PATROCINADORES

Universidad de Granada

Ayuntamiento de Granada

ciber

ENTIDADES ORGANIZADORAS

Comunidad de Docentes e Investigadores de las Escuelas y Facultades de Informática de distintas Universidades Españolas.

ATC- ugr http://atc.ugr.es
Departamento de Arquitectura y Tecnología de Computadores de la Universidad de Granada.

ADIE http://chico.inf-cr.uclm.es:8080/adie/index.html
Asociación para el Desarrollo de la Informática Educativa.

SISTEDES
Sociedad de Ingeniería del Software y Tecnologías de Desarrollo de Software.

AENUI http://www.aenui.org/" \t "_blank
Asociación de Enseñantes Universitarios de la Informática.

AEPIA http://www.aepia.org
Asociación Española para la Inteligencia Artificial.

AERFAI http://www.aerfai.org/index2.phtml
Asociación Española de Reconocimiento de Formas y Análisis de Imágenes.

AIPO http://grihoudl.es:8080/aipo
Asociación Interacción Persona-Ordenador.

EUROGRAPHICS http://www.eg.org
Capítulo Español de la European Association for Computer Graphics.

EUSFLAT http://www.eusflat.org/index.htm
European Society for Fuzzy Logic and Technology.

SC of the IEEE CIS
Capítulo Español de la IEEE Computational Intelligence Society.

RADISC
Red Andaluza en Sistemas Complejos.

SEPLN http://www.sepln.org
Sociedad Española para el Procesamiento del Lenguaje Natural.

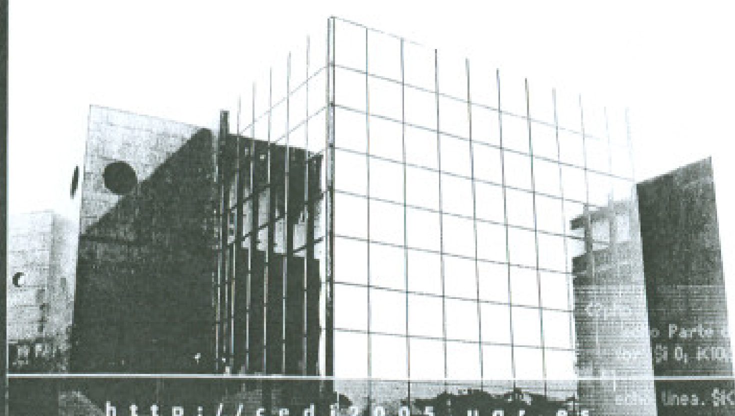
TIN - MEC
Programa Nacional de I+D en Tecnologías Informáticas. Ministerio de Educación y Ciencia.

CEA-IFAC http://www.cea-ifac.es
Comité Español de Automática de la International Federation of Automatic Control.

ISTANET http://www.istanet.net
Red Andaluza de Tecnología de Sistemas Inteligentes.

W3C http://www.w3c.es
Consorcio World Wide Web. Oficina Española.

RADI-AEB
Red Andaluza de Algoritmos Evolutivos y Bioinspirados.



ISBN: 84-9732-454-4

http://cedi2005.ugr.es

THOMSON

II Taller en Desarrollo de Sistemas Multiagente [DESMA'2005]

CEDI 2005

THOMSON

CEDI 2005

I CONGRESO ESPAÑOL DE INFORMÁTICA

GRANADA DEL 13 AL 16 DE SEPTIEMBRE

II Taller en Desarrollo de Sistemas Multiagente

[DESMA'2005]

EDITORES

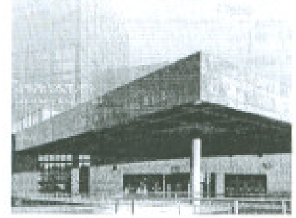
Juan C. González Moreno • Pedro Cuesta Morales • Jorge J. Gómez Sanz

Nuevos retos científicos y tecnológicos en Ingeniería Informática

I CONGRESO ESPAÑOL DE INFORMÁTICA

CEDI 2005

Nuevos retos científicos y tecnológicos en Ingeniería Informática



ACTAS DEL

II Taller en Desarrollo de Sistemas Multiagente [DESMA'2005]

EDITORES

Juan C. González Moreno
Pedro Cuesta Morales
Jorge J. Gómez Sanz

TALLER ORGANIZADO POR

Grupo Web de Agentes Inteligentes, Universidad de Vigo
Grupo de Agentes Software Ingeniería y Aplicaciones, Universidad Complutense de Madrid

ENTIDADES COLABORADORAS

Red Temática AgentCities.ES

THOMSON

THOMSON

Actas del II Taller en Desarrollo de
Sistemas Multiagente [DESMA'2005]

© Los Autores

Editores de la serie de Actas del CEDI

Rafael Molina Soriano
Antonio Díaz García
Alberto Prieto Espinosa

Editores de las Actas del presente Taller

Juan C. González Moreno
Pedro Cuesta Morales
Jorge J. Gómez Sanz

Diseño de Cubiertas

dixi
www.dixi-e.com

Impresión

THOMSON

COPYRIGHT© 2005 International
Thomson Editores Spain
Paraninfo, S.A.
Magallanes 25 · 28015 Madrid España
Tel: 91 446 33 50 · Fax: 91 445 62 18
clientes@paraninfo.es

Impreso en España
Printed in Spain

ISBN: 84-9732-454-4
Depósito legal: SE-3927-2005 European Union
Printed by Publidisa

Antiguo Hotel Reuma. Granada
Foto realizada para el CEDI2005 por DIXI

II Taller
Desarrollo de Sistemas Multiagente
(DESMA-2005)

13 de septiembre, Granada (España)

Organizadas por:

Juan C. González Moreno, Pedro Cuesta Morales
Grupo Web de Agentes Inteligentes (gwai!)
Departamento de Informática
Universidad de Vigo

Jorge J. Gómez Sanz
Grupo de Agentes Software Ingeniería y Aplicaciones (grasia!)
Departamento de Sistemas Informáticos y Programación
Universidad Complutense de Madrid

Comité Científico:

- Josep Ll. Arcos - IIIA (CSIC)
- Juan A. Botía - Universidad de Murcia
- Vicent J. Botti - Universidad Politécnica de Valencia
- Luigi Ceccaroni - Universidad Politécnica de Cataluña
- Juan M. Corchado - Universidad de Salamanca
- Rafael Corchuelo - Universidad de Sevilla
- Ulises Cortés - Universidad Politécnica de Cataluña
- Luís A. García-Fernández - Universidad Jaume I
- Ana García-Serrano - Universidad Politécnica de Madrid
- Alma Gómez - Universidad de Vigo
- Vicente J. Julián - Universidad Politécnica de Valencia
- Rosalía Laza - Universidad de Vigo
- Beatriz López - Universidad de Girona
- Michael Luck - University of Southampton (UK)
- José M. Molina - Universidad Carlos III de Madrid
- Antonio Moreno - Universidad Rovira i Virgili
- Juan Pavón Mestras - Universidad Complutense Madrid
- Joaquín Peña - Universidad de Sevilla
- Ana M. Sánchez - Universidad de Granada
- Aida Valls - Universidad Rovira i Virgili
- Javier Vázquez-Salceda - Universidad de Utrecht
- Aurora Vizcaíno - Universidad de Castilla la Mancha

Comité organizador

- Juan C. González Moreno - Universidad de Vigo
- Pedro Cuesta Morales - Universidad de Vigo
- Jorge J. Gómez Sanz - Universidad Complutense Madrid
- Pedro Villar Castro - Universidad de Granada

Estas actas contienen los artículos seleccionados para su presentación en el segundo taller sobre Desarrollo de Sistemas Multiagente (DESMA-2005). El objetivo principal de este taller es recoger experiencias prácticas en el desarrollo de software orientado a agentes, que permitan definir los aspectos clave de este proceso y marquen las líneas futuras de investigación en este campo.

Este año se nos ha dado la oportunidad de integrarnos en el Primer Congreso Español de Informática (CEDI'2005), que pretende servir de marco de encuentro para profesionales dedicados preferentemente a la investigación, desarrollo, innovación y enseñanza universitaria, dentro del ámbito de la Ingeniería Informática.

El CEDI se celebra por primera vez en España, y para ello ha elegido una ciudad a la altura de las ocasiones. Granada, además de contar con una prestigiosa Universidad, es una ciudad que el viajero no olvida. Sus monumentos, calles y gentes le otorgan ese encanto especial que queda en la memoria.

Aprovechando el alojamiento de honor que se nos ha concedido, este año hemos hecho un esfuerzo especial por atraer a los investigadores en agentes. En esta edición se ha decidido valorar especialmente aquellos trabajos referentes a implementaciones de Sistemas Multiagente. Los artículos aceptados exponen trabajos de interés que acercan la tecnología de agentes al mundo industrial. Estos artículos se han estructurado entorno a tres bloques temáticos:

- Tecnologías de Agentes
- Metodologías de desarrollo de Agentes
- Agentes móviles e interacción

Hay que resaltar la gran aceptación de la convocatoria. Este año el DESMA ha contado con 15 contribuciones, de las cuales 12 han pasado el proceso de revisión. Este proceso ha dispuesto un mínimo de tres evaluadores por artículo y un máximo de cuatro. En general, las contribuciones recibidas han hecho gala de una calidad notable. Todas ellas constituyen un esfuerzo importante por acercar los agentes al nivel de madurez que todos deseamos.

Por ello, hay que felicitar a todos los autores el interés mostrado para participar en esta iniciativa. No nos cabe la menor duda de que esta experiencia resultará enriquecedora para todos. Además, queremos agradecer tanto a Thomson por la publicación de estas actas, como al CEDI, por facilitarnos la organización de este taller, que estamos seguros que promoverá los resultados del DESMA y ayudará a promocionar ediciones futuras. Para finalizar los agradecimientos, queremos reconocer públicamente el apoyo dado por la Red Temática AgentCities.ES (http://grusma2.etse.urv.es/AgCitES), una iniciativa sin la que la investigación en agentes en España no sería lo que es hoy en día.

Como dijo Machado, *"...caminante, no hay camino, se hace camino al andar ..."*, por ello queremos animar a todos los investigadores en agentes a continuar con su labor y a diseminar su trabajo, sobre todo en España. Todavía queda mucho por hacer y, si lo hacemos juntos, no cabe duda de que lograremos que el DESMA crezca y se convierta la conferencia de referencia en agentes software en España.

Granada, 13 de Septiembre de 2005
El Comité Organizador

CONTENIDOS

Tecnologías de Agentes

Developing a Multi-Agent Knowledge Management System with INGENIAS

Oscar M. Rodríguez-Elias[1], Ana I. Martínez-García[1], Aurora Vizcaíno[2], Jesús Favela[1], Mario Piattini[2]
1 CICESE, Departamento de Ciencias de la Computación, México
{orodrigu | martinea | favela}@cicese.mx
2 Grupo Alarcos, Universidad de Castilla-La Mancha, Escuela Superior de Informática, España
{Aurora.Vizcaino | Mario.Piattini}@uclm.es

Abstract

The literature proposes some characteristics that methodologies for the development of agent oriented systems should provide. They should assist in all the phases of the development process of an agent oriented system; for example, by providing models and views for the different phases of analysis and design, and tools to support the methodology. INGENIAS is an agent oriented methodology that covers most of the required characteristics proposed by the literature. In this paper we describe our experience using INGENIAS to develop a multi-agent knowledge management system for a software maintenance group. We present some of the main models developed to describe the system structure and behaviour; focusing on the advantages and disadvantages that we have found using this methodology. Finally, we present some of the functionality of a prototype developed based on the models proposed by INGENIAS.

1. Introduction

The role of agent oriented methodologies is to assist in all the phases of the lifecycle of an agent-based application, including its management [4]. Thus, an agent oriented methodology must provide some elements that enable this. These elements could be grouped into four main categories [10]: *concepts and properties* are basic notions about the domain area where the methodology will be applied; for example, notions of agent and its characteristics; *notations and modelling techniques* are related to the specific symbols used in the methodology for representing the concepts and properties (the modelling language); *process* indicates which stages of the software development cycle are covered by the methodology; and finally, *pragmatics* considers aspects related to the management and use of the methodology; for example, facility and costs of adopting it, expertise required, support tools for the usage and application of the methodology, etc.

INGENIAS [7] is a methodology that covers these four aspects, and some other characteristics that the literature about agent oriented methodologies proposes for these kinds of methodologies [10]. It provides a process to guide the software development; a language based on the main concepts of agent theory (for example the notions of agent, role, mental state, goals, believes, tasks, etc.); different models for describing different views of the system at different levels of abstraction, and a tool to help in the modelling phase. Moreover, INGENIAS is not oriented to a particular agent platform such as other agent based methodologies, e.g. [1, 5, 6].

This paper presents the design of a multi-agent knowledge management system using INGENIAS. The content of the paper is organized as follows: section two presents the INGENIAS methodology; then, in section three the domain area of the system is described; section four illustrates how INGENIAS was used to model the multi-agent system; after that, a scenario of use of the implemented prototype is presented in section five. Finally, conclusions are in section six.

2. The INGENIAS Methodology

INGENIAS is an agent based methodology which has evolved from an object oriented approach; it reuses work from MESSAGE/UML [2], a methodology which has based its notation in UML. INGENIAS provides a visual language for multi-agent systems definition, a process to guide the lifecycle of the software development based on the Rational Unified Software Development Process (RUP), a tool that provides an editor for modeling and a code generator which can be customized to generate code for different agents' platforms [8].

There are five main system views described from five meta-models in INGENIAS, these meta-models can be used to define five different models: 1) the *agent model*, describes single agents, their tasks, goals, roles and initial mental state; 2) the *interaction model*, describes how interactions among agents take place, the involved actors, and the goals and protocols of the interactions; 3) the *tasks and goals model*, describes relationships between tasks and goals, as well as their structures; 4) the *organizational model*, describes how the system components are grouped together, the tasks that are carried out in common, the goals they share, and the constrains that exist in the interaction between agents; and 5) the *environment model*, defines agent's perception in terms of existing elements of the system.

The development process followed by INGENIAS has the same six main phases of the RUP and these are: 1) the analysis-inception phase where *organizational models* are produced to define a preliminary view of the system architecture; 2) the analysis-elaboration phase where the organizational model is refined to identify goals and tasks of the agents, for this purpose *task-goal models* must be constructed; 3) the analysis-construction phase where the models generated in the other phases are refined to cover the use cases that the system must fulfill; 4) the design-inception phase in which a fast prototype of the system is developed to evaluate the models defined in the analysis phases, 5) then, in the design-elaboration phase more details are added to the models by defining *workflows* between agents in the organizational models, completing these workflows with agents' interactions with *interaction models*, and defining agent mental states with *agent models*; 6) finally, in the design-construction phase the use cases that deal with special situations, but do not involve changes in the systems architecture are considered for modeling.

3. The Domain of the System

The domain of our system is software maintenance. The characteristics required for the system were obtained from a study performed in two software maintenance groups [9]. The study showed that on many occasions, organizations have documents or people with the information or knowledge necessary to support or help the software maintainers to do their activities, but either the latter did not know that other documents or people could have provided useful information to help them to complete the assignment or the people with useful information did not know what the latter was working on.

From the analysis of the results of the case study, some scenarios emerged; these showed how an agent based system can help to support some of the problems that maintainers actually have. These scenarios were used to obtain the main requirements and characteristics for the multi-agent system [9].

The system has been designed to help maintainers searching and informing about knowledge sources that can help them while they perform their jobs. To do this, there have been identified the main roles played by the maintainers, the sources of information they consult, the kinds of knowledge they have or require to perform their jobs, and how all these are involved in the processes and tasks performed by the maintainers. In the following section, the design of the multi-agent system is presented.

4. Using INGENIAS to Design the Multi-Agent System

As we have mention before, INGENIAS proposes a process in which the analysis starts focusing in the main structure of the system, the main agents, the roles they play, and how they are organized. Then, the main goals and tasks of the agents are defined. The process continues with the identification of workflows and agents interactions, and goes on until use cases of special situations are modelled. In the development of the knowledge management system, we have focused on the system architecture, the agents' internal structure, their goals, tasks, mental states, and how they interact with other agents and their environment (such as the user, resources or other

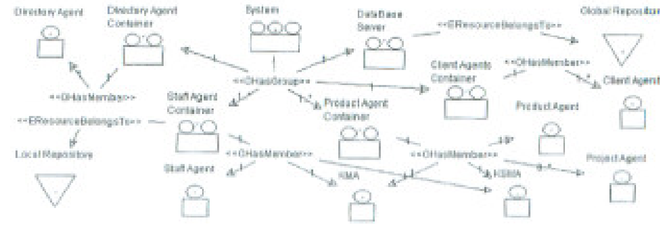


Figure 1. Organizational model that illustrates the system architecture.

applications). For space limitations, in this paper only the modelling of the system architecture, a scenario of agents' interactions and an example of an agent structure are presented.

4.1. Modelling the System Architecture

The architecture of the system is based on the main elements involved in the groups studied [9]. From the analysis of this information, five main agents were defined: *staff agents* which are the mediators between the staff members and the system; *product agents* which manage information related to the products maintained by the group, including their maintenance requests and the main elements that integrate the products (documentation, source code, databases, etc.); *project agents* which are in charge of the management of the modification projects (tasks, resources assigned, etc.); *client agents*, which bring support to the clients when they send a maintenance request or report a problem; and a *directory agent*, which manages information required by agents to know how to communicate with other agents that are active in the system. In addition, two auxiliary types of agents are also considered in the architecture, *Knowledge Manager Agents* (KMA), which are in charge of the management of the knowledge base, and *Knowledge Source Manager Agents* (KSMA), which are in charge of the management of information sources (such as electronic documents).

Figure 1 illustrates the organizational model that describes the system architecture. As can be seen, the agents are organized in four containers, directory, staff, project, and client agent container. This is because those containers can be each one in a different machine. For example, each staff member can have a staff agent in his/her local machine, and the client agents can be in a web server with an application for the management of modification requests. In the architecture there are also a data base server, with a global repository where the knowledge base will reside, and a local repository for each staff agent container, since each staff member can have documents and information, in its local machine that can be relevant to other members.

The organizational meta-model of INGENIAS provides the main elements for the definition of multi-agent systems' architectures [7]. However, there are some aspects that are not considered. Particularly, it is not clear how to model the physical distribution of the system's components, the authors of INGENIAS argue that this can be done with UML's deployment diagrams, but these are not addressed in the actual version of INGENIAS.

4.2. Modelling Workflows and Agents Interactions

The system design was based on scenarios identified from the case studies carried out. These scenarios describe some problems that maintainers have, and how agents can help to address these problems. In this section, the modelling of agents interactions are described using one of the scenarios identified which is explained as follows.

First, the maintainer looks at a list of the projects s/he has assigned in a project manager application. When the maintainer selects one project, an event is triggered and captured by the *staff agent*, which obtains the information of the project, identifies knowledge topics (system and

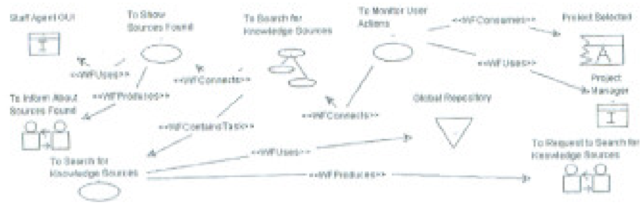


Figure 2. Workflow diagram that illustrates the main activities and interactions in the scenario.

module where the problem appeared, kind of problem, etc.) and generates some rules to request the KMA to search for knowledge sources. To create the rules, the *staff agent* tries to identify the knowledge that the engineer would need to do the assignment. Also the agent considers the types of sources the engineer consults, assigning more relevance to the sources that he consults most frequently. When the search has finished, the KMA sends a message to the *staff agent* informing it about the sources found. The *staff agent* displays a message with the number of knowledge sources found in order to inform the maintainer of their availability. Finally, if the engineer wants to look for the sources, s/he chooses a button in the *staff agent* screen, and the agent will display a window with the list of sources (see Figure 5). When the maintainer selects one source from the list, the system shows information related to that source such as: location, the knowledge that it has, etc.

Figure 2 shows an example of a workflow model of the scenario. In the model, the event "Project Selected" is captured in the "To Monitor User Activities", then, this starts the "To Search Knowledge Sources" workflow where the activities "To Search for Knowledge Sources" and "To Show Sources Found" take place. These activities generate the "To Request to Search for Knowledge Sources" and "To Inform about Sources Found" interactions respectively. The model does not consider some aspects of the scenario because this can make the model difficult to follow. Thus, there have been constructed different models for defining the details of the scenario.

It was difficult to model all the details we wanted in some of the models because these made the models too difficult to follow and understand, particularly in workflow models. However, we found that there were many ways for modeling these details in different views and levels of abstractions.

4.3. Modelling Agents' Tasks and Goals

A very important step in multi-agent systems design is the identification of the goals that agents will follow, and the tasks they must perform in order to complete those goals. These tasks and goals can be involved in different models of the INGENIAS methodology, for example organizational, workflow and interaction diagrams; and the details about the relations

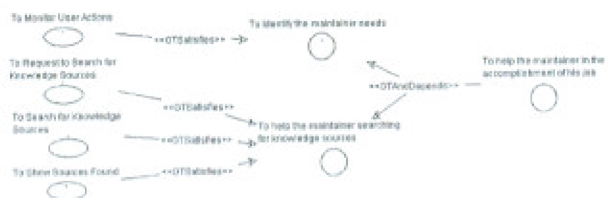


Figure 3. Diagram of the main tasks and goals of the scenario.

between goals and tasks are described in the goals and tasks models. Then, agent models can be used to define how the agent mental state is affected by the tasks performed by the agent.

Figure 3 shows the main goals and tasks identified from the scenario. As can be seen, the main goal is to help the maintainer in the accomplishment of his/her job. This goal is composed of other two; the first is to identify the maintainer needs, and the second to help him/her to search for knowledge sources that can help to solve those needs. In order to fulfil those goals, four main tasks are defined: to monitor the user activities, to request to search for knowledge sources, to search knowledge sources, and to show the sources found. In the diagram is not defined which agents perform those tasks; this was done in the agent models, like the one of Figure 4.

Figure 4 presents the agent model of the staff agent; as can be seen, it has two main goals: to identify the maintainer needs, and to help the maintainer searching for knowledge sources based on those needs. In order to do this, the staff agent performs three main tasks: to monitor the user activities, to request to search for knowledge sources and to show the sources found; these last two activities are done in collaboration with the KMA. On the other hand, the model also illustrates the main mental states that the staff agent has in the scenario described. The first mental state is activated when the agent receives the event that informs that the user has selected a project. In this state, the agent obtains information about the project. Then, the agent starts to generate the rules for the search, so it changes its mental state. In this second mental state, the agent has two believes, that the user might need some knowledge to accomplish his job, and that there are sources that can help the user to obtain that knowledge. Thus, the agent generates some rules that would be used to search for those sources and starts the "to request to search for knowledge sources" task. The third mental state starts when the staff agent receives from the KMA the message informing about the sources found. This causes that the staff agent initiates the "to show sources found" task. Finally, there are two important entities proposed by INGENIAS which are presented in the staff agent model, these are the *Mental State Manager*, and the *Mental State Processor* (inference engine in the model). The first one is in charge of managing the mental state and the knowledge of the agent. The second one is the decision making mechanisms. These entities enable to separate the autonomy and intelligence mechanisms of the agent from the agent conceptualization.

Even though INGENIAS provides some entities that can be used to define some aspects of the agents' knowledge and reasoning, it is not enough for some cases. There is not a knowledge model where the domain knowledge of the agent can be modeled, or where the problem solving strategies of the agents and how it chooses or uses them can be defined (for example like the knowledge modeling stage proposed by MAS-CommonKADS [3]). However, some of these aspects can be addressed using other approaches, for example UML class diagrams for the definition of ontologies of the domain knowledge of the agents.

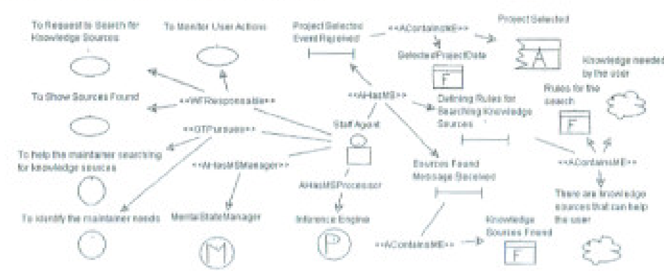


Figure 4. Diagram of the staff agent model.

5. Implementation of the System

To evaluate the feasibility of implementation of the architecture, we have developed a prototype which was tested following scenarios like the one described in section 4.2. Figure 5 illustrates how the prototype presents information about the knowledge sources found by the KMA.

In this example, the maintainer decides to solve a problem reported by a client. When the maintainer selects to view the problem data to start solving it, an event is triggered and captured by the staff agent. This agent obtains some information from the report (such as the system, module, type of error, etc.). Then, the staff agent request the KMA to search for knowledge sources that could help the maintainer to solve the problem, by sending a query generated with the information obtained from the report. When the search is finished, the KMA informs the staff agent about the sources found, and the staff agent informs to the maintainer the number of sources found that can be relevant to the activity. If the user wants to consult the sources, s/he presses a button in the staff agent screen, and a window similar to the one of Figure 5 is displayed. In this window, the sources are listed by type, and if the user selects one source, the system presents information about where the source is, and the kind of knowledge that the source has and that can be relevant to the activity.

The tests made to the prototype showed us that it can help maintainers to find and locate sources of information that might be relevant to the activities performed by maintainers. Thus, sources that are not consulted for ignorance of their existence or location could now be consulted thanks to the automatic search done by the agents. Therefore, the replication of mistakes or the re-invention of a solution for a problem that has already been solved can be avoided.

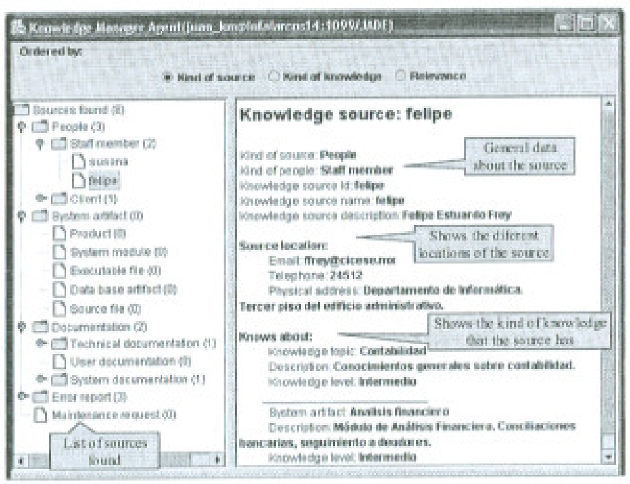


Figure 5. Screen shot of list of knowledge sources found.

6. Conclusions and Future Work

The INGENIAS methodology provides characteristics that were very useful in the analysis and design of the multi-agent system. For example, the organizational model helped us to define the main elements of the system architecture; workflow models helped us to conceptualize the main activities involved in some scenarios used to define the requirements that the system must fulfil; goal-task models were very useful to identify the relations of goals and tasks; and agents models helped us to identify the internal aspects of the agents, such as their main goals and mental states. However, some aspects could not be addressed directly with the elements proposed by the methodology. For example the physical distribution of the architecture's components, the domain knowledge of the agents, or the problem solving strategies of them. Nevertheless, INGENIAS satisfied the majority of the basic elements for agent-based systems that our system required.

After the study of the literature about different agent-oriented methodologies, we think that INGENIAS is one of the most complete methodologies for the development of multi-agent systems, since it proposes a very descriptive language, an easy to follow process, models that can be constructed with different levels of abstractions and from different views or perspectives, and a tool based on meta-models that facilitates the modelling of the multi-agent system which is not oriented to a particular agent platform.

Currently, we have developed a prototype of the multi-agent system. In order to address the missing aspects of INGENIAS for the development of the complete system we continue studying other methodologies. Although, we hope that INGENIAS could be complemented with other approaches that help us in the development of the final version of the system.

Acknowledgements

This work is partially supported by CONACYT under grant C01-40799 and the scholarship 164739 provided to the first author, the MAS project (grant number TIC2003-02737-C02-02), Ministerio de Ciencia y Tecnología, SPAIN, and the red agentcities, number TIN2004-20447-E.

Referencias

[1]. P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia and J. Mylopoulos, "A knowledge level software engineering methodology for agent oriented programming", in Proc. of the 5th Intl. Conf. on Autonomous agents, Montreal, Canada, ACM Press, (2001), p. 648-655.

[2]. G. Caire, W. Coulier, F. Garijo, J. Gómez, J. Pavón, F. Leal, P. Chainho, P. Kearney, J. Stark, R. Evans and P. Massonet, "Agent Oriented Analysis using MESSAGE/UML", in AOSE, (2001), p. 119-135.

[3]. C. A. Iglesias, M. Garijo, J. Centeno-González and J. R. Velasco, "Analysis and Design of Multiagent Systems Using MAS-Common KADS", LNCS 1365, Springer, (1998), p. 313-327.

[4]. C. A. Iglesias, M. Garijo and J. C. González, "A Survey of Agent-Oriented Methodologies", LNCS 1555, Springer, (1998), p. 317-330.

[5]. H. S. Nwana, D. T. Ndumu, L. C. Lee and J. C. Collis, "ZEUS: A Toolkit for Building Distributed Multiagent Systems", Applied Artificial Intelligence, 13(1-2), (1999), p. 129-185.

[6]. L. Padgham and M. Winikoff, "Prometheus: A Methodology for Developing Intelligent Agents", in Proceedings of the 3rd Intl. AOSE, at AAMAS 2002, Bologna, Italy, (2002), p. 174-185.

[7]. J. Pavón and J. J. Gómez-Sanz, "Agent Oriented Software Engineering with INGENIAS", LNCS 2691, Springer, (2003), p. 394-403.

[8]. J. Pavón, J. J. Gómez-Sanz and R. Fuentes, "INGENIAS web site", (2004), http://grasia.fdi.ucm.es/ingenias/, consulted at 8-May-2004.

[9]. O. M. Rodríguez, A. I. Martínez, J. Favela, A. Vizcaíno, M. Piattini, "Understanding and Supporting Knowledge Flows in a Community of Software Developers", LNCS 3198, Springer, (2004), p. 52-66.

[10]. A. Sturm and O. Shehory, "A Framework for Evaluating Agent-Oriented Methodologies", in 5th Intl. Bi-Conf. AOIS-2003 at AAMAS'03, Melbourne, Australia, (2003), p. 95-110.