Developing MAS
The GAIA Methodology

A Brief Summary by
António Castro and Prof. Eugénio Oliveira

Outline

- GAIA Methodology Overview
- Analysis Phase
- Architectural Design
- Detailed Design
- Scopes and Limitations
- Full Cycle: Big Picture
- References

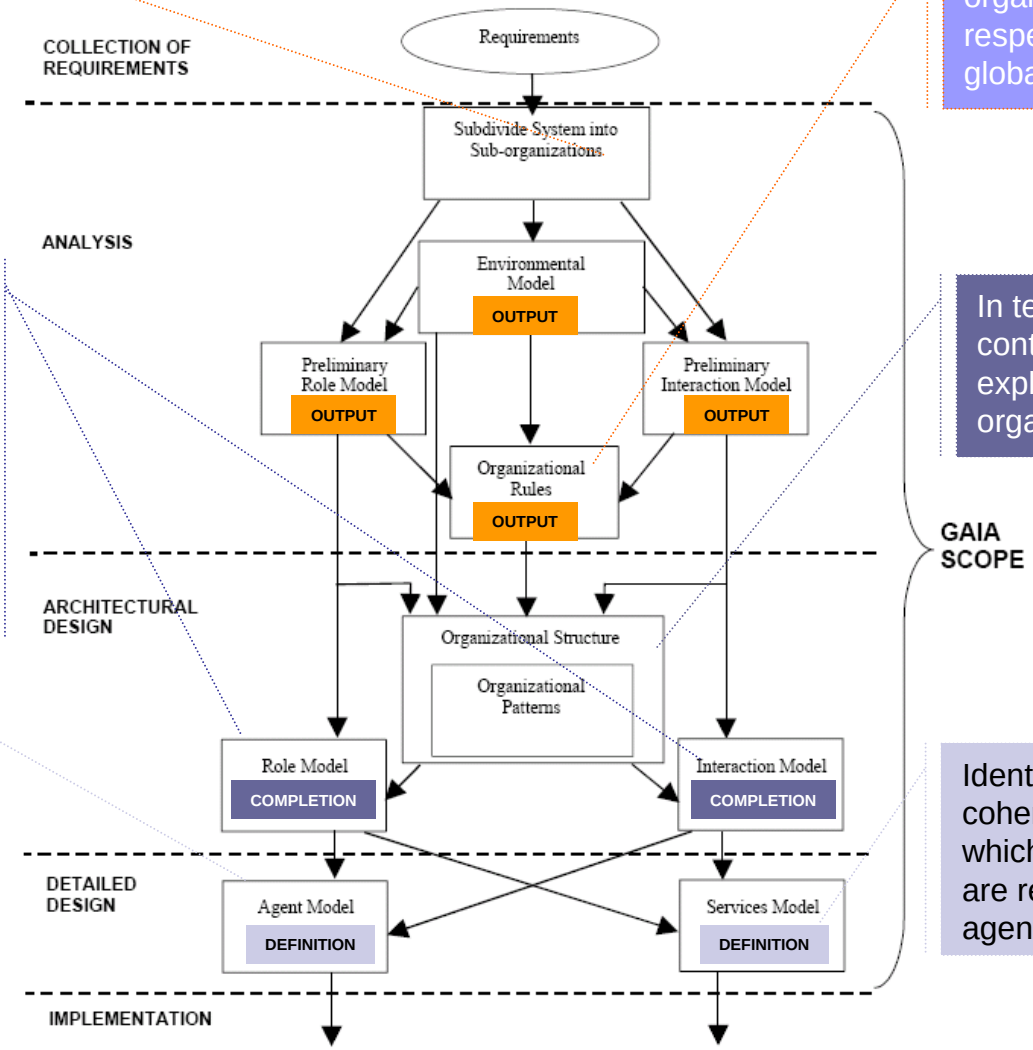
Overview of GAIA Methodology

The goals of the organizations that constitute the overall system and their expected global behavior

The rules that the organization should respect and enforce in its global behavior

Separating, when possible, the organizational independent aspects (detected in analysis) from the organizational dependent ones (derived from the adoption of a specific organizational structure).

In terms of its topology and control regime. Can also exploit catalogues of organizational patterns.



Identifies the agent classes that will make up the system and the agent instances that will be instantiated from these classes.

Identifies the main services – coherent blocks of activity in which agents will engage – that are required to realize the agent's roles and properties.

Analysis Phase I

- *Organizations*: determine whether multiple organizations have to coexist in the system and become autonomous interacting MAS.
- *Environmental Model*: abstract computational resources, such as variables or tuples, made available to the agents for sensing (read), for effecting (change) or for consuming (extract)

Analysis Phase II

- *Preliminary Role Model*: identify the “basic skills” that are required by the organization to achieve its goals, as well as the basic interactions that are required for the exploitation of these skills.
- *Preliminary Interaction Model*: captures the dependencies and relationships between the various roles in the MAS organization, in terms of one protocol definition for each type of inter role interaction.

Analysis Phase III

- *Organizational Rules*: responsibilities of the organization as a whole. These are safety (invariants that must be respected) and liveness (dynamics of the organization) organizational rules.

Preliminary Roles

Example of a Analysis Diagram

Preliminary Interactions

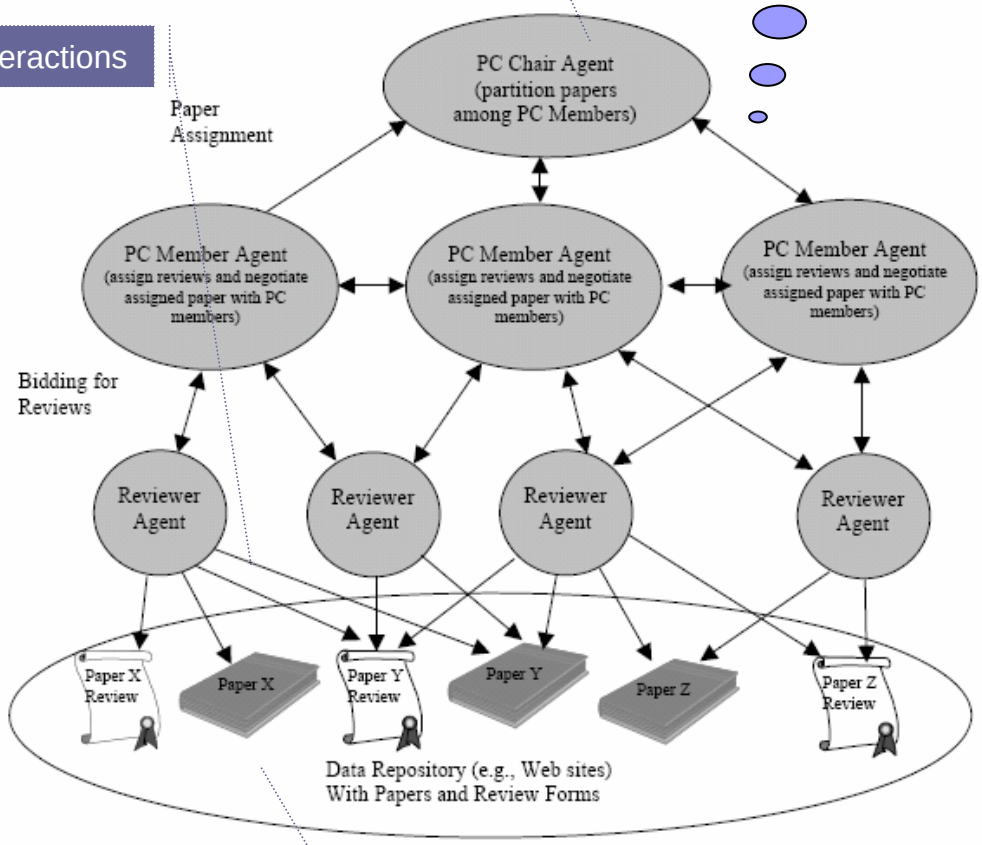


Fig. 5. Conference management: Multilevel hierarchy.

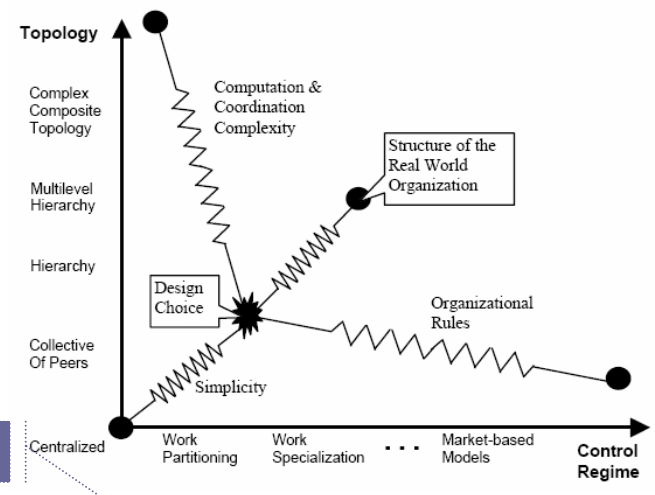
Environment Model

Architectural Design I

Including a detailed description of the semantics of the relations

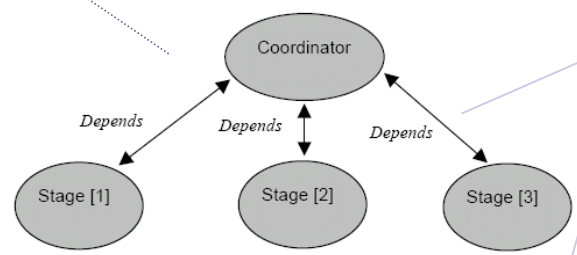
- *Organizational Structure*: identify the appropriate organizational structure, including, topology and control regime.

Graphical Representation



- *Organizational Patterns*: catalogue of possible modular and composable "organizational structures" that will help the designer.

Formal Representation



$$\forall i, \text{PC CHAIR} \xrightarrow{control} \text{REVIEWCATCHER}[i]$$

$$\forall i, j, \text{REVIEWCATCHER}[i] \xrightarrow{control} \text{REVIEWER}[j]$$

$$\forall i, j, \text{REVIEWCATCHER}[i] \xrightarrow{peer} \text{REVIEWCATCHER}[j]$$

Architectural Design II

- *Completion of Role and Interaction Models*: (1) define all the activities in which a role will be involved, (2) define organizational roles, (3) complete the definition of the protocols required by the application and (4) define organizational protocols.

Role Model

Role Schema: REVIEWER

Description:
This preliminary role involves receiving papers for review from some conference official, reviewing that paper, and sending back a completed review form.

Protocols and Activities:
ReceivePaper, ReviewPaper, SendReviewForm

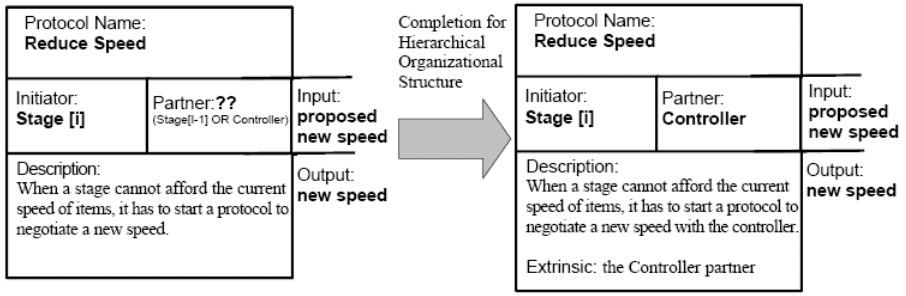
Permissions:

reads	*Papers*	*// all the papers it receives*
changes	*ReviewForms*	*// one for each of the papers*

Responsibilities
Liveness:

$$\text{REVIEWER} = (\text{ReceivePaper.ReviewPaper.SendReview})^{maximum_number}$$

Safety:
- *number_of_papers = number_of_reviewforms*



Interaction Model

Detailed Design I

- *Agent Model*: to define the agent model it is necessary to identify which agent classes are to be defined to play specific roles and how many instances of each class have to be instantiated in the actual system. The model can be defined using a simple diagram (or table) specifying, for each class, which roles will map it. In addition, the model can document the instances of a class that will appear in the MAS.

$\text{STAGE}^N \xrightarrow{play} \text{STAGE}$

$\text{COORDINATOR}^1 \xrightarrow{play} \text{COORDINATOR}$

Detailed Design II

- *Services Model*: identify the services associated with each agent class, or equivalently, with each of the roles to be played by the agent classes. For each service it is necessary to document its properties: inputs, outputs, preconditions and postconditions. The services are derived from the list of protocols, activities, responsibilities and liveness properties of the roles it implements.

Scopes and Limitations of GAIA

- Does not directly deal with particular modeling techniques. It proposes but does not commit to, specific techniques for modeling (e.g., roles, environment, interactions). In the future: "… AUML is a useful companion to GAIA."
- Does not directly deal with implementation issues. The outcome is a detailed but technology-neutral specification. Should be easy to implement with, for example, a FIPA-compliant agent system.
- Does not explicitly deal with activities of requirements capturing and modeling. In the future: "… integrate methods and techniques from goal-oriented analysis."

Full Cycle Big Picture

- Requirements Analysis using methods and techniques from goal-oriented analysis (Castro et al. 2002, Mylopoulos et al. 1999)
- Analysis and Design with GAIA Methodology
- AUML as a notation.

References

- F. Zambonelli, N. Jennings, M. Wooldridge
  Developing MAS: The Gaia Methodology
  ACM Vol. 12, N. 3, July 2003.
- J. Castro, M. Kolp, J. Mylopoulos
  Towards requirements-driven information systems engineering: The tropos project
  Inf. System Vol. 27 N. 6, June 2002.
- J. Mylopoulos, L. Chung, E. Yu
  From object-oriented to goal-oriented requirements analysis
  ACM Vol. 42, N. 1, January 1999
- J. Odell, H. Parunak, C. Bock
  Representing agent interaction protocols in UML
  Proc. 1st Int. Work. AOSE, Vol. 1957, 2001
- B. Bauer, J. P. Muller, J. Odell
  Agent UML: A formalism for specifying multiagent software systems.
  Int. Journal Soft. Eng. Knowl. Eng. Vol. 11 N. 3, April 2001