AOSE

The Tropos Methodology

AOSE: Tropos (1)

- **Tropos is a methodology proposed** by J. Mylopoulos, M. Kolp and P. Giorgini in the article " Agent Oriented Software Development " (2002, but since **2000** was matter of study) [14]
- This presentation is based on the **latest article** written by P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos and A. Perini "TROPOS: An Agent-Oriented Software Development Methodology " (**May 2004**) [15]

FIVE MAIN DEVELOPMENT PHASES:

- Early Requirements
- Late Requirements
- Architectural Design
- Detailed Design
- Implementation

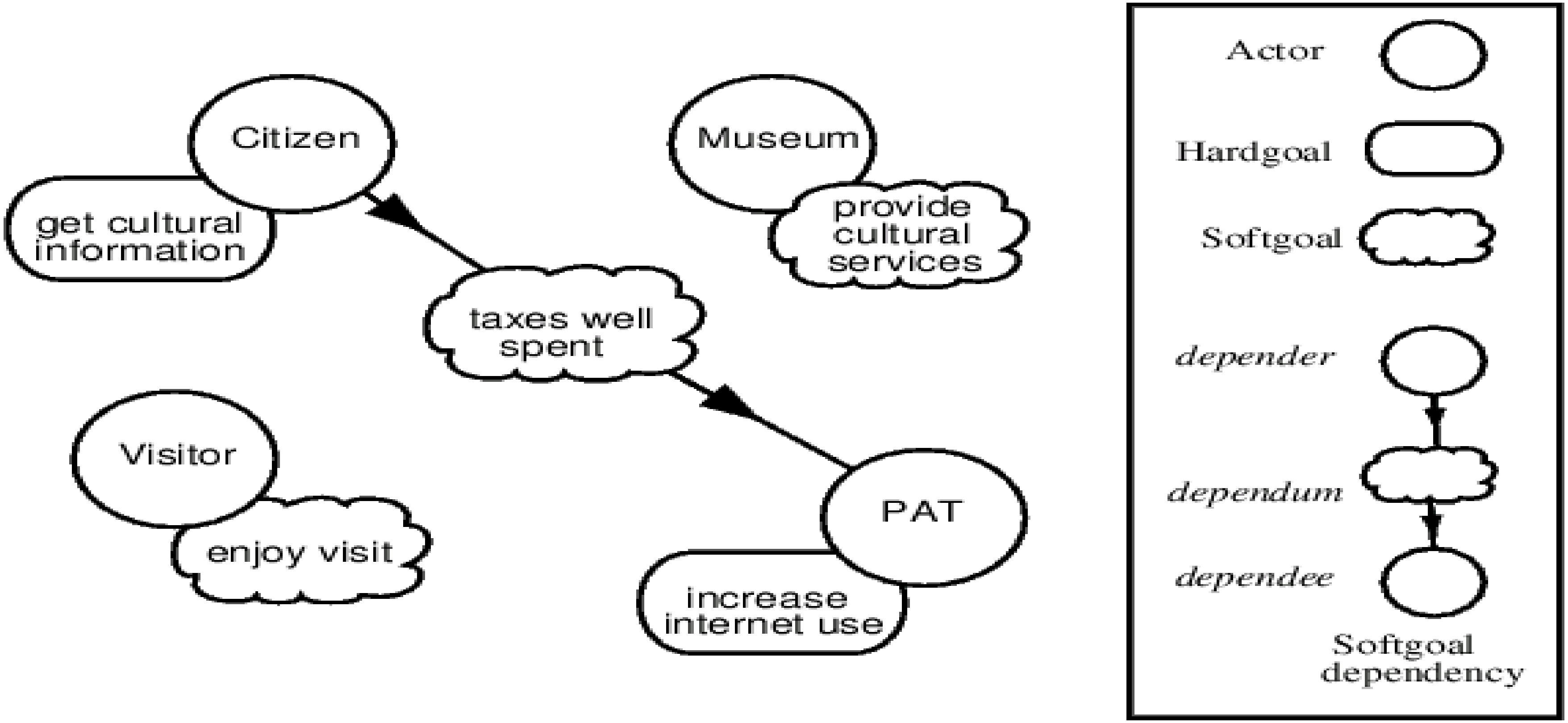
NOTE: Tropos from Greek "tropé" which means "easily changeable", also "easily adaptable"

AOSE: Tropos (2)

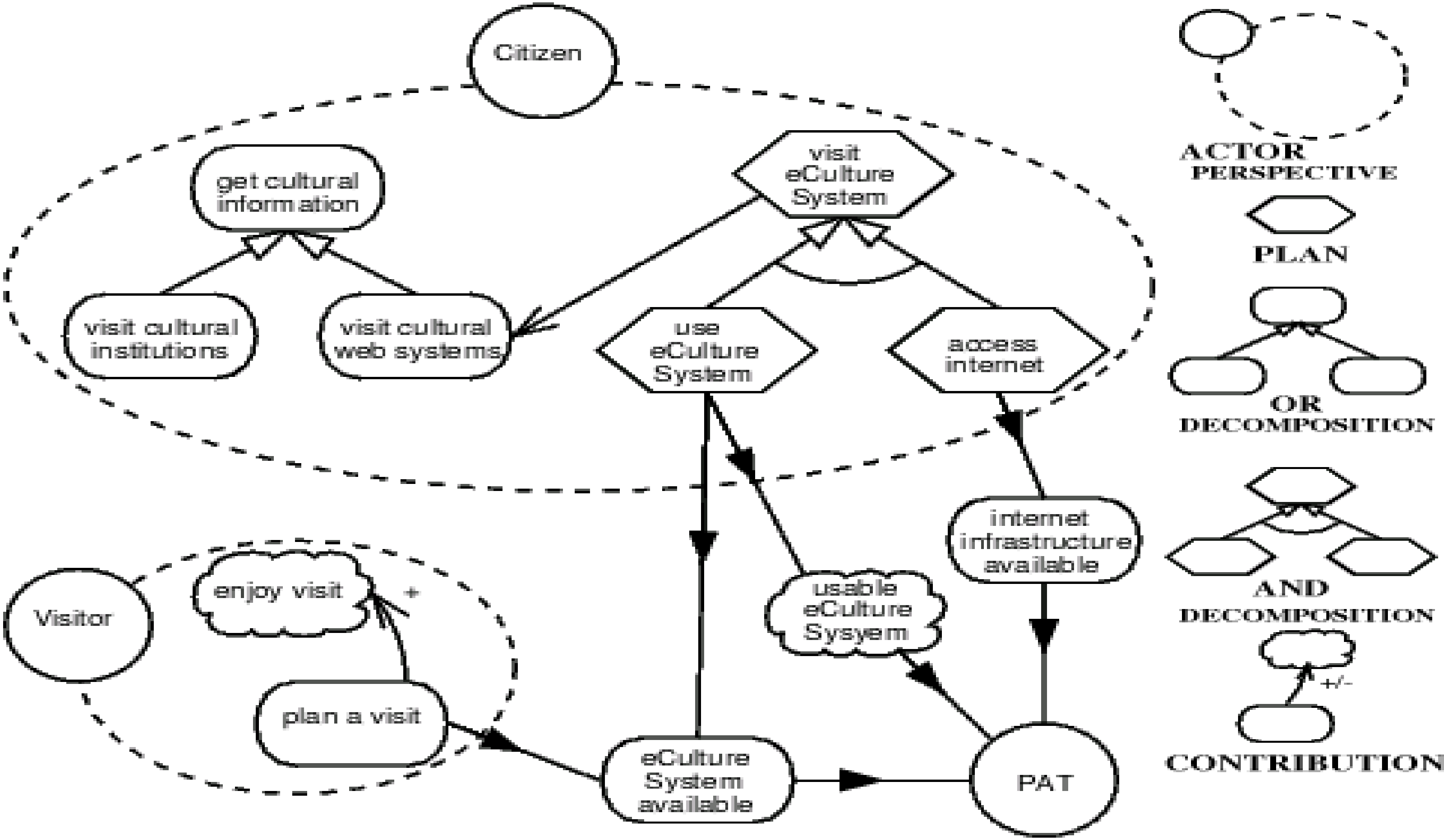
- MODELING ACTIVITIES:
  - **Actor modeling,** which consists of identifying and analyzing both the actors of the environment and system' s actors and agents
  - **Dependency modeling**
  - **Goal modeling** based on 3 basic techniques: means-end analysis, contribution analysis, and AND/OR decomposition
  - **Plan modeling**
  - **Capability modeling**

AOSE: Tropos (3)

- **EARLY REQUIREMENTS ANALYSIS:** consists of identifying and analyzing the stakeholders and their intentions. We must create Actor Diagrams and Goal Diagrams



AOSE: Tropos (4)

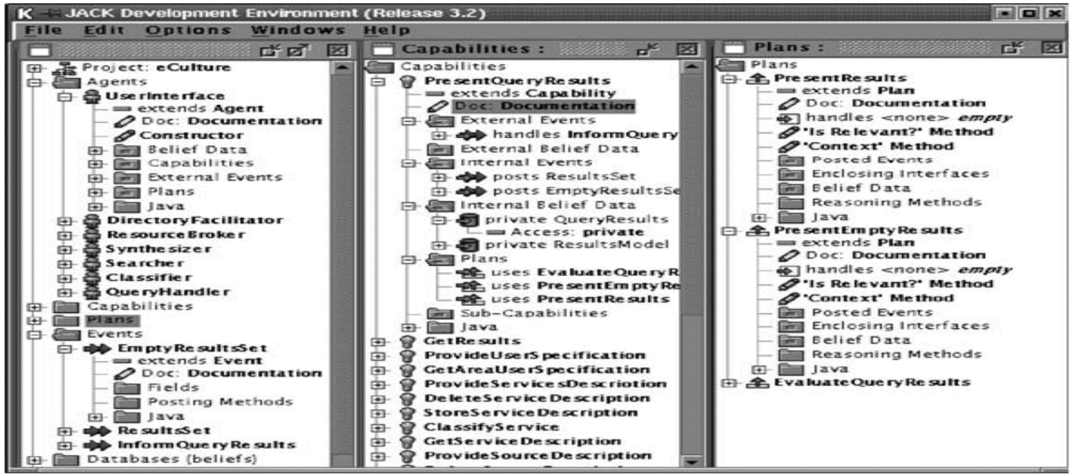


AOSE: Tropos (5)

- **LATE REQUIREMENTS ANALYSIS:** focuses on the **system-to-be within its operating environment**. System-to-be is represented **with a goal diagram** as one actor which has a number of dependencies with the other actors of the organization.
- **ARCHITECTURAL DESIGN:** defines **the system' s global architecture** in terms of sub-systems (actors) interconnected through data and control flows (dependencies). It is articulated in 3 steps:
  - Step 1: the **overall architecture** is defined (extended actor diagram)
  - Step 2: the **capabilities** is defined from actor dependencies
  - Step 3: a set of **agent types** with one or more different capabilities (agent assignment) is defined

AOSE: Tropos (6)

- **DETAILED DESIGN:** deals with the specification of the agents' micro level
  - **Capability diagrams:** model a capability with UML activity diagrams. In particular action states model plans
  - **Plan diagrams:** each plan node of a capability diagram can be further specified by UML activity diagrams
  - **Agent interaction diagrams:** AUML sequence diagrams
- **IMPLEMENTATION:** in JACK Intelligent Agents [16], an agent-oriented development environ


JACK Development Environment (Release 3.2)
File
Edit
Options
Windows
Help
Project: eCulture
Agents
UserInterface
extends Agent
Doc: Documentation
Constructor
Belief Data
Capabilities
External Events
Plans
Java
DirectoryFacilitator
ResourceBroker
Synthesizer
Searcher
Classifier
QueryHandler
Capabilities
Plans
Events
EmptyResultsSet
extends Event
Doc: Documentation
Fields
Posting Methods
Java
ResultsSet
InformQueryResults
Databases (beliefs)
Capabilities :
Capabilities
PresentQueryResults
extends Capability
Doc: Documentation
External Events
handles InformQuery
External Belief Data
Internal Events
posts ResultsSet
posts EmptyResultsSe
Internal Belief Data
private QueryResults
Access: private
private ResultsModel
Plans
uses EvaluateQueryR
uses PresentEmptyRe
uses PresentResults
Sub-Capabilities
Java
GetResults
ProvideUserSpecification
GetAreaUserSpecification
ProvideServicesDescriotion
DeleteServiceDescription
StoreServiceDescription
ClassifyService
GetServiceDescription
ProvideSourceDescription
Plans :
Plans
PresentResults
extends Plan
Doc: Documentation
handles <none> empty
"Is Relevant?" Method
"Context" Method
Posted Events
Enclosing Interfaces
Belief Data
Reasoning Methods
Java
PresentEmptyResults
extends Plan
Doc: Documentation
handles <none> empty
"Is Relevant?" Method
"Context" Method
Posted Events
Enclosing Interfaces
Belief Data
Reasoning Methods
Java
EvaluateQueryResults