A MULTI-AGENT SYSTEM FOR INFORMATION SHARING

Marco Mari, Agostino Poggi, Michele Tomaiuolo
DII, University of Parma, Parco Area delle Scienze 181/A, Parma, Italy

Keywords: Information sharing, multi-agent systems, peer-to-peer.

Abstract: This paper presents RAIS, a peer-to-peer multi-agent system supporting the sharing of information among a community of users connected through the internet. RAIS has been designed and implemented on the top of well known technologies and software tools for realizing multi-agent and peer-to-peer systems, for the searching of information and for the authentication and authorization of users. RAIS offers a similar search power of Web search engines, but avoids the burden of publishing the information on the Web and guaranties a controlled and dynamic access to the information. Moreover, the use of agent technologies simplifies the realization of three of the main features of the system: i) the filtering of the information coming from different users on the basis of the previous experience of the local user, ii) the pushing of the new information that can be of possible interest for a user, and iii) the delegation of access capabilities on the basis of a network of reputation built by the agents of the system on the community of its users.

1 INTRODUCTION

The storage capability of hard disks is constantly growing, while the new available space is quickly filled with a large amount of data in different and heterogeneous formats. The ordering of such data is a time wasting and often boring task: the result is that more and more files, containing important information, are lost or forgotten on hard drives. The classical file searching tools (e.g., Windows search function) are not effective: for each request they analyse the whole drive, and they support only few file formats. In last months, the providers of the main Web search engines (Google, Microsoft with MSN and Yahoo!) have released desktop search tools that make a research on a local drive as easy and fast as a Web search. These tools run in the background (when CPU load is low) and index the content of a wide range of file formats (e.g.: Office, PDF, e-mail, HTML, …) in a way similar to a Web crawler. Moreover, Google has released an SDK for its desktop search software. The SDK empowers developers to write plug-ins thanks to a set of APIs using COM and HTML/XML.

Besides a local drive, the better place to find files and, more in general, information is the Internet. The most used tools to share contents and information avoiding the burden of exporting them on the Web are peer-to-peer systems. The exchange of multimedia files with a peer-to-peer system is highly effective because their contents can be easily categorized by the title. On the other hand, files holding more information (e.g., documents, e-mails, etc.) could be effectively shared only if the categorization takes into account the whole content.

This paper presents a system, called RAIS, that tries to couple the features of peer-to-peer information sharing systems and the Web. The next section introduces the main features and the behaviour of the RAIS system. Section three describes how this system has been designed and implemented by using some well-known technologies and software tools. Finally, section four gives some concluding remarks and presents our future research directions.

2 RAIS

RAIS (Remote Assistant for Information Sharing) is a peer-to-peer and multi-agent system composed of different agent platforms connected through the internet. Each agent platform acts as a “peer” of the system and is based on three agents: a personal assistant, an information finder and a directory facilitator; moreover, another agent, called personal proxy assistant, allows a user to remotely access

Mari M., Poggi A. and Tomaiuolo M. (2006).
A MULTI-AGENT SYSTEM FOR INFORMATION SHARING.
In *Proceedings of the Eighth International Conference on Enterprise Information Systems - SAIC*, pages 147-152
DOI: 10.5220/0002446501470152

her/his agent platform. Figure 1 shows the RAIS multi-agent system architecture.

A personal assistant (PA) is an agent that allows the interaction between the RAIS system and the user. This agent receives the user's queries, forwards them to the available information finders and presents the results to the user. Moreover, a PA allows the user to subscribe her/him to be notified about new documents and information on some topics in which she/he is interested. Finally, a PA maintains a profile of its user preferences: in fact, the user can rate the quality of the information coming from another user for each search keyword (the utility of this profile will be clear after the presentation of the system behaviour).

An information finder (IF) is an agent that searches information on the repository contained into the computer where it lives and provides this information both to its user and to other users of the RAIS system. An IF receives users' queries, finds appropriate results and filter them on the basis of its user's policies (e.g.: results from non-public folders are not sent to other users). Moreover, an IF monitors the changes in the local repository and pushes the new information to a PA when such information matches the interests subscribed by this PA.

A personal proxy assistant (PPA) is an agent that represents a point of access to the system for users that are not working on their own personal computer. A PPA is intended to run on a pluggable device (e.g., a USB key) on which the PPA agent is stored together with the RAIS binary and configuration files. Therefore, when the user starts the RAIS system from the pluggable device, her/his PPA connects to the user's PA and provides the user with all the functionalities of her/his PA. For security reasons, only a PA can create the corresponding PPA and can generate the authentication key that is shared with the PPA to support their communication. Therefore, for a successful connection, the PPA has to send the authentication key, and then the user must provide his username and password.

Finally, the Directory Facilitator is responsible to register the agent platform in the RAIS network. The DF is also responsible to inform the agents of its platform about the address of the agents that live in

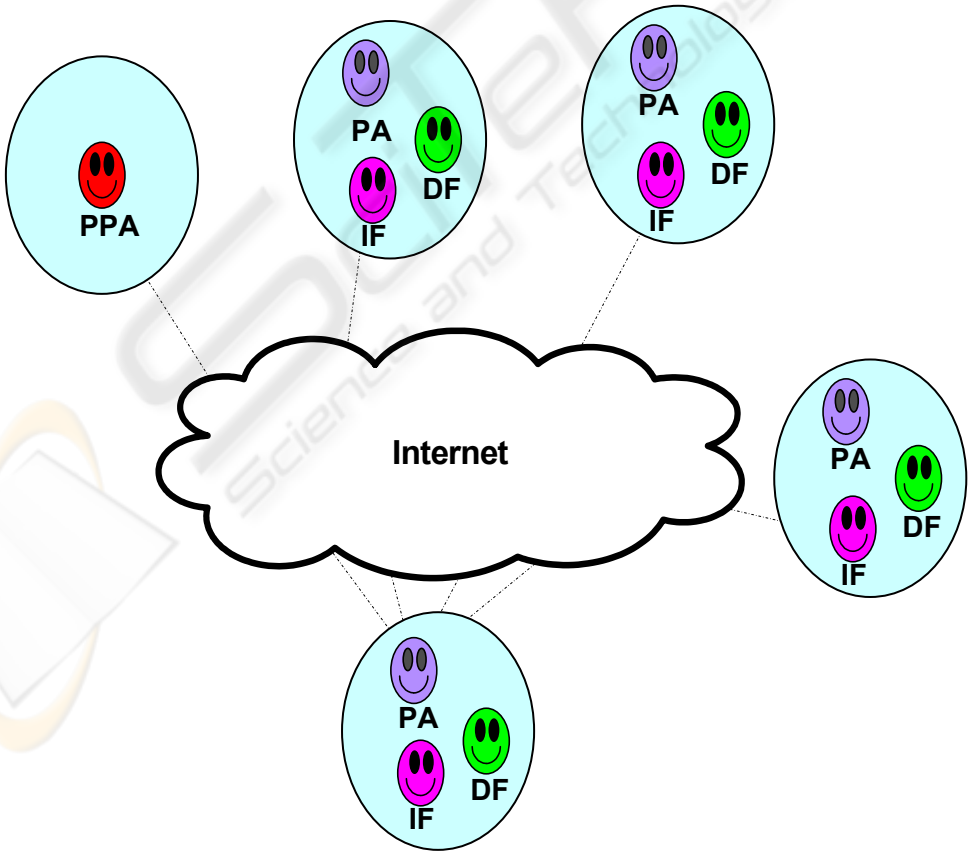

Figure 1: The RAIS multi-agent system architecture.

the other platforms available on the RAIS network (e.g., a PA can ask about the address of the active IF agents).

2.1 Searching and Pushing Information

In order to understand the system behaviour, we can present two practical scenarios. In the first, a user asks her/his PA to search for some information, while in the second the user asks to subscribe her/his interest about a topic. In both cases the system provides the user with a set of related information.

In the first scenario, the system activity can be divided in four steps:

Search: the user requests a search to her/his PA indicating a set of keywords and the maximum number of results. The PA asks the DF for the addresses of available IF agents and sends the keywords to such agents. The information finders apply the search to their repositories only if the querying user has the access to at least a part of the information stored into its repositories.

Results filtering: each IF filters the searching results on the basis of the querying user access permissions.

Results sending and presentation: each IF sends the filtered list of results to the querying PA. The PA orders the various results as soon as it receives them, omitting duplicate results and presents them to its user.

Retrieval: after the examination of the results list, the user can ask her/his PA for retrieving the information corresponding to an element of the list. Therefore, the PA forwards the request to the appropriate IF, waits for its answer and presents the information to the user.

In the second scenario, the system activity can be divided in five steps:

Subscription: the user requests a subscription to her/his PA indicating a set of keywords describing the topic in which she/he is interested. The PA asks the DF for the addresses of available IF agents and sends the keywords to such agents. Each IF registers the subscription if the querying user has the access to at least a part of the information stored into its repository.

Monitoring and result filtering: each IF periodically checks if there are some new information satisfying its subscriptions. If it happens, the IF filters its searching results on the basis of the access permissions of the querying user.

Results sending and user notification: each IF sends the filtered list of results to the querying PA. The PA orders the various results as soon as it receives them, omitting duplicate results and stores them in its memory. Moreover, it notifies its user about the new available information sending her/him an email.

Results presentation: the first time the user logs into the RAIS system, the PA presents her/him the new results.

Retrieval: in the same way of the previous search scenario, the user can retrieve some of the information indicated in the list of the results.

As introduced above, a PA receives from the user a constraint on the number of results to provide (N_r) and uses it to limit the results asked to each IF agent. The number of results that each IF agent can send is neither N_r nor N_r divided to the number of IF agents (N_r/N_{if}), but a number (between N_r and N_r/N_{if}) for which the PA is quite sure to provide at least N_r results to its user without the risk of receiving a burden of unnecessary data. Moreover, each IF, before sending the list of results, creates a digest of each result and sends them together with the list. Therefore, the PA can use the digests to omit duplicate results coming from different IF agents.

After the reception of results and the filtering of duplications, the PA has the duty of selecting N_r results to send to its user (if they are more than N_r) and order them. Of course, each IF orders the results before sending them to the PA, but the PA has not the information on how to order results from different IF agents. Therefore, the PA uses two more simpler solution on the basis of its user request: i) the results are fairly divided among the different sources of information, ii) the results are divided among the different sources of information on the basis of the user preferences. User preferences are represented by triples of the form <source, keyword, rate> where: source indicates an IF, keyword a term used for searching information, and rate a number representing the quality of information (related to the keyword) coming from that IF. Each time a user gets a result, she/he can give a rate to the quality of the result and, as consequence, the PA can update her/his preferences in the user profile that the PA maintains.

The information stored into the different repositories of a RAIS network is not accessible to all the users of the system in the same way. In fact, it's important to avoid the access to private documents and personal files, but also to files reserved to a restricted group of users (e.g.: the participants of a project). The RAIS system takes care of users' privacy allowing the access to the information on the basis of the identity, the roles and

the attributes of the querying user defined into a local knowledge base of trusted users. In this case, it is the user that defines who and in which way can access to her/his information, but the user can also allow the access to unknown users enabling a certificate based delegation built on a reputation network of the users registered into the RAIS community. For instance, if the user U_i enables the delegation and grants to the user U_j the access to its repository with capabilities C_0 and U_j grants to the user U_k the access to its repository with the same capabilities C_0, then U_k can access U_i's repository with the same capabilities of U_j. The trust delegation can be useful when the system is used by open and distributed communities, e.g. to share documents among the members of an Open Source project.

2.2 Mobile Users Support

People travelling for work may often be in need of access from a remote system their own computer. In this situation, a solution could be to install a VNC server on the desktop computer and to find a system with a VNC client while travelling. This solution has the advantage that the user gains the complete control on his remote PC, but it has also two main drawbacks: it's not easy to find computers with VNC clients available and the VNC connects only to one computer, not to the whole set of files and information of a workgroup.

For users that don't require a complete control over a remote computer, but need to search and access a distributed set of documents, we have included in our system a remote search feature. The user can ask his/her PA to create a PPA on a pluggable device, e.g., an USB key or a removable hard disk. The PA copies on the device the RAIS run-time, the PPA and the authentication key shared by the PPA and the PA itself. When the user inserts the pluggable device on another computer, he can immediately launch his PPA and connect to its corresponding PA.

Therefore, the way of using the RAIS system is analogous to the situation in which the user works on her/his own computer, except for the interactions between the PPA and the PA, that, however, are transparent to the user. In fact, at the initialization, the PPA sends an authentication key to the PA. If the key matches those of the PA, the user can provide his/her username and password and enter the system (step that must be done by the user when she/he uses the RAIS system from her/his own computer too). After these two steps, the PPA acts as a simple proxy of the remote PA.

3 RAIS DEVELOPMENT COMPONENTS

The RAIS system has been designed and implemented taking advantage of agent, peer-to-peer, information retrieval and security management technologies and, in particular, of three main software components: JADE, JXTA (Gong, 2001) and Google Desktop Search.

RAIS agent platforms have been realized by using JADE: JADE (Java Agent Development Framework) is probably the most known agent development environment enabling the integration of agents and both knowledge and Internet-oriented technologies. Currently, JADE is considered the reference implementation of the FIPA (Foundation for Intelligent Physical Agents) specifications. In fact, it is available under an LPGL open source license, it has a large user group, involving more than two thousands active members, it has been used to realize real systems in different application sectors, and its development is guided by a governing board involving some important industrial companies.

The JADE development environment does not provide any support for the realization of real peer-to-peer systems because it only provides the possibility of federating different platforms through a hierarchical organization of the platform directory facilitators on the basis of a priori knowledge of the agent platforms addresses. Therefore, we extended the JADE directory facilitator to realize real peer-to-peer agent platforms networks thanks to the JXTA technology and thanks to two preliminary FIPA specifications for the Agent Discovery Service and for the JXTA Discovery Middleware.

JXTA technology (Gong, 2001) is a set of open, general-purpose protocols that allow any connected device on the network (from cell phones to laptops and servers) to communicate and collaborate in a peer-to-peer fashion. The project was originally started by Sun Microsystems, but its development was kept open from the very beginning. JXTA comprises six protocols allowing the discovery, organization, monitoring and communication between peers. These protocols are all implemented on the basis of an underlying messaging layer, which binds the JXTA protocols to different network transports.

FIPA has acknowledged the growing importance of the JXTA protocols, and it has released some specifications for the interoperability of FIPA platforms connected to peer-to-peer networks. In particular, in the "FIPA JXTA Discovery

Middleware Specification" a Generic Discovery Service (GDS) is described, to discover agents and services deployed on FIPA platforms working together in a peer-to-peer network. RAIS integrates a JXTA-based Agent Discovery Service (ADS), which has been developed in the respect of relevant FIPA specifications to implement a GDS. This way, each RAIS platform connects to the Agent Peer Group, as well as to other system-specific peer groups. The Generic Discovery Protocol is finally used to advertise and discover df-agent-descriptions, wrapped in Generic Discovery Advertisements, in order to implement a DF service, which in the background is spanned over a whole peer group.

Different techniques and software tools can be used for searching information in a local repository. If the information is stored in form of files, the Google desktop search system can be considered a suitable solution because Google provides an SDK for developing plug-ins based on its desktop search system. The API that comes with the SDK uses COM objects, so it's not directly available for JAVA development, but a bridge between the API and JAVA is provided by the Open Source project GDAPI. Google desktop search indexes the content of the files of a local drive in a way similar to the Google Web crawler, providing a searching engine fast, effective and with the support for a wide range of file formats.

As introduces before, authentication and authorization are performed on the basis of the local knowledge base of trusted users, though they can be delegated to external entities through an explicit, certificate based, delegation. In this sense, the system completely adheres the principles of trust management. The definition of roles and attributes is also made in a local namespace, and the whole system is, in this regard, completely distributed. Local names are distinguished by prefixing them with the principal defining them, i.e., an hash of the public key associated with the local agent platform. Links among different local namespace, again, can be explicitly defined by issuing appropriate certificates. The theory of RAIS delegation certificates is founded on SPKI/SDSI specifications (Ellison et al., 1999), though the certificate encoding is different. As in SPKI, principals are identified by their public keys, or by a cryptographic hash of their public keys. Instead of s-expressions, RAIS uses XML signed documents, in the form of SAML assertions, to convey identity, role and property assignments. As in SPKI, delegation is possible if the delegating principal issues a certificate whose subject is a name defined by another, trusted, principal. The latter can successively issue other certificates to assign other principals (public keys) to its local name. In this sense, local names act as distributed roles (Li et al., 2003).

Finally, the extraction of a digest for each search result is required to avoid the presentation of duplicate results to the user. This feature is provided by a Java implementation of the hash function MD5 (Rivest, 1992).

4 CONCLUSIONS

In this paper, we presented a peer-to-peer multi-agent system, called RAIS (Remote Assistant for Information Sharing), supporting the sharing of information among a community of users connected through the internet.

RAIS is implemented on the top of well known technologies and software tools for realizing: i) the agent platforms, i.e., JADE, ii) the peer-to-peer infrastructure, i.e., JXTA, iii) the searching of information into the local repository, i.e., Google Desktop Search, and iv) the authentication and authorization infrastructure, i.e., SPKI/SDSI specifications and the SAML assertions. Therefore, RAIS can be considered something more than a research prototype that couples the features of Web searching engines and of peer-to-peer systems for the sharing of information. First, RAIS improves the security rules provided to check the access of the users to the information. In addiction, it offers a similar search power of Web search engines, but avoids the burden of publishing the information on the Web and guaranties a controlled and dynamic access to the information. Moreover, the agent based implementation simplifies the realization of three main features of the system: i) the filtering of the information coming from different users on the basis of the previous experience of the local user, ii) the pushing of the new information that can be of possible interest for a user, and iii) the delegation of authorization on the basis of a network of reputation built by the agents of the system on the community of its users.

A first prototype of the RAIS system has already been completed and experimented. The prototype includes all basic features and a graphical user interface simplifies the interaction between the user and the system (see figure 2).

Practical tests on the first prototype were done installing the system in different labs and offices of our department asking some students and colleagues to use it for sharing and exchanging information.

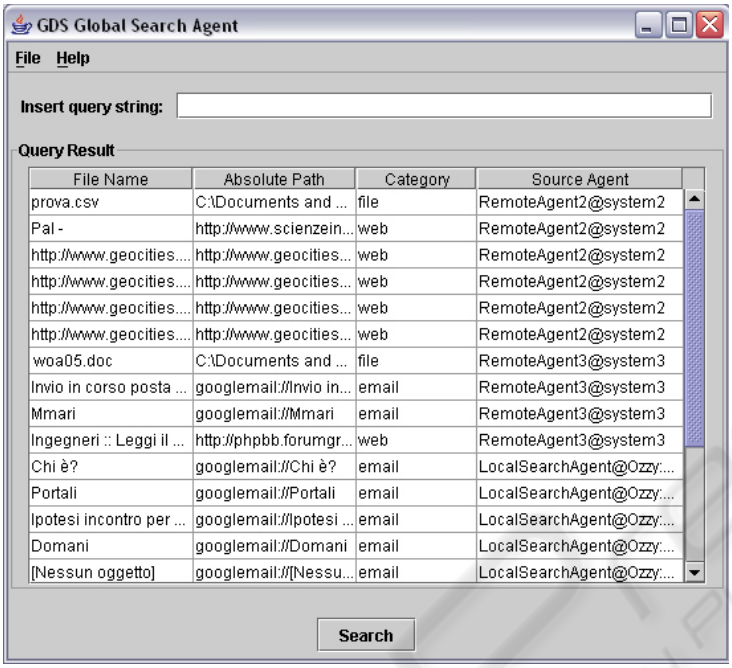

Figure 2: RAIS search graphical user interface.

Moreover, we tested the system setting some computers of a Lab with different access policies and distributing information on their repositories to have different copies of the same information on different computers. The tests covered with success all system features: basic research, user defined policies, results filtering, duplicate results management and remote search through a proxy personal agent. The tests results were fully satisfactory and, in particular, the involved users were interested in continuing its use for supporting their work activities. The successful experimentation encouraged us in the further development of the system and we are currently working on studying the best way for introducing new types of information that can be managed (e.g., information stored into databases) and for including new techniques for the searching of such information (e.g., semantic Web techniques).

REFERENCES

Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T. SPKI Certificate Theory. RFC 2693, 1999.

FIPA Specification. (n.d.). Retrieved January 11, 2006, from http://www.fipa.org.

FIPA Agent Discovery Service Specification. (n.d.). Retrieved January 11, 2006, from http://www.fipa.org/specs/fipa00095/PC00095.pdf

FIPA JXTA Discovery Middleware Specification. (n.d.). Retrieved January 11, 2006, from www.fipa.org/specs/fipa00096/PC00096A.pdf.

Gong, L. JXTA: A network programming environment. In *IEEE Internet Computing*, 5:88-95, 2001.

Google Desktop SDK. (n.d.). Retrieved January 11, 2006, from http://desktop.google.com/developer.html.

Google Desktop Search. (n.d.). Retrieved January 11, 2006, from http://desktop.google.com.

GDAPI, Google Desktop Search Java API. (n.d.). Retrieved January 11, 2006, from http://gdapi.sourceforge.net.

JADE software development framework. (n.d.). Retrieved January 11, 2006, from http://jade.tilab.com.

JXTA technology. (n.d.). Retrieved January 11, 2006, from http://www.jxta.org.

Li, N., Mitchell, J.M.. RT. A Role-based Trust-management Framework. *In Proc of the Third DARPA Information Survivability Conference and Exposition (DISCEX III)*, pp. 201-212, 2003. Washington, D.C.

Rivest, R.L. The MD5 Message Digest Algorithm. Internet RFC 1321. 1992.

SAML - Security Assertion Markup Language. (n.d.). Retrieved January 11, 2006, from http://xml.coverpages.org/saml.html.