

---

# Modeling-Notation Source:

## ADELPHE

Version 03-03-13 17:25EST  
Document author- Marc-Philippe Huget

### 1 Description

In this document, we consider the ADELFE notation. Actually, ADELFE does not refer to a notation but to a methodology. ADELFE stands for “Atelier pour le Développement de Logiciels à Fonctionnalité Emergente” (Toolkit for Designing Software with Emergent Functionality). The main difference between ADELFE and other agent-oriented methodologies is to take into account the adaptability into the systems, to cope the unpredictable events that occur in the environment in order to realize the “right” task.

ADELFE is based on the view of a multi-agent system as a dynamic organization consisting of various cooperative agents.

The aim of ADELFE is to cover all the phases of a classical software design from the requirements to the deployment. It is based on RUP methodology and uses the UML notation [Jacobson, 1999]. ADELFE reuses, as far as possible, in one hand the most parts of the RUP methodology and on the other hand UML extensions already done in agent-oriented software engineering.

Developing an adaptive multi-agent system with ADELFE passes by several phases:

1. Requirements
2. Analysis
3. Design
4. Implementation
5. Test

### 2 Requirements Workflow

The aims of this stage are to define the system to be, to transform this view in a use-case model, and to organize and to manage the requirements (functional or not) and their priorities. In fact, at this stage, the designer has to define the function of the studied system and to model its environment.

The definition of the studied system is given through keywords and concepts. For instance, in the timetabling problem [Bernon, 2002], we have the keywords planning, rooms, teachers, students, constraints, organizations and constraints managing.

The definition of the environment model has two steps: a detailed definition of the environment and the context.

The detailed definition of the environment describes the active and the passive entities in the environment. Active entities are teachers, students, planning manager and room manager in the timetabling problem. Passive entities are rooms and the National Pedagogic Plan.

The context of the system is the characterization of data flows and interactions between active or passive entities and the system. The output of this context is a set of collaboration diagrams and sequence diagrams.

Finally, the last step in the requirements workflow is to define use cases in order to clarify the different functionalities the system has to respond to.

### 3 Analysis Workflow

The analysis workflow is composed of several tasks: (1) domain analysis and architecture study, (2) adequacy of the AMAS theory, (3) agent identification, (4) adequacy of the AMAS theory at the local level, and (5) study of interactions between the different entities.

The domain analysis and architecture study defines the entities that could populate the system based on the study of the use cases. These entities are defined as class diagrams. Then, designers check what entities are useful in the system.

The adequacy of AMAS theory helps designers to know if AMAS theory is adequate to solve their problem.

The agent identification identifies which entities can be defined as an agent

The study of interactions step describes a set of activity diagrams and sequence diagrams, which explain the possible interactions between the different entities within the system.

53

## 54 **4 Design Workflow**

55 The design workflow is split in four steps: (1) detailed architecture and agent model, (2) cooperative agent  
56 architecture, (3) non cooperative situation model and (4) class diagrams.

57 The first step of the design requires to identify the software components and to describe them. The result provides  
58 the architecture of the system in terms of needed blocks, classes, agents and interactions. The agent model,  
59 which represents the relationships between agents, is included in this architecture.

60 The cooperative agent architecture helps the designer to fill in a generic architecture given for an agent used in  
61 the AMAS theory.

62 The non cooperative situation model represents the main contribution of ADELFE to this workflow. During it, the  
63 designer must fill up a table describing each non cooperative situation (NCS) encountered by each previously  
64 identified agent. This table contains the name of the NCS, the state in which the agent is when detecting it, the  
65 textual description of the NCS and the conditions and actions linked to it. The conditions describe the different  
66 elements that enable the agent to locally detect this NCS. The actions describe what the agent has to do to  
67 remove it.

68 Finally, the class diagram step refines the class diagrams previously defined.

69

## 70 **5 Implementation and Tests Workflow**

71 This workflow is similar to the one found in the RUP methodology.

72

## 73 **6 References**

74 [Bernon, 2002] Carole Bernon, Marie-Pierre Gleizes, Sylvain Peyruqueou and Gauthier Picard, « ADELFE, a  
75 Methodology for Adaptive Multi-Agent Systems Engineering » in Third International Workshop  
76 "Engineering Societies in the Agents World" (ESAW-2002), 16-17 September 2002, Madrid

77 [Bernon, 2002a] Carole Bernon, Marie-Pierre Gleizes, Gauthier Picard and Pierre Glize, « The Adelfe  
78 Methodology For an Intranet System Design » in Fourth International Bi-Conference Workshop on  
79 Agent-Oriented Information Systems (AOIS-2002), 27-28 May 2002, Toronto (Ontario, Canada) at  
80 CAiSE'02

81

82 systems (AAMAS 2003), Melbourne, Australia, July 2003.