Modeling
an OMG-MASIF Compliant Mobile Agent Platform with the RM-ODP Engineering Language

Florin MUSCUTARIU(*+) and Marie-Pierre GERVAIS(*)

* Laboratoire d'Informatique de Paris 6 (*), 8, rue du Capitaine Scott — F 75015 PARIS
+ CS TELECOM (+) 28, rue de la Redoute — F 92263 FONTENAY-AUX-ROSES Cedex
{Florin.Muscutariu, Marie-Pierre.Gervais}@lip6.fr

Abstract. In order to model telecommunications services as mobile agent system, we are defining a methodology based on the RM-ODP standards. Our approach makes the distinction between the service behavior specification, that is independent of the support environment, and the complete service specification that must take into account the target environment. To obtain this specification, the designer must be able to model the target environment according to the concepts used in the methodology, i.e., the RM-ODP concepts. We describe in this paper such a modeling activity. The target environment that we consider is an OMG-MASIF compliant mobile agent platform. We model it by using the RM-ODP engineering language.

Keywords: methodology for agent-based services development, mobile agent architecture, RM-ODP, OMG MASIF standard

1. Introduction

We will present in this paper a model of an OMG MASIF-compliant platform using the RM-ODP engineering language. This work is integrated in the ODAC[1] project developed at LIP6.

The ODAC project aims to provide a methodology for designing telecommunications agent based services using the Reference Model of Open Distributed Processing (RM-ODP)[6]. The methodology provides methods, models and tools to define, in a first step, the so-called "behavioral specification" of a telecommunications service, that is the description of a telecommunications service behavior as a set of interacting agents[2], i.e., a multi-agent system [2, 3]. This specification is independent of the

[1] ODAC stands for Open Distributed Applications Construction
[2] An agent in this paper is considered as a software entity that acts autonomously on the behalf of another entity (human, software or organization) [1].

support environment. The second step in the methodology is the targeting, the projection of the behavioral specification on an operational environment in order to define the so-called "operational specification". It must provide the telecommunications service designer with tools, first for modeling the target environment, and then for representing the service specification according to this environment model. Modeling the target environment must be according to the concepts used in the methodology, i.e., the RM-ODP concepts.

This paper is concentrating on this modeling activity. The target environment we consider is an OMG-MASIF compliant mobile agent platform, such as Grasshopper[3] [4, 5]. We focus on the engineering viewpoint of the RM-ODP, which deals with the mechanisms and functions required for the support of distributed interactions between ODP-objects in the system. Thus we provide the model of such a mobile agent platform by using these RM-ODP engineering concepts.

2. ODP Concerns

RM-ODP is a standard developed by ISO and ITU-T that defines an architectural framework within which support of distribution, internetworking and portability can be integrated in order to specify a distributed processing system [6]. The specification of a complete system is divided in viewpoints relevant to some particular area of concern during the design of the system. There are five viewpoints:

1. *Enterprise viewpoint*: a viewpoint on the system and its environment that focuses on the purpose, scope and policies for the system;
2. *Information viewpoint*: a viewpoint on the system and its environment that focuses on the information semantics and information processing performed;
3. *Computational viewpoint*: a viewpoint on the system and its environment that enables distribution through a functional decomposition of the system into objects which interact at interfaces;
4. *Engineering viewpoint*: a viewpoint on the system and its environment that focuses on the mechanisms and functions required to support distributed interactions between objects in the system;
5. *Technology viewpoint*: a viewpoint on the system and its environment that focuses on the technology choices for that system.

The computational viewpoint is concerned with “when” and “why” objects interact, while the engineering viewpoint is concerned with “how” they interact. In RM-ODP, an ODP *object* is defined as a model of an entity. A computational object is the basic entity resulting from the decomposition of the problem done previously during the specification process. The ODP objects involved in an engineering specification are called *engineering objects* (EO). A *basic engineering object* (BEO) is the engineering object that requires the support of a distributed infrastructure. It is the direct mapping

[3] Grasshopper is a mobile agent platform developed by IKV++ GmbH (www.ikv.de)

of a computational object. We consider that a basic engineering object (BEO) may be an agent.

The engineering language, used in the engineering viewpoint, defines the concepts of cluster, capsule, node, nucleus and channel. *A cluster* is a configuration of basic engineering objects forming a single unit for the purposes of deactivation, checkpointing, reactivation, recovery and migration. A *capsule* is a configuration of engineering objects forming a single unit for the purpose of encapsulation of processing and storage. A *node* is a configuration of engineering objects forming a single unit for the purpose of location in space, and which embodies a set of processing, storage and communication functions. The node is under the control of a *nucleus***,** which is responsible for creating groups of engineering objects, for making the communication facilities available and for providing other services.

A set of nodes could form an engineering interface reference management domain (IRMD). The IRMD is a naming domain and determines the policy for content, allocation, tracking and validation for the engineering interface references (EIR). The EIR is an identifier for an engineering object interface available for distributed bindings. A *channel* is a configuration of engineering objects providing a binding among a set of interfaces to basic engineering objects, through which interactions can occur. There is a relation of inclusion illustrated in the Fig.1.

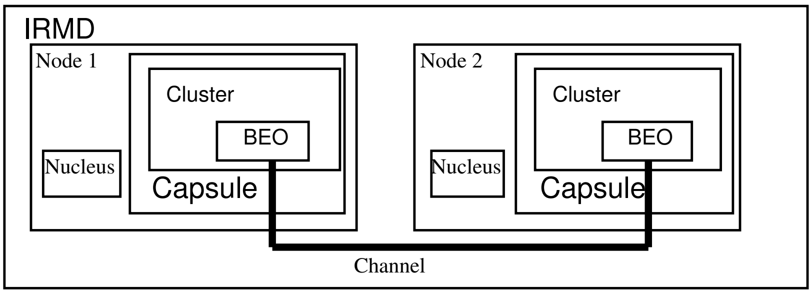

Fig.1. Hierarchical organization of the engineering objects

3. The OMG-MASIF Architecture

MASIF is about interoperability between agent systems [4]. The areas of MASIF contributions contain agent management, agent transfer, agent and agent system names, agent system types and location syntax. The architecture of a MASIF compliant distributed agent environment is composed of agents (stationary and mobile), agencies, places and regions. The *Agency* is the runtime environment, while the *place* provides a logical group of functionality inside an agency. A *Region Registry* maintains information about the components associated with a different region. The *Region* may exist to facilitate the management of the distributed components. The Core

Agency represents the functionality required by an agency for agent execution support (Fig2).

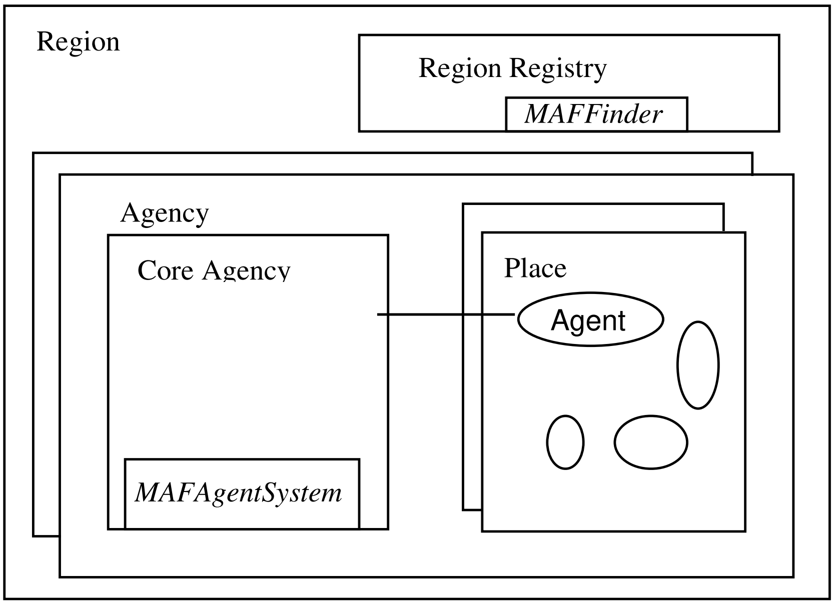

Fig. 2 Hierarchical Component Structure

MASIF is a collection of definitions and interfaces that provides an interoperable interface for mobile agent systems. Two interfaces are defined:

- MAFAgentSystem interface;
- MAFFinder interface.

4. Specification of a MASIF Platform Using the RM-ODP Engineering Language

An engineering specification defines the infrastructure required to support functional distribution of an ODP system by: a) identifying the ODP functions necessary to manage physical distribution, communication, processing and storage; b) identifying the roles of different engineering objects supporting the ODP functions.

In order to do this, we specify:

1. *A configuration of engineering objects*, structured as clusters, capsules and nodes;
2. *The activities* that occur within those engineering objects;
3. *The interactions* of the engineering objects.

For achieving that, we respect the engineering language rules such as: interface reference rules, binding rules, cluster, capsule and node rules, etc.

4.1. Engineering Objects Configuration

The implementation of an ODP *node* in a MASIF platform is an *agency*, and all the agents in an agency share the common processing, storage and communication. A node, implemented as an agency, is a member of an *interface reference management domain (ODP-IRMD)*, implemented as a *Region*. The *nucleus*, implemented as a c*ore agency*, provides a set of node management interfaces for each capsule within the node. The implementation of a *capsule* is a *place.* As we said before, the representation of a BEO may be an agent. If so, the agent autonomy makes us consider that we must have only one BEO per cluster. Actually, since an agent may migrate and initiate autonomous behavior, this reduces the number of agents per cluster to one. If the BEO is not represented as an agent but as a normal object, we may consider several BEOs in a cluster as an option.

According to the ODP rule establishing that an ODP-IRMD contains a set of ODP nodes, a region contains a set of agent systems (Table 1). The region must provide unambiguous object references in the naming context that correspond to the ODP-engineering interface references. This is done by implementing the MAFFinder naming service, which is an IDL interface defined in MASIF. The Agent System, viewed as an ODP-Node in the engineering viewpoint, provides a set of management methods and objects to support agent location and management tasks. These management methods are grouped in an ODP specification in the Nucleus. An agent, viewed as a BEO in the ODP specification, is located in a place. An ODP capsule represents a MASIF place.

MASIF Architecture	*ODP Engineering Viewpoint Architecture Description*
Region MAFFinder	Interface Reference Management Domain (IRMD)
Core Agency MAFAgentSystem	Nucleus

Agent System	Node
Place	Capsule
Agent or Object	BEO

Table 1. ODP engineering concepts vs. OMG-MASIF concepts

Agent communication is outside the scope of the MAF specification and is done by the implemented system communication infrastructure, e.g., CORBA object communication, RMI or RPC. Modeling these implementation choices would result in a channel specification in the ODP engineering viewpoint.

4.2. Engineering Objects Activities

The functions of an agent system are:

- Transferring an agent;
- Creating an agent;
- Providing globally unique agent names and locations;
- Supporting the concept of a region;
- Ensuring a secure environment for agent operations.

We specify these functions of an Agent System with the ODP functions.

The coordination of an agent transfer, which corresponds to the migration management of a single BEO, is done by the coordination migration function. This uses the cluster management function and the capsule management function delegated to the node level.

To create a BEO, the cluster management function is implemented at the node level. The node management function assigns an engineering interface reference within a nominated engineering interface reference management domain (IRMD). The IRMD information is maintained by the engineering interface reference tracking function. The security functions are responsible for the security policies established by a security authority in a security domain. The correspondences are shown in Table 2.

OMG MASIF	*ODP Functions*
Transferring an Agent	Coordination Migration Function
Creating an Agent	Cluster Management Function
Providing Globally Unique Agent Names and Locations	Node Management Function
Supporting The Concept of a Region	Engineering Interface Reference Tracking Function
Ensuring a Secure Environment for Agent Operations	Security Function

Table 2. ODP functions of an Agent System

The appendix details the classification of the MAFAgentSystem and the MAF-Finder interfaces methods as ODP functions.

4.3. Engineering Objects Interactions

MASIF addresses three types of interactions related to interoperability:

- Remote agent creation;
- Interaction needed for the agent transfer;
- Agent method invocation.

A client could be a non-agent program or an agent from an agent system having the same system type as the destination agent system or not. This client authenticates itself to the destination agent system and interacts with the destination agent system to request the creation of an agent. In an ODP specification, a BEO interacts with the Node for the creation of another BEO.

When an agent transfers to another agent system, the agent system creates a travel request providing information that identifies the destination place. In order to fulfill the travel request, the destination agent system transfers the agent's state, authority, security credential and the code. In ODP language, we say that the destination Node transfers the BEO and the data associated.

An agent invokes a method of another agent or object if it has the authorization and a reference to the object. This reference can be obtained from the MAFFinder. In the ODP specification, this is described as an interaction between a BEO and the IRMD.

4. Conclusions

The standardization process of the mobile agent environment reduces the necessary effort for target environment modeling. In this way, most of the standard target environments can be modeled using the corresponding standard. Depending on the implementation choices made for non-standard aspects, the full modeling process could be completed with particular mobile environment solutions.

Providing a model for different target environments is our goal in order to achieve "operational specification" at the end of the specification process, passing through the methodology that we are developing in the ODAC project.

The work in progress done by the international research community to standardize the mobile agent platforms, to unify and to merge the existing standards, permits a better efficiency in the methodology development corresponding to the actual or future platforms that are implementing these standards [7].

References

[1] "Mobile Software Agents: An Overview", VU Anh Pham, Ahmed Karmouch, IEEE Communication Surveys, 1999

[2] "Enhancing Telecommunication Service Engineering with Mobile Agent Technology and Formal Methods", Marie-Pierre Gervais, Alioune Diagne, IEEE Communications Magazine, July 1998

[3] " Specifying and Verifying the Behavior of Telecommunications Services " by J.F. Dauchez and M.P. Gervais,, in Proceedings of the 6th International Conference on Intelligence in Networks (ICIN'2000), Arcachon, France, January 2000

[4] The OMG Mobile Agent System Interoperability Facility (MASIF) Specification, http://www.omg.org/cgi-bin/doc?orbos/97-10-05;

[5] "Basic and Concepts", Grasshopper Development System, IKV++.

[6] Open Distributed Processing – Reference Model, ISO 10746/ITU-T X.90x

[7] OMG Agent Technology Green Paper, Agent Working Group, 1 March 2000

Appendix:

MAFAgentSystem methods	*RM-ODP Functions*
Create_agent	Cluster Management Function
Fetch_class	Related to RM-ODP technology view-point
Find_nearby_agent_system_of_profile	Trading Function
Get_agent_status	Cluster Management Function
Get_agent_system_info	Node Management
Get_authinfo	Authentication Security Function
GetMAFFinder	Node Management
List_all_agents	Trading Function
List_all_agents_of_authority	Trading Function
List_all_places	Trading Function
Receive_agent	Migration Function
Resume_agent	Cluster Management Function
Suspend_agent	Cluster Management Function
Terminate_agent	Cluster Management Function
Terminate_agent_system	Node Management Function

Appendix 1. The MAFAgentSystem methods described as RM-ODP functions implemented at the Node level.

MAFFinder methods	*RM-ODP Functions*
Register_agent	Engineering Interface Reference Tracking Function
Register_agent_system	
Register_place	
Lookup_agent	Trading Function
Lookup_agent_system	
Lookup_place	
Unregister_agent	Engineering Interface Reference Tracking Function
Unregister_agent_system	
Unregister_place	

Appendix 2. The MAFFinder methods described as RM-ODP functions at the IRMD level.