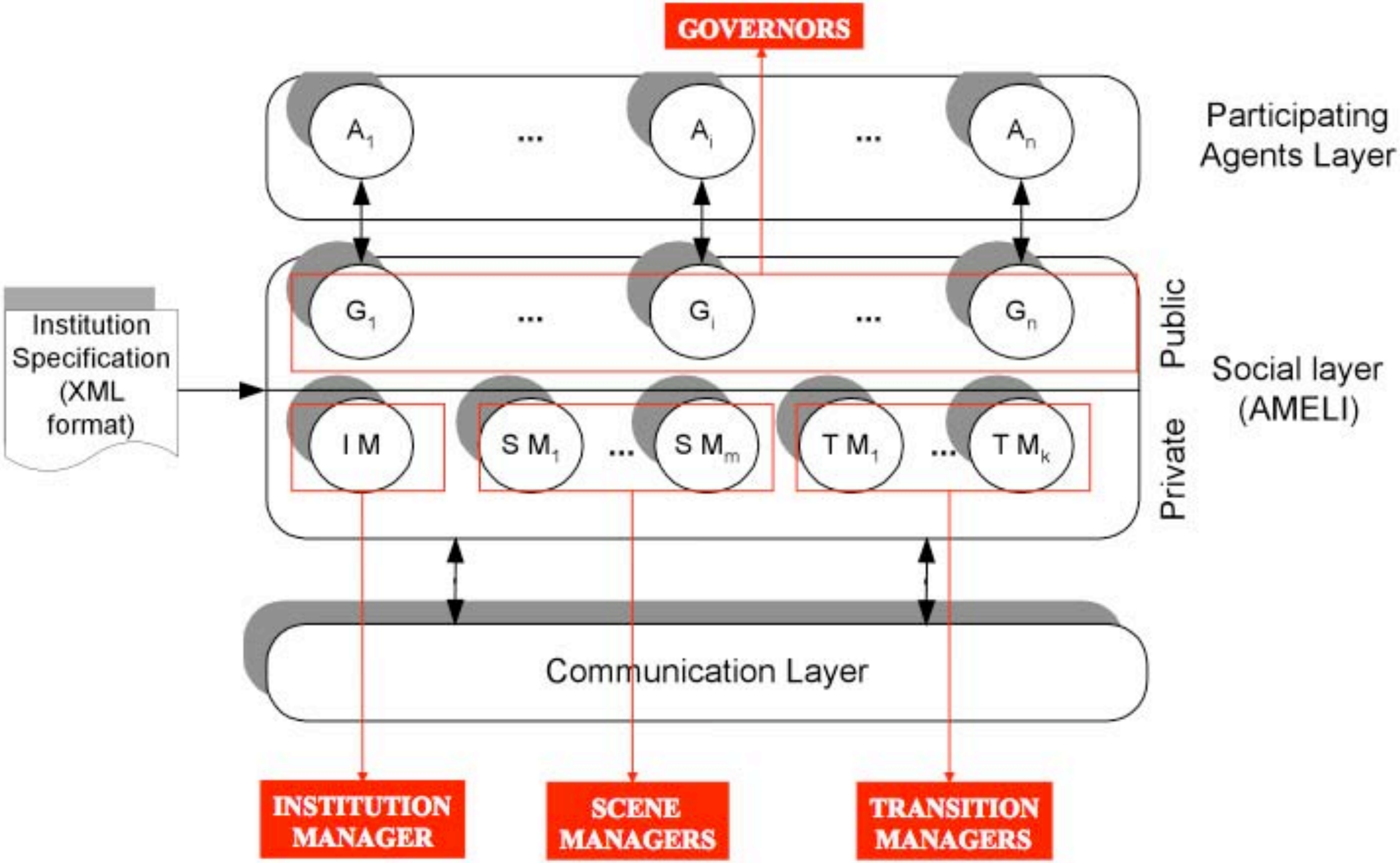


- **MEDIATION**
  - To facilitate agent communication within scenes.

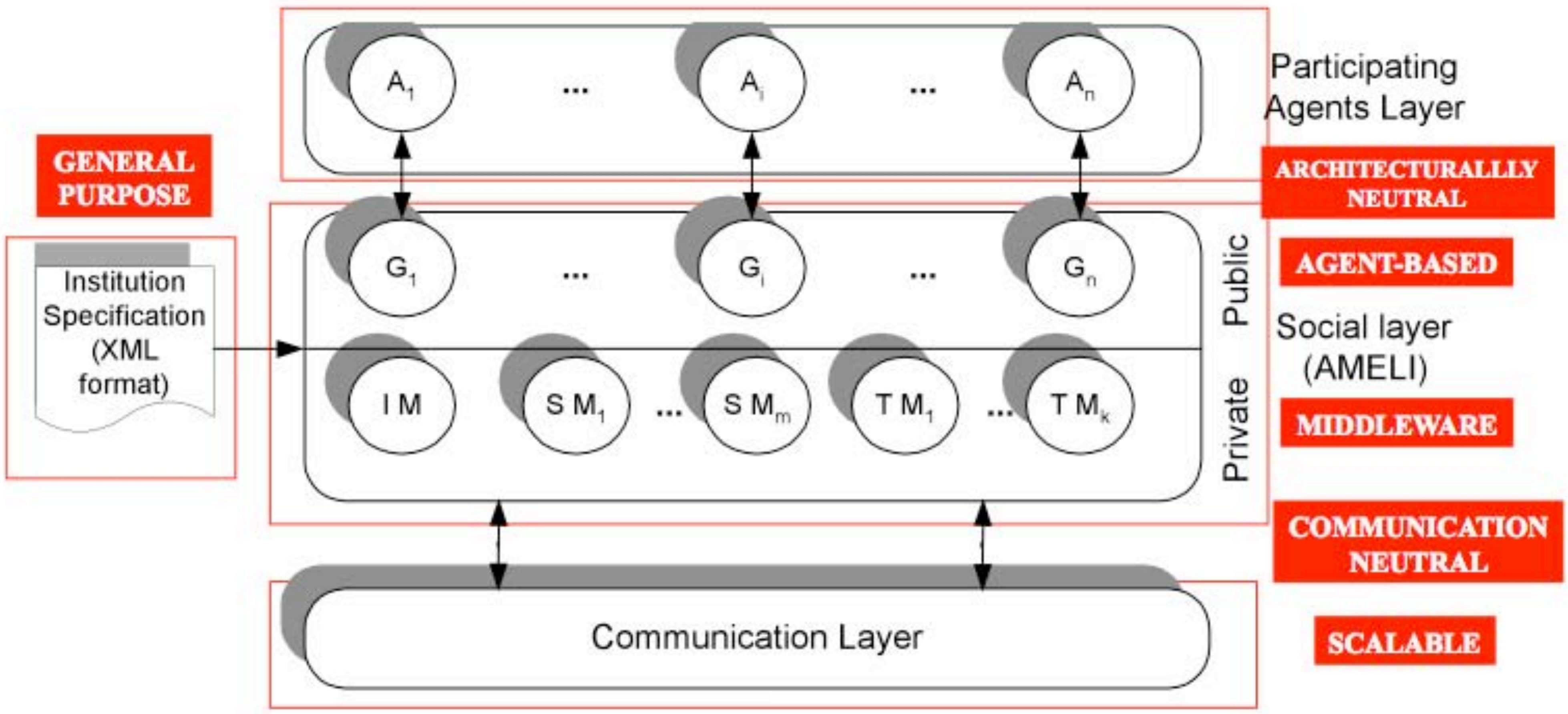
- **COORDINATION AND ENFORCEMENT**
  - To guarantee the correct evolution of each scene.
  - To guarantee legal movements between scenes.
  - To control the obligations participating agents acquire and fulfil.

- **INFORMATION MANAGEMENT**
  - To facilitate the information agents need in the institution.

AMELI architecture



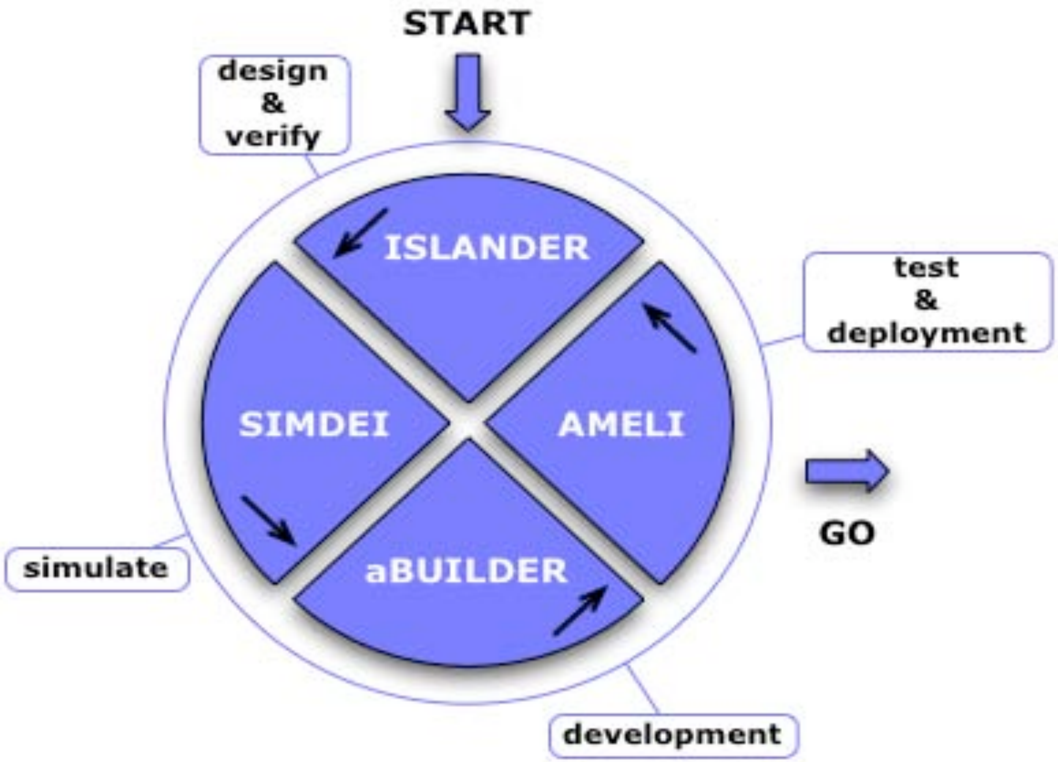
AMELI implementation features



I. Introduction
II. Specifying electronic institutions
III. Running electronic institutions
IV. Electronic Institutions Development Environment
V. Conclusions

Electronic Institutions Development Environment

http://e-institutions.iiia.csic.es



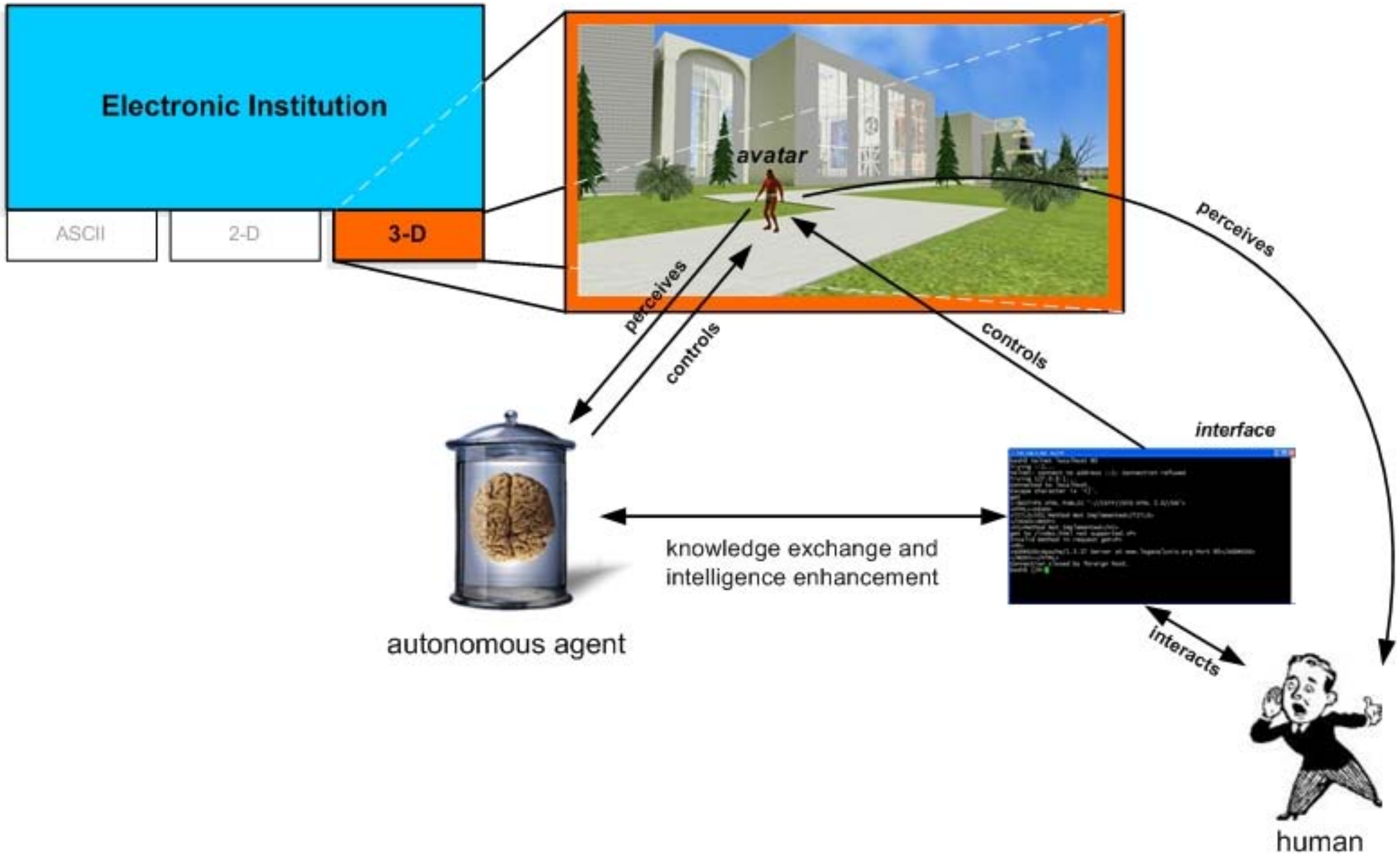
- Goal: To ease agent development
- Approach:
  - Graphical specification of an agent's inner behaviour
  - Automatic generation of agent skeletons.
  - Agent architecture based on tasks and performances
    - Performance – Actions whithin a particular scene
    - Task – Sequence of performances related by a performative structure path

- Goal: To verify dynamic properties of EIs
- Approach:
  - To run discrete event simulations
  - Support the simulation of Electronic Institutions with different agent populations.
  - What if analysis

I. Introduction
II. Specifying electronic institutions
III. Running electronic institutions
IV. Electronic Institutions Development Environment
V. Conclusions

- Engineering open multi-agent systems is a highly complex task.
- Electronic institutions reduce this complexity by introducing normative (regulatory) environments.
- We have presented **an Electronic Institutions Development Environment (EIDE) that facilitates the deployment of electronic institutions**.
- EIDE targeted at supporting **environment engineering** in open multi-agent systems.

Integrating human users.The Vision



Social Virtual World

- IIIA Researchers: Carles Sierra, Juan Antonio Rodriguez-Aguilar, Josep Lluis Arcos, Pablo Noriega

- IIIA developers: Bruno Rosell, David de la Cruz, Guifre Cuni