Design of a Community Manager System for a Multiagent Web Operating System.

[*]NIRIASKA PEROZO [†]JOSÉ AGUILAR

Resumen

Este artículo describe el diseño de un Sistema Manejador de Comunidades para un Sistema Operativo Web Multiagente. El sistema permite el agrupamiento de nodos en la comunidad de acuerdo a un criterio específico, con el objetivo de mejorar la eficiencia y el rendimiento del Sistema Operativo Web Multiagente. El diseño es desarrollado verificado según MASINA, una metodología para especificaciones de agentes.

Palabras Clave: Sistemas multiagentes, sistemas operativos web, autorganización, comunidades virtuales

Abstract

This paper describes the design of a Community Manager System for a Multiagent Web Operating System. The system allows virtual gathering of nodes in communities, according to specific criteria, with the objective of improving the efficiency and performance of the Multiagent Web Operating System. The design is made and verified according to MASINA, a methodology for agent specification.

Keywords: Multiagent Systems, Web Operating Systems, Self-Organization, Virtual Communities.

1 Introduction

Given the wide variety of unimaginable services everyday in the web, it is difficult to design an operating system to support those services individually. The Multiagent Web Operating System (MWOS) has been developed to cover these needs, with a main objective of providing a platform that allows users to benefit from the computational potential offered by the web, through resource sharing and problems solving of heterogeneity and adaptability. Thus, in [1] is proposed an architecture for a MWOS, defining it as an intelligent versioned operating system, which can be dynamically self-configured in order to allow easy and clear access to the resources of Internet. This MWOS is made up of four sub-systems amongst which the Community Manager System (CMS) is found. The MWOS demands great interaction between different nodes, since each one offers different services and applications. In addition, since the number of existing nodes can grow at an incredibly way, there is a need to gather them in communities dynamically created in a specific environment and context. Thus, the groups of nodes that continually interact with each other due to the relationships between them and their environment will form communities. This way, the architecture proposed in [1] for a MWOS defines a CMS with the goal of establishing the necessary policies for the dynamic and emerging creation, modification, and elimination of the existing communities in the MWOS. In this project we will create a detailed design of the CMS using the agent's theory, for which we will use the MASINA methodology [2], [3].

[*] *Unidad de Investigación en Inteligencia Artificial,Universidad Centroccidental "Lisandro Alvarado, Barquisimeto, Venezuela", Barquisimeto, Lara 3001*

[†] *CEMISID, Facultad de Ingeniería. Universidad de los Andes, Mérida, Venezuela 5101*

2 Theoretical Aspects

2.0.1 Background

The concept of community has been used and expanded to different areas, but it is currently having an important utilization in the Internet, which is one of the fastest and most penetrating phenomena in the society. The key consists in creating and maintaining virtual communities thought as efficient mechanisms of resource management over the Internet. Different works have been proposed at level of communities, for example the notions of Groups (users grouped according to common interests), Chat Rooms (online dialogue channels), Intranets (company employees using Internet technology), Extranets (people who do not belong to a given company, using that organization's intranet), among others have been used.

Particularly, the notion of virtual communities emerges from Internet. Among the most relevant works in this domain we find: "Adaptive Web-Based Database Communities" (creating data base communities specialized by area of interest) [6], "Adaptive Content Management in Structured P2P Communities" (using an adaptive, distributed algorithm to duplicate and replace contents in a community) [7], "Building Adaptive E-Catalog Communities Based on User Interaction" (it creates communities of electronic catalogs that continually adapt and restructure themselves in a dynamic environment) [8], "Ws-Catalog-Net: An Infraestructure for Creating Peering, and Querying e-Catalog Communities" (it consists of the creation of communities for making flexible the search when the information is not available locally in the portals) [11], "Toward Self-organizing service communities" (it is a structure where the communities of catalogue of services are made, linked, checked and adapted for constant interactions) [12], "Virtual Clusters for Grid Communities" (it allows to the grid's authorized clients, to negotiate the creation of virtual groups ("clusters") on virtual machines which are formed for satisfying the client's requirements in a determined time) [13]. The communities presented in these studies have adaptive, intelligent and emerging properties.

But the MWOS not only considers the communities as a means to integrate resources through isolated entities (like federations in Jini [4]), but the evolving entities developed within a specific context and/or define a set of implicit or explicit relationships between its components (like WOS TM [9], [10]) in a multiagent system.

2.1 Proposed MWOS

The architecture of the MWOS studied here is detailed in [1], let us take a look at the most relevant aspects. Our MWOS is a versioned and intelligent operating system that dynamically self-configures in order to allow an easy access to resources distributed over the Internet. The basic model of the proposed MWOS is composed of four subsystems: **the Resource Manager Subsystem (RMS)** offers the means to locate and assign resources in the MWOS, and is also responsible of activities such as the management of the inference engine, offering the configuration of a given service according to the user's requirements as a final result. **The Repository Manager Subsystem (LRMS y RRMS)** provides techniques to organize the information stored in the local and remote repositories and also coordinates the access to these. The **Community Manager Subsystem (CMS)** establishes mechanisms that allow the grouping of nodes into communities organized by attributes or properties that must be present in these. And the **Web Object Manager Subsystem (WOMS)** provides mechanisms to manage the migration and replication of Web objects over the Internet in order to reduce the number of communications and to improve the tolerance to failures in the system.

Due the dynamics present in the Web and the large number of nodes that can be connected to the MWOS, **communities** are created, which simply are a virtual gathering of the nodes that belong to the MWOS, exhibiting functional and behavioral similarities. The use of communities will optimize the search of any service, since instead of looking node by node; the search will be done community by community.

3 Description of the Proposed Community Manager System

3.0.1 Proposal

The nodes interact in the MWOS through mechanisms of request/response and negotiation. The nodes in the MWOS are defined according to their skills and behavior. These are the elements considered as node features, which as a whole (with its resources and elements among others) is associated to a given community, that is, it's grouped according to certain similarities with other nodes (see figure).

The CMS must have protocols for the communication between nodes in a given community and between the different communities. Additionally, it must have mechanisms to manage the elements that belong to a given community. In addition, there are protocols that manage the creation, modification, and elimination of communities, and in general, the incorporation, elimination or migration of nodes between the different existing communities.

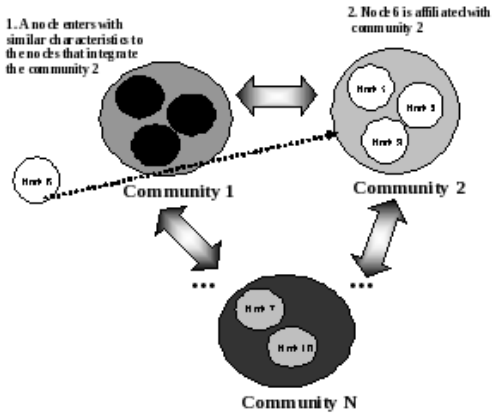

Figure 1: Integration of a node to the set of communities of the MWOS

3.1 Architecture

In the CMS of the MWOS, the operations are coordinated by two components: the community manager and the search manager. A description of each one of these units (the architecture of the CMS is shown in figure 2) is seen below.

Community Manager: it is the main unit in the CMS, since it's in charge of managing the existing communities in the MWOS. It characterizes a new emerging community in the MWOS through its community characterization module, and with its updating module it is in charge of creating, eliminating and modifying a given community. Additionally, it allows the location of some communities with certain features or profiles through its community search module, finally, through its node characterization module; it is in charge of describing a node to be incorporated into a given community.

Search Manager: it is the unit in charge of receiving requests for search services from an RMS or from another CMS, and of processing them through its service processing module. It conducts the search for a given service, primarily through its internal search module, and if that is not successful, through its external search module.

For designing purposes we must take into account the following CMS features:

Each node in the MWOS holds information about the communities to which it belongs. Additionally, each node is capable of belonging to various communities, and each community belongs to a given type of community, understanding for a type of community a subset of attributes generated from the CMS generic template. For each type of community, ranges are established for each attribute, and different

communities are derived for this type of community according to the values of their attributes. The search for services is carried out first at the Intra level (in the communities to which the node belongs), and in case of failure, at the Inter level (communities external to the node).

Each community is capable of self-organizing and adapting to its environment. Communities emerge when a node is incorporated into the MWOS with features or a profile that does not fit into existing communities, or to lodge a node that needs to be part of a community different from the currently existing ones, because its features have changed and do not match with one of the current communities. A community disappears when it is empty that is, without any nodes.

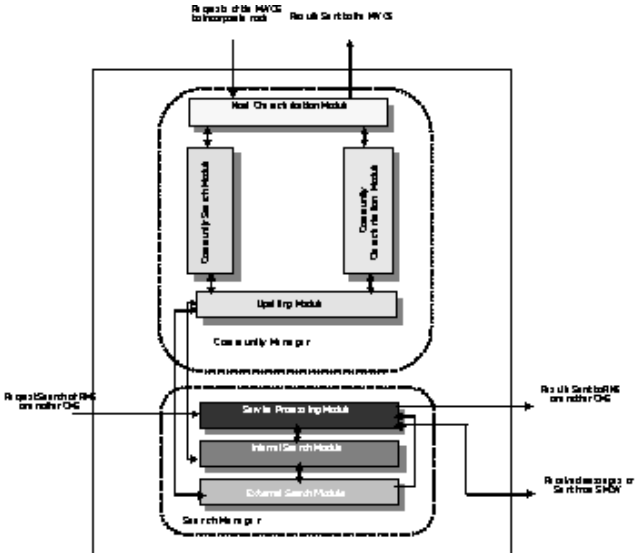

Figure 2: Architecture of the CMS

4 Design

The CMS can be seen as a system made up of intelligent agents capable of cooperating to obtain the solutions to problems related to the management of communities. This way, the CMS is designed following an approach based on Multi-agent Systems (MASINA [2]) considering the following phases: conceptualization, analysis, design and implementation. The conceptualization phase consists of defining the actors and cases of use (see figure y).

In the analysis phase the models for agents, tasks, communication, coordination and intelligence are developed [2]. For the purpose of this paper, we will first present the agents and tasks of the CMS and then an example of the model for agents and intelligence of the Search Coordinator Agent. Agent identification is carried out based on the actors defined in the conceptualization phase. Depending on the roles defined during this phase, we have the following functionalities: community updating and service location at the community level. We will keep all of these actors as agents of our system. We can then identify two agents, which will be called:**Community Administrator Agent (CAA) and Search Coordinator Agent (SCA)**. These two agents will be distributed in each node of the MWOS. Let's next see an example of the agent model of the Search Coordinator Agent [5].

Finally, the Intelligence Model lets us describe the aspects that give an agent its intelligence, such as its reasoning, learning and experience accumulating mechanisms. We next present the intelligence model for the search coordinator agent.

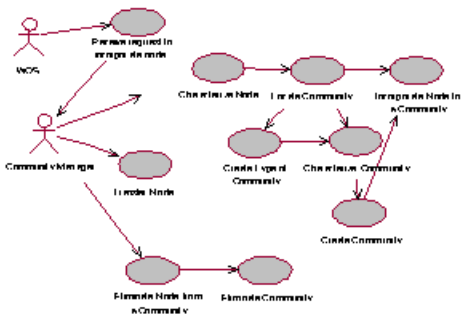

Figure 3: Use Cases of the Actor Community Manager of CMS

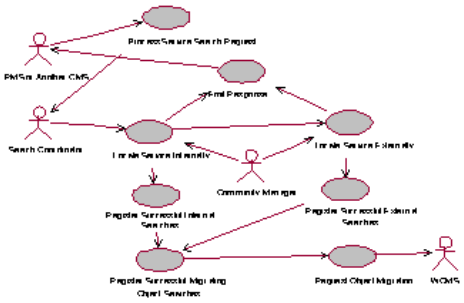

Figure 4: Use Cases of the Actor search coordinator of CMS

5 Design Verification and Obtained Results

The CMS was verified following a verification method proposed in [3] for the MASINA methodology [2]. Verification was made at two levels: Macro and Micro.

At the **macro level** it was verified that the CMS design satisfies the objectives of the MWOS. Regarding the CMS, we can say that it is divided into two agents: the Search Coordinator Agent and the Community Administrator Agent, which help satisfy the objectives set for the CMS. Additionally, the CMS relationships described in figure 2 are kept by the CMS agents, and the tasks to be carried out by the CMS are distributed between the two agents.

At the **Micro Level**, the model cross proposed in [3] was carried out. The results achieved for example for Agent model cross with the others models were the following:

Agents Model vs. Tasks Model: it was verified that every task in the CMS be assigned to one of the CMS agents; all the ingredients required or provided by the tasks are provided or received by an agents defined in the MWOS; every CMS agent satisfied its objectives by carrying out the tasks defined

AGENT	**SEARCH COORDINATOR (SC)1.1**
NAME	Search Coordinator
TYPE	Software Agent: Reactive Agent
ROLE	Management of service requirements.
POSITION	It's an agent with specific tasks related to locating a particular service.
DESCRIPTION	This is the agent in charge of providing mechanisms that help locate a required service, aiming to satisfy a received request.

Tabla 1: Search Coordinator Agent (SCA)

	OBJECTIVE - SC AGENT 1.2
NAME	Locating a given service.
TYPE	Persistent Objective.
INPUT PARAMETERS	Service information (ID, Name, Version).
OUTPUT PARAMETERS	Service location.
ACTIVATION CONDITION	To receive a search request from the RMS or another CMS.
SUCCESS CONDITION **FAILURE CONDITION** **REPRESENTATION LANGUAGE**	The location of the required service is obtained. Exit condition, the allowed waiting time expires. Natural language.
ONTOLOGY DESCRIPTION	CMS Service ontology The SC agent has as its objective to provide search mechanisms that allow for the location of a required service.

Tabla 2: SC Agent Objective

	SERVICE - SC AGENT 1.3
NAME	Service finding
TYPE	Free, concurrent
INPUT PARAMETERS	Service information (ID, Name, Version)
OUTPUT PARAMETERS	Service address found. Exception parameters such as: service not found, estimated waiting time expired.
REPRESENTATION LANGUAGE	Natural language.
ONTOLOGY	CMS Service ontology

Tabla 3: SC Agent Service

	SERVICE - SC AGENT 1.4
NAME	Statistics control
TYPE	Free, concurrent
INPUT PARAMETERS	Successful outcome in the search for a service.
OUTPUT PARAMETERS	Entry with Node, Community and service location. It's necessary the use frequency of migrating objects used to satisfy a service.
REPRESENTATION LANGUAGE	Natural language.
ONTOLOGY	CMS Service ontology

Tabla 4: SC Agent Service

	SERVICE PROPERTY - SC AGENT 1.5
PROPERTIES	Service performancel
	Service reliability

Tabla 5: SC Agent Service-Property

in the tasks model and the capacities required by the tasks are provided by the agents that carry them out (defined in the agents model). **Agents Model vs. Communication Model**: It was verified for each act of speech in the designed communication model, the used ontologies, the involved agents, and who initiated them, coinciding with the defined agents and the ontologies known by each CMS agent.

	SERVICE PROPERTY - SC AGENT 1.5.1
NAME	Performance
VALUE	Bad, Good, Excellent.
DESCRIPTION	This property has to do with the response time that is achieved after carrying out all the necessary activities in order to satisfy a require service

Tabla 6: Agent Service-Property Performance

	SERVICE PROPERTY - SC AGENT 1.5.2
NAME	Reliability.
VALUE	Reliable, Non-reliable.
DESCRIPTION	This property has to do with the SC, agent's ability to give useful, timely

Tabla 7: Agent Service-Property Reliability

	GENERAL CAPACITY - SC AGENT 1.6
ABILITIES	It has the capacity to intelligently coordinate the search
	of a particular service, both at the internal level (searching in the communities to which the node requesting the service belongs (Intra)) as well as at the external level (looking in the rest of the communities (Inter)) and for this purpose it has the ability to compare patterns. It's additionally in charge of initiating an object's migration process following the established object migration policies
REPRESENTATION LANGUAGE	Natural language.

Tabla 8: Agent General Capacity

	RESTRICTION - SC AGENT 1.7
RULES	The search coordinator agent must activate efficient search mechanisms in order to provide acceptable response times. It's the agent responsible of remote searching for a service in the MWOS, thus, its actions are not subject to any other agent in the MWOS
PREFERENCES	The requested service is satisfied at the internal level as long as possible(Intra search)
CLEARANCE	Only the AC, RMS, SMOW agents have access to the search coordinator agent.

Tabla 9: SC Agent Restriction

Agents Model vs. Coordination Model: It was validated that the agents involved, the objects to be satisfied, the required capacities, the provided services and the synchronizations required by the conversations in the CMS coordination model correspond to the agents, objectives, capacities and services defined in the CMS agent model. **Agents Model vs. Intelligence Model**: it was verified that the reasoning capacities expressed in the intelligent model for the Community Administrator and the Search Coordinator agents, correspond to the reasoning capacities defined for the Community Administrator and the Search Coordinator agents in the CMS agent model.

	REASONING MECHANISM 2.1
INFORMATION SOURCE	Results from service searches.
ACTIVATION SOURCE	Service search request.
INFERENCE TYPE	Inductive
REASONING STRATEGY	Diffuse expert system, neuronal nets for pattern recognition and data mining; search mechanisms based on evolving computation. It's necessary to generalize, based on experience, the conduction of the search itself in an easier way in the future (induction). To face unknown situations in SC Agent Reasoning Mechanism

Tabla 10: SC Agent Restriction

	LEARNING MECHANISM 2.2
NAME	Diffuse Classifying System
TYPE	Adaptive
REPRESENTATION TECHNIQUE	Rules
LEARNING SOURCE	Historic information, stemming off the dynamic search results.
UPDATING MECHANISM	Rule modification. Learning is updated considering previous experiences

Tabla 11: SC Agent Learning Mechanism

	EXPERIENCE 2.3
REPRESENTATION	Rules
TYPE	Case-based
DEGREE OF RELIABILITY	Moderate

Tabla 12: SC Agent Experience

6 Conclusions

Through the proposed CMS the MWOS could optimize the management of the services requested at the community level. It helps manipulate and search through communities, and it also could contribute to offer acceptable response times. Additionally, considering that the grouping of nodes and the updating processes of existing communities are carried out in a dynamic and emerging way, the proposed CMS could allow the MWOS to achieve these objectives in a flexible way. Thus, this CMS proposal is perfectly adaptable to the necessities of a Multiagent Web operating system, since in the Web there are not anything predefined due to its dynamic and volatile nature. Further work involves implementing a simulation prototype, for a specific problem, in order to instantiate each agent and component, validating and evaluating the design from an implementation point of view.

References

[1] Aguilar, J., Perozo, N., Ferrer, E., Vizcarrondo, J. *"Architecture of an Operating System based on Multiagent Systems"*, Lecture Notes in Artificial Intelligence, Springer-Verlag, Vol. 3681, pp. 700-706, 2005.

[2] Aguilar, J., Rivas F., Hidrobo F., Cerrada M. *"Análisis y Diseño de Sistemas Multiagente usando la Metodología MASINA"*, Technical Report - Universidad de los Andes, January 2004.

[3] Aguilar, J., Perozo, N., Vizcarrondo, J. *"Definition of a Verification Method for the MASINA Methodology"*, International Journal of Information Technology, Springer-Verlag, Vol 12, No.3, 2006.

[4] Morgan S. *"Jini to the Rescue"*, IEEE Spectrum, 37(4):44-49, 2.000.

[5] Perozo N. *"Diseño de un Sistema Manejador de Repositorios Locales y un Sistema Manejador de Comunidades para un Sistema Operativo Web"*, Tesis de Maestra, Universidad de los Andes, 2.003.

[6] Bouguehaya A. *"Adaptive Web - Based Database Communities"*, Department of Computer Science, Virginia Tech, USA, 2.002.

[7] Kangasharju J, Ross, K. *"Adaptive Content Management in Structured P2P Communities"*, http://citeseer.nj.nec.com/571891.html, 2.002.

[8] Paik H., Benatallah B. *"Building Adaptive E-Catalog Communities Based on User Interaction"*, IEEE Intelligent Systems, November-December p.p. 2-10, 2.002.

[9] Kropf P., Plaice J. *"WOS Communities - Interactions And Relations Between Entities in Distributed Systems"*. 1.999.

[10] Kropf P., Plaice J. *"Intentional Communities"*, Proceedings of Distributed Communities on the Web (DCW'2000) workshop, 2.000.

[11] Baina, K., Benatallah, B., Paik, H.Y., Toumani, F., Rey, C., Rutkowska, A., Harianto, B. *"WS-CatalogNet: An Infraestructure for Creating, Peering, and Querying e-Catalog Communities"*, Proceedings of the 30th VLDB Conference, Toronto-Canada, 2004.

[12] Paik, H.Y., Benatallah, B., Toumani, F. *"Toward self-organizing service communities"*, System, Man and Cybernetics, Part A, IEEE Transactions. Mayo, 2005.

[13] Foster, I., Freeman, T., Keahey, K. Scheftner, D., Sotomayor, B., Zhang, X. *"Virtual Clusters for Grid Communities"*, Proceedings of the 3rd Europena Across Grids Conference, 2005.