Mobile Agents: An alternative to information exchange systems in public government

Abstract

Among the different paradigms suggested to increase effectiveness in the access to external systems we ought to consider mobile agents. Their self-management allows them to perceive the environment and select the best course of actions to take in order to complete their work independently and efficiently.

The use of mobile agents by governmental agencies poses some problems that ough to be solved before implementating any system, particularly when the exchange of information with citizens must be addresed effectively.

The exchange of Information between public administrations is characterized by high levels of securitywhich must be achieved due to the legal requirements set out in the legislation in the different sites where it is applied.

There are numerous architectures to be defined based on the abstraction of tasks and the inclusion of both local and external layers, which allow access to remote services. So ESB, SOA and others focus their attention on services that are already being used, incorporating a greater degree of abstraction.

Genesis aims to experiment with the issue of information exchange management-management and government agencies–citizens, allowing the optimisation of resources as well as creating a fully-functional infrastructure of free and open services to optain solutions for the problems of multiple platforms.

Keywords: Mobile Agents, Public Administration, data interchange

Where is the study

The term "services" is taking much meaning for a large users number. The trend is based on the current or expected capacity of communications. In order to make real systems based on remote assistance we must be able to abstract these services at the highest level and use them across different platforms, regardless of the technology the business users are willing to target their developments.

In several countries, decentralisation of competentes is a fact that must be taken into account in the provision of citizen services. Both federated models and regional autonomous agencies are a fact which cannot be overlooked by political trends. Dispersal to national, supranational, local and/or public companies shared the responsibilities of a more or less efficient in view the ultimate goal of providing service to citizens. Bearing in mind that different administrations have powers, which sometimes can not be transferred to other entities. In general, these powers pose problems in the area in question, the information exchange. We should also note that these competence "islands" manage their own data. The possibility of exchanging information is taken into account by the need to articulate transfer mechanisms, reliable and safe, to render services to other government and private citizens.

If we consider that the information transfer should aim to 24x7 availability, this can be a big financial outlay or the mechanisms for information exchange that inherently satisfy this requirement ought to be altered. An efficient utilization of network resources is a non trivial issue. Due to data distributed nature this question leads us to design efficient, fault-tolerant and decentralized.

Among the main benefits to be achieved with the mobile agent it can be highlighted the collection of information in a distributed and parallel processing tasks of personal assistants [24], reasons why the services of an agent are desirable in dealing with public government.

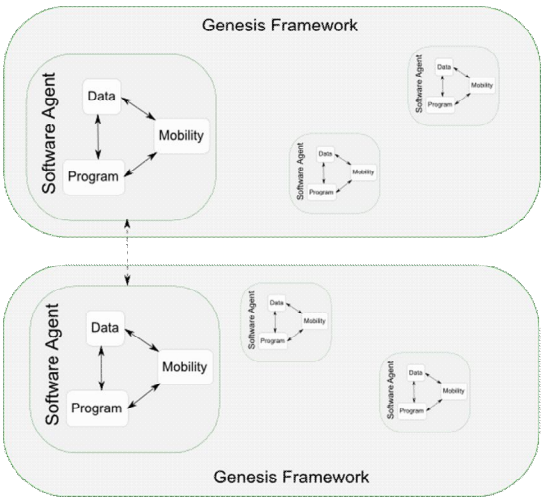
The mobile agent technologies have been evolving and adapting to the technological environment at the time. The following items describe technologies that are being used, or may be used for mobile agents implementation.

Sometimes mobile agent technologies are characterized by physical systems that move from one location to another leaving the term mobility agent technology called transportable agents. These pages use the name more mobile agents, taking into account that mobility is inherent to the device agent not running it, although it is possible that this device can be moved physically. Similarly, it is common to differentiate between "moving target" when you want to emphasize mobility as opposed to the term "Mobile Agent" that seems more oriented to sofware cognitive abilities. In our case this separation will not be made.

Updating the first models

In the early days of modern computing, we learned to differentiate data and programs to perform more tasks more efficiently. Currently, other challenges arise mainly in the field of information exchange due to the possibilities offered by communication networks. We can not stand still in the mere separation between programs and data; we must learn to discern between that programming needed for the treatment of data and those software programs aimed at the mobility agent and that allows users to add distributed tools attributes.

Bearing the above in mind, we should organize into three parts that can store information: data, programs, and eventually data and information relating to mobility (Data, Program and Mobility). Not stressing the importance of mobility, or any of its two parts, may lead us to the production of over-complex sotware.



A little field work

The size and the characteristics of the files to be transfered must be defined in the first place. We found several studies like [33] that analyze the distribution of files on UNIX systems, in order to set the size of the block in the file system. Many of these files will never be transferred by a data transmission system. Therefore, we must focus as in [9] that carries out an analysis of the files transfered on a specific computer department. Nonetheless it is better to conduct a study of the transfer requirements which may determine a draft system and also enable us to set out the rest of hypotheses.

The files exchanged in the production systems of the department under analyzis can be obtained by numerous methods and sources, ranking from the current batch systems to the innovative online systems for the exchange of XML files. Their goal is (always) the same, processing by internal software to the organization. Make a scale of these E / S files might be difficult due to its variability both in term of size and time. During the anályisis average values of files of about 350MB were obtained, and 4GB sizes up information stored in files with an annual time period. These files have their origin and / or destination in distant locations using WAN connections. Its composition is provided in text and have not been taken into account binary files in other environments may be more common. These files are composed of position-delimited plain text and its codification in other formats may involve the increase in size by a factor of three [20].

The data above has been obtained from a real production system, but this does not mean that other systems with similar characteristics have similar values in both size and file formats, as work environments used by governments and citizens can differ largely on the environment where the study was carried out.

In another development, the implementation of Java has resulted in an increased diversity of installed systems, but with the variety of environments and different versions of libraries and versions, this proposal is a complicated. The performance problems that have interpreted programming languages, especially in production environments and high productivity, are inevitable. For all this, she supports an environment not dependent on programming environments and a proposal is made for performance, not implementation.

What is a mobile agent?

A definition of mobile agents can be provided in respect for what they are capable of. An agent can be defined as "software that is determined by the relationships you have with your environment and the tasks it has done." In summary, an agent should not be defined for what is but for what it is capable of producing.

The mobile agent is a software product capable of performing actions to achieve improvements in the performance of themselves or the process that has as its aim. Its distinguishing feature, the mobility should be developed through a cycle that has been defined by some writers as in [28] and which will be updated later.

- Stop the agent
- Identify the transfer agent status.
- Serialize the typology of agent and its state.
- Categorise it by a transport protocol.
- To provide authentication information to the server.
- Transfer the agent.

Upon receipt of the agent

- Authenticating agent
- De-coding the agent.
- De-serialize agent class and status.
- Instantiate the agent.
- Restore the state of the agent.
- Resume agent execution.

Standards for mobile agent architecture

The market of agent standards is based on two proposals by companies already known in other areas of computing.

FIPA (Foundation for Intelligent Physical Agents) [11] is a non-for-profit organization created in 1996 to promote standardisation to facilitate the exchange of agents. It was officially accepted by the IEEE in 2005 and later became IEEE FIPA Standards Committee.

It has been defined in an abstract way by all the components that should have an architecture based on agents, but the most notorious one has been the Agent Communication Language (ACL). One of the main parts of this standard is its ontology, describing the communication language to be used by two agents in their trading.

The migration of agents is not treated in depth due to the low probability of success of migration between heterogeneous systems [15].

MASIF (Mobile Agent System Interoperability Facility) is the standard specification of OMG (Object Management Group) for the exchange of information among mobile agents.—The first proposal for standardization for mobile agents was performed (1997) and specifies two components: MAFAgentSystem defining standard operations management (creation, receipt, termination of the agent, etc ...) and MAFFinder that defines operations for agent registration and location. [28]

In the MASIF context, a mobile agent acts as a CORBA object or object ORG (Object Request Broker) communicating through interfaces IDL (Interface Definition Language) or by internal calls to the API ORG.

This standard emphasizes that in order to achieve interoperability within the environment, Masif agents must use the same development language.

Implementations of mobile agents

From the late 1990s there have been several proposals for implementing mobile agent-based architectures. Among all the existing proposals (D'Agents, Aglets [1], Grasshopper, Concordia, Mole, Voyager, Odyssey, Jumping Beans, Swarm, JAMES, etc ...) can highlight the following because of its size, importance and relevance of standards:

Aglets [1] was developed in the early 1997 by IBM and is maintained by the Open Source community since 2001. It

can be considered as the most prominent development system for architecture Masif. If we are to select the platform which has contributed the most to the development of mobile agents that would Aglets.

It was fully developed in JAVA that provides a highly portable to both agents and between platforms. It consists of two parts: a server Aglets (Tahiti) and a library for the development of new mobile agents.

JADE (Java Agent Development Framework) [4] is a platform for developing and supporting intelligent agents for multi-agent systems ensuring compatibility with FIPA standards. It became in 2000 Open Source when it was distributed by Telecom Italia under LGPL license.

This architecture offers two services in particular: AMS (Agent Management System) and DF (Directory Facilitator) and is completely developed in JAVA.

The JADE middleware is based on agents sharing the Java serialization API [5].

Close technologies

The following technologies have been used in the field of information exchange and must be taken into account when making proposals for the exchange of data.

XML (eXtensible Markup Language) [6] [37] is a metalanguage developed by the W3C (World Wide Web Consortium) aiming to represent structured information for both books and documents, transactions or agents. It is based on SGML (Standard Generalized Markup Language) Standard ISO 8879.

This metalanguage is used as the basis for the implementation of other standards such as XHTML (combination of XML and HTML). One of its major features is the selft-validation against a predefined frame. There have been numberous proposals to carry out this validation of which it would be worth highlighting two DTD's (Document Type Definition) and W3C XML Schema [34].

It has a large number of associated technologies to facilitate its use in many environments. XSL, XSL-FO, XLT, XPointer, XQuery, XPath, DOM or SAX are some of them. But we have also seen the renovation of existing protocols that have been transformed to acquire this new technology like XML-RPC or the creation of new for other needs, SOAP.

XML must be considered when discussing the present exchange of information between heterogeneous platforms. XML is simple and independent of the platform to use. Besides, most programming languages contain tools such as parsers or libraries to handle this kind of structured information.

P2P (Peer-to-peer). One of the distributed alternative which better supports the different problems that can arise in distributed environments are P2P networks. Its usefulness is based on all systems connected to the system can perform both functions as a client and server, hence its name. With the advent of this type of network the name of the nodes and the search for them becomes more complex as the hierarchical model of the Internet is not as efficient in these cases. Peer-to-peer is not a new technology but an existing one adapted to the needs of information exchange today.

We can highlight some of the possible categorisations relate to the proposal made later. The easiest way would be to place them in three categories: centralized, mixed or pure, depending on their degree of dependence on a central node. Perhaps it would be more convenient to treat them by their level of structuring. In such cases they can be structured or unstructured [26].

P2p platforms have been implemented in numerous programming languages as their protocols, in some cases opened, have become different and more optimized designs.

JXTA (Juxtapose) is Open Source platform created by Sun Microsystems as a set of XML protocols. These protocols allow the creation of peer-to-peer networks.

HTTP (Hypertext Transfer Protocol) is the result of the collaboration between the World Wide Web Consortium and the Internet Engeneering Task Force culminating in the publication of RFC 2616 [10] and the latest version 1.2 (RFC 2776) in 1999 [29]. It is a stateless protocol that can be used by different systems for transferring information.

The ability of having already configured all internet routing systems for the transfer of Web pages, provides this protocol a way to get to places where others have restricted access. We can find references to the use of HTTP as the transport protocol for the exchange of mobile agents [25].

XMPP (Extensible Messaging and Presence Protocol) is presented as a set of open protocols for the exchange of presence information and other messages between two entities on the network. The standard defines three types of messages [15] [16] [17] [18]:

> <message/>: used to exchange messages from one entity to another.
> <presence/>: to exchange information on availability.
> <iq/>: implements the request-response mechanism between entities.

Some proposals for implementation-Jabber XMPP protocols have already been made for the transport of agents such as [8] or [13].

Close Architectures

JEE (Java Enterprise Edition) [32] formerly known as J2EE (Java 2 Enterprise Edition) to version 1.4, is a platform for developing business applications with an architecture of N levels. Initially it was developed by Sun Microsystems which later handed over control to JCP (Java Community Process) [JEE3]

There are several high quality tools for developing applications on this platform. Some of them are open source which has facilitated their development and usage. The number of application servers, some of which are also open source, are easy to operate using different settings

JEE represents a specification conglomerate with a well-known API that provides a powerful development environment for enterprise applications. Thus, there is a wide range of companies that have migrated applications to this technology. Its use makes the work especially in Web-related developments.

.NET. This platform can be defined as an environment where different components interact, regardless of language, as far as they are written in. NET. This ability is due mostly to the Common Language Runtime (CLR) which has met the main features of C, C + + and Visual Basic. Just as JAVA,. NET allows the compilation in an intermediate code MSIL (Microsoft Intermediate Language) similar to the bytecode. The JIT (Just-In-Time) is the intermediate code interpreter that will allow implementation across multiple platforms. The BCL (Basic Class Library) possesses the essential operations in building applications.

This set of class libraries is Microsoft's response to the advance of Sun's JEE architecture.

ESB (Enterprise Service Bus) [27] [31] architecture can be considered a software infrastructure that provides, through the exchange of messages, a system based on events. The middle layer (middleware) is able to integrate different

applications through messaging and synchronization services.

The ESB is based upon message processing to achieve independence of operating systems and programming languages.

The main features are *transparency of the locations*, meaning one must know the name of the service, and *transparency of transpor*t freeing the protocol from the services, which allow us to solve the problem of N * M connections that exist in any of the current business systems and characterizes the peer-to-peer.

SOA (Service Oriented Architecture) is an architecture that is currently being implemented more widely in the business world. It is focused on the procedures for creating and using various processes called services. The gathering of various services can be described as a *Business Process*.

The technologies used are based on three: XML, WSDL (Web Service Description Language) and SOAP (Simple Object Access Protocol). The services by design have a low level of coupling that allows the performance of its components with many programming languages.

According to [7] there is a set of principles that must be met to provide a service-based architecture which can be discussed to make the following proposal:

- Reusable services.
- Provide a formal agreement.
- Weak coupling.
- Allow the composition.
- They must be autonomous.
- They should be stateless.
- They should be able to be discovered.

The problem of serialization

The exchange of information in mobile agent environments has the uniqueness of using the same serialization (marshalling). This involves the coding of the object in a storage medium for later reconstruction either in the system itself or by other remote mechanisms. No wonder that one of the first steps before transmiting the agent is the serialization of code as data, which characterize the running process, for later transfer.

Initially, the environment must be restored using the agent in another machine but with a high cost in some cases. The main issue concerning the serialization in environment where large-scale files are used must be considered when taking into account to arriving at a solution valid and operational.

Our application will become more heterogeneous with regard to the use of different programming languages by ignoring these issues..

Numerous proposals have been put on the table to serialised objects in XML. The W3C has made some to find feasible solution to send information using XML in those cases where binary formats were commonly used [20] [38].

Even making different types of proposals for the serialization with XML technology, it has not been taken into account the possibility that different parts of the object itself are not required at one location and therefore their transmission is not necessary.

XML has described it as a sub-optimal bit format for tranfering data in relation to existing binaries systems. Therefore, the data-binding is no exception and has designed several alternatives for data transfer on systems where the bandwidth is limited and requires a further optimization of the transmission. This is not our case and this issue will not be discussed (any) futher.

What we want to incorporate features to mobile agents

The architecture of mobile agents proponed is a set of specifications that allow operators to move through different environments where they use the services they are provided to them. The mobile agent must not be understood as a software program that is capable of performing tasks completely independently or semi-autonomous but they are the sum of the services we are able to use on the go. Considering this characteristic, the heterogeneity of systems is an indispensable need for this type of environments, and migration of agents guaranteed by both object-oriented environments like any other for which they are designed.

La serialización de los datos nunca debe ser dependiente del lenguaje de programación. Por ello se propone una modelo de agente móvil que permita el intercambio como base para la pruebas. En esta propuesta, el agente no se conforma por sus datos y programas sino que se constituye por los metadatos que deberá utilizar, eso si, teniendo en cuenta que los datos propios del agente están disponibles en cualquier momento en la propia red en la que se está ejecutando.

The serialization of data should never depend on the programming language. Therefore a proposal has been put forward of a mobile agent model which allows for the exchange as a basis for testing. Under this proposal, the agent does not accept their data and programs but that is constituted by the metadata to be used, conditioned to the availability of the agent's own data at any time in the network in which it is running.

In this way we approach the idea of a "Joint Services" available to perform tasks, usable by a user group not previously set. This idea ressembles the delivery in public services.

What is Genesis?

Genesis represents an attempt to create an environment for new proposals of mobile agent technology as transportable objects seeking services independently and intelligently.

The organisational structure of the genesis of agents is horizontal. All those who enter the system are at the same level as the existing ones, competing with each other for services on an equal footing. It can be categorised as a P2P network characterised as a pure structure and unstructured, allowing the exchange of basic information among different locations on the services that they are capable of doing.

Genesis agents

Genesis agents are defined as entities using services that are later transferred through the exchange of metadata using XML-formatted files. The main function of an agent is to coordinate the implementation of a management task using the services available provided by the existing infrastructure.

With the arrival of the agent metadata, information is no longer available to run the system. Genesis is the system which is responsible for building the environment necessary for the implementation of the transfer agent based on the information provided by the metadata. Thus, an agent is composed of several parts that may or may not be necessary depending on the timescale of the execution.

La idea es similar al funcionamiento de las páginas web realizadas con código HTML. No toda la información está dentro del documento que se transfiere en un principio, sino que este dispone de referencias a imágenes externas que el

navegador se encarga de cargar cuando es necesario realizar la presentación (ej. <IMG SRC="little_car.gif">). Dos pueden ser las diferencias básicas respecto al entorno génesis, por un lado no solamente son imágenes sino que se ha generalizado y pueden ser otro tipo de datos y por otro lado, no tienen que bajarse cada vez que se muestran en pantalla sino que se realizará la descarga únicamente cuando el agente, dentro de su lógica, considere necesario su uso.
The idea behind this concept resembles how web pages are set up using HTML. Initally not all information is within the document to be transferred, but this has references to external images that the browser is responsible for loading when a presentation needs to be carried out (eg SRC="little_car.gif"> <IMG). There are two main differences in a genesis environment, first there are not just images but their use is nowadays widespread and may be other types of data. Second, they do not have to be downloaded every time the screen is on but the download will be undertaken when the agent, subject to its internal logic, deems its use necessary.

The exchange of files using encrypted prototype is always carried out through the exchange protocol HTTP. These exchanges also include the files that make up the parameters of the agent and who move from one host to another based solely on the need for the agent himself.

An example of an agent Genesis is as follows:

```
<?xml version="1.0" encoding="UTF-8"?>
    <agent name="Genesis" version="1.0">
        <control>
            <number>10258</number>
            <name>Test</name>
            <version>1_0_0</version>
            <service_name>service_1</service_name>
            <service_version>1_0_0</service_version>
            <start>
                <host>
                    <id>aeDf134</id>
                    <location>
                        <protocol>http</protocol>
                        <dir>172.20.20.54</dir>
                        <port>8080</port>
                        <path>genesis/upload</path>
                    </location>
                </host>
            </start>
            <origin>
                <host>
                    <id>aedgh</id>
                    <location>
                        <protocol>http</protocol>
                        <dir>172.20.20.54</dir>
                        <port>8081</port>
                        <path>genesis/upload</path>
                    </location>
                </host>
            </origin>
            <dest>
                <host>
                    <id>asde123</id>
                    <location>
                        <protocol>http</protocol>
                        <dir>172.20.20.50</dir>
                        <port>8082</port>
                        <path>genesis/upload</path>
                    </location>
                </host>
            </dest>
            <date>2010-11-17</date>
            <time>10:23:00 </time>
            <hops>5</hops>
        </control>
        <data>
            <params>
                <file_param label="final">
                    <genesis_name>history.old</genesis_name>
                    <remote_name>history0.txt</remote_name>
                    <location>
                        <protocol>http</protocol>
                        <dir>www.host.com</dir>
                        <port>8080</port>
                        <path>genesis/download</path>
                    </location>
                </file_param>
                <file_param label="">
                    <genesis_name>histo.old</genesis_name>
                    <remote_name>histo.txt</remote_name>
                    <location>
                        <protocol>http</protocol>
                        <dir>www.host.com</dir>
                        <port>8080</port>
                        <path>genesis/download</path>
                    </location>
                </file_param>
                <data_param>
                    <data_name>status</data_name>
                    <data_value>true</data_value>
                </data_param>
                <data_param>
                    <data_name>name</data_name>
                    <data_value>anna</data_value>
                </data_param>
            </params>
        </data>
    </agent>
```

This system is offered as a solution serialization of agents, and thus facilitate its exchange. Also, duplication or cloning of agents becomes a trivial matter because the data are not duplicated. The encapsulation of metadata for SOAP agents can be performed by XMPP or any other format developed in the near future. In order to perform the tests to be described later, it has been used only the encapsulation in a transfer protocol HTTP.

In summary, the Genesis system provides three major tasks:

- The serialization of the agents, as programs are transferred the responsibility for this task. The platform is independent of programming languages and systems to be used.
- Information sharing is efficient. The total reconstruction of the agent is not made until the final moment of his execution by preventing their transfer to nodes that are unwilling to use it or do not have the necessary permits for treatment.
- It provides implementation of model-program-mobility data (DPM) in which the movement of the agent is a part of software program design.

The submission process is redefined in Genesis for the system in the following steps:

- The agent updates the data needed for transfer
- Metadata are encoded in a transfer protocol.
- Transfer the agent.

At the reception

- The agent is received.
- The essential data are located for implementation.
- Agent is instantiated.
- The agent requests additional data that may be required.

As we see, the proposal differs greatly from the proposal shown above because it gave the responsibility to agents to

make obtaining the data when necessary and will not unnecessarily need to transfer all the data at each level.

Genesis Communication between agents

One of the most essential functions within a generic infrastructure that provides agents is the communication between them. The exchange of messages between mobile operators is particularly complex. That is why Genesis is designed to make the transfer of information between agents by using a special kind of agent-message so allowing some level of intelligence in the search process,. The transfer of a message between source and destination is done using an agent-message that will be able to locate the agent-target to deliver the order.

This proposal, easily implementable in small environments, significantly increases the complexity when the aim is finding an agent in heterogeneous environments or large systems.

It has been suggested that the process of finding an agent is by tracking it down. The footprint is the information left in each of the locations in the implementation process of an agent. Once the footprint trail of the movement of destination-agent is found, the agent-message will follow it to achieve a perfect match on the same infrastructure, then it will deliver the message.

What is a service for Genesis?

A service represents the capacity of a system to perform a transaction fully and independently. The services do not have an exclusive location and several may be scattered in different infrastructures. Always choose the service server in a random fashion disregarding the physical location.

The procedure of numbering agents should ensure that they have a unique identifier on the system. Any node has a unique code that has been implemented through a combination of random numbers and the system time data (SUC - Unique System Code). The code of the agent is generated from the system SUC. This system does not ensure that the numbering is unique, but it is highly unlikely that the identifiers of the agents and/or nodes match.

The system can be represented as a platform where different nodes locate services with a well-defined name, and that allows the other systems, locate them properly. Therefore, the agents moving in the environment are able to locate the necessary services regardless of their physical location. Since the service with the same name carry out the same functions, regardless of their geographical location, the needs of agents in terms of published services can meet regardless of the origin or destination.

In a practical case, one of the services available is the capability to digitally sign a document containing information on the date and time. This service is called time-stamping, is done by certification authorities called TSA (Timestamping Authority). The TSA generates, based on the result of a short hash function, an electronic document which ensures the integrity of the source of the accompanying document. This service is a candidate to be presented as a genesis service because the data of the source document is not required to move to the TSA for sealing only the abstract will be transfered.

Designs based agents in the proposed DPM (Data-Program-Mobility)

All programs created in Genesis are based on DPM. Its design is based on the use of a well defined interface that can be summarised in four methods that must be executed sequentially.

- local_runner
- local_return_runner
- img_runner
- img_return_runner

During their execution agents go through two phases: the local stage and the imaginary stage which corresponds to the software that handles the typical running of the program and the implementation/running of the new part of mobility respectively. The information about the data is always shared as member variables.

There is no difference between the invocation of agents and services. In fact, the implementation of an agent is considered as the implemention of a service with the name of the agent. Hence, it is necessary that a genesis.transmisor-type agent locates a genesis.transmisor-type service for its execution after go through a service [genesis.servicio.S].

Identification Services

It is the responsibility of the agents to decide the destination where they can received the services required, but infrastructure is responsible for the maintenance of services location. In order to carry out this function, a list of locations is available. This service, known as SNS (Server Name Service), uses an algorithm for the exchange of agent-based tables. There is a specialized agent with such purpose that performs the search function in the different infrastructure and is capable of returning to its origin with this information.

The DHTs using Genesis are based on processes carried out by SNS-type agents. The the process begins with the location of the service, called global.genesis.friend. This service must be provided at the beginning of the system as connector between the existing services. An agent is sent to conduct the search for those services that may be of relevant. The agent will carry out the search the [global.genesis.friend]-type services having the system moving from one to another to obtain the data required to return them to the originator or exceeding the maximum number of hops permitted.

The data retrieved will be listed in a table with information containing the type of service and its location which will be renewed with the information gathered by different agents locating information in a given time frame.

If a specific entry within this table is not found, infrastructure will have the right to issue/create a search agent that can locate a particular service.

Criteria to be followed to evaluate the system

It is a complex task to evaluate the system proposed to set the criteria since they can vary significantly depending on the systems to which it applies. It is also difficult to make a comparison with the systems currently being used due to the different principles on which they were designed. The criteria followed to make the comparison between the current systems of data exchange based on other architectures and the proposed sytem is based on the following quality standards:

- Flexibility in the design process.
- Adaptability and availability.
- Delegation of work.
- Efficiency.
- Quality of information.
- Collaboration.
- Portability.
- Response time.
- Maintenance.

- Organisation.

Assessment of the migration system

In the same way as in other studies [23] have made comparisons between different ways to migrate objects, this will have a series of tests which evaluate the solutions given by Genesis for the transfer of information.

Two tests have been selected to verify the results of any hypothesis made. The first test relates to the speed of agents' transfer whereas the second test will concern the agent's ability to search services independently.

For the first test, it is expected during the design phase, improved response times but experiments and testing have revealed a highly optimized response even in very small volumes of data. In the second case the convergence of data was an essential task of the system.

The environment in which it was tested consisted of two machines desktop PC with Pentium processors and Windows OS and Linux Red Hat, which have four settings Apache Tomcat versions 5.5 and 6, and simulating the presence of four interconnected by infrastructure Genesis 100Mb Ethernet network.

JAVA and JEE programming were used. The exchange of agents' metadata is carried out using servlets receiving agents sent by another servlet. The code is assembled in a file extention .war with the possibility to be set up on any system available in a Tomcat servlet server.

Test A. Test Initially, five files have been prepared for the transfer test of various sizes to simulate a real production environment. It ruled that the maximum sizes of the field study and does not constitute other skills that are not reflected in the tests with smaller sizes.

We have implemented an agent with a distance of five steps that is common to all tests. The origin and the end was the same infrastructure to achieve in this way share a common clock and set the time spent in testing

The newly-designed system suggests that the files transfered as parameters are received immediately (Full type) or be brought in only when the agent decides they are required (Type Final). The agent has the sole function of moving through all test environments, although the data file is not required until it reaches the final location, which simulates the treatment site. The values of time needed to complete the task, which obtained empirically, is expressed in the table below:

		File type transmission	
		Full	**Final**
File Size	**1Byte**	698	464
	100KB	870	385
	1MB	28.729	885
	2MB	35.464	2.240
	10MB	666.448	2.125

all time in ms

Table1. Response Time (ms) test system based on the types of transmission used

graphically:

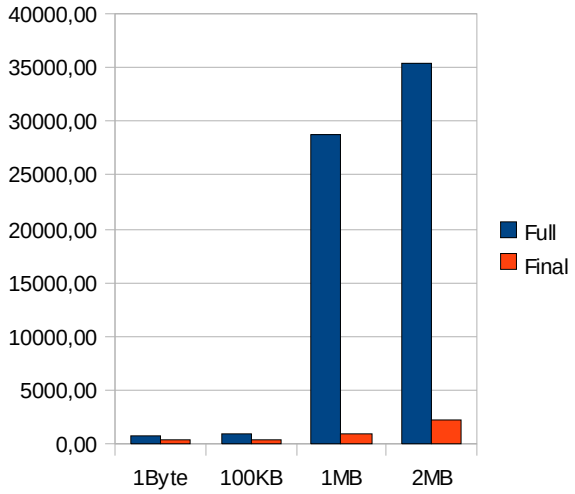


Figure 1. Vista's response time (ms) test system based on the types of transmission used

The time display 10MB file transfer has been removed from figure 1. That means that the order of 310 times slower in Full option in Final choice, demonstrating a significant improvement of subsequent sizes have not been detailed.

Test B. The ability of the system to converge and track the status of needed services. It verifies that each of the infrastructure is able to get a valid array of services consisting of the various services provided by other systems of installed agents.

Reproducing the same environment as Exhibit A, various services in different systems have been made. Thereafter they have been told the friendly system "system.genesis.friend" for its entry into the system.

Depending on the time set used to exchange information sns system tables, in all cases where the configuration was correct, has reached a point where the boards were replicated across all systems in the different machines.

Comparison and evaluation of the system.

A system with objective is required to make feasible comparisons in the design phase, similar to those that may be undertaken with a mobile agent system and made with a different architecture, but has been ad-hoc implemented to carry out this task via web services or in the file that exchanges are conducted in a near-classical fashion.

Depending on the criteria set above the following observations can be made:

- Adaptability and Availability.—A mobile agent-based system allows for greater adaptability and availability of services intrinsecly.
- Delegation of work. Agent-based systems are characterised by the delegation of work between agents. It is a fact that in other systems should be implemented.
- Flexibility in the design process. The process of defining the actions of an agent to indicate the genesis allows remote tasks to be run rather than the sites where they ought to be undertaken. In summary, it is not until the time of the execution when the exact place is located for its implementation through the structure DPM.
- Efficiency. By understanding the proper use of the factors that make it up, it can be asserted that its highest level is reached since

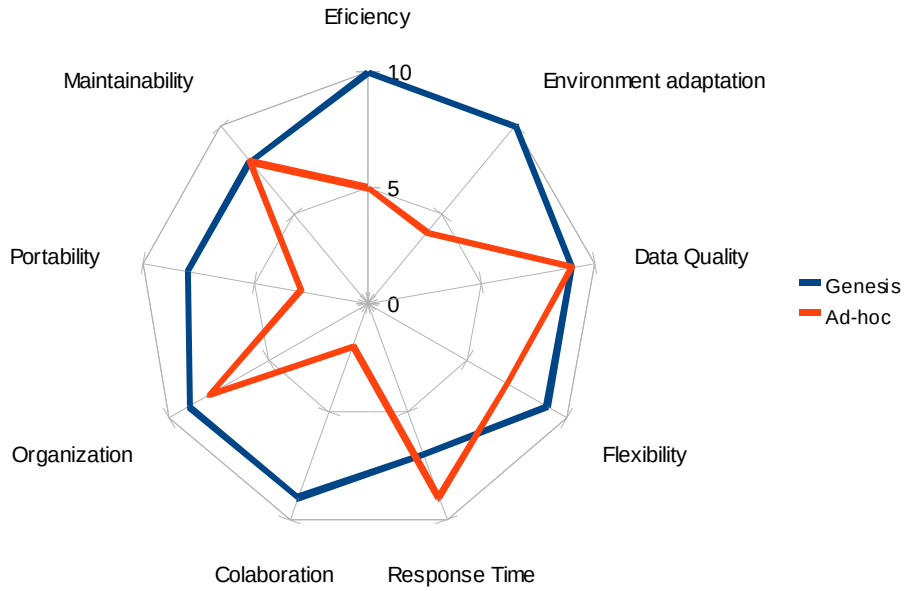


Figure 2.. Comparison of Genesis and a system designed Ad-hoc

the tasks performed by agents running on the nodes are prepared to do the job.

- Quality of information. The information is obtained directly from the sources without the use of local copies.
- Collaboration. The nodes are able to cooperate providing the environment in which those services are more efficient or are better adapted.
- Portability. The system is initially available in Tomcat and JAVA environment having implications in terms of portability to different systems, but it is also portable to other web architectures and is capable of carrying out the transfer by other protocols of agent metadata.
- Response time. The response time depends directly on the system's ability to perform the appropriate task at the right place. According to further testing, a significant improvement can be achieved in this field.
- Maintenance. The maintenance of a mobile agent application should not exceed the amount that is made for an ad-hoc system. Possibly, the separation of mobility of other components will provide a better level of maintainability/maintenance.
- Organization. Agent systems maintain strictly the organization with which they are executed, allowing "islands" of competence.

- Adaptability to the environment. If any of the characteristics of mobile agent systems are to be highlighted, this would be the ability to perceive and adapt to the environment in which they run.

Based on the criteria discussed above and compared with an ad-hoc system designed to meet similar data transfer needs, we would arrive to Figure 2.

Let's talk about security

Although dealing with security in depth is beyond the scope of this document, some features which arise from the use of both mobile agent model and the platform can be highlighted.

The mobile agent model does not provide greater security for the fact of having been designed as a distributed system. Article [12] named the different elements that we must cover to achieve an acceptable security level, and the different issues that both the host that contains the agent and mobile agent might have. [21] focuses on four basic situations:

- host protection against agents
- protection against other agents
- protection against the host agent
- network protection

We should note that the services provided by a host origin are authorized by the system administrator. Genesis does not allow the execution of unknown software, so eliminating a large group of potential security issues. Yet problems such as eavesdropping, impersonation, modification, forwarding and non-repudiation, are issues to be addressed in more depth.

Genesis is not intended to define the means of transmission of data between different hosts, so be open to using any protocol, secure or insecure, you want to implement. Thus issues such as eavesdropping or alteration of the message are issues already been solved by many transfer protocols.

The failure to include the data within the agent itself allows many security designs to be applied to them, as well as the possibility of making them available only at origin and destination makes it easier to apply to them some sort of encryption/compression.

Conclusions

Mobile agents allow the exchange of information more efficiently and easily. The implemention of an architecture which ensures complex tasks can be undertaken in a natural and distributed, paves the way to the task of design of these exchanges.

Having described the basic ideas for the implementation of mobile agent technology in the movement and transfer of large files, it can be argued that, based on data from tests with the prototype, the use of mobile agents Genesis facilitates and clearly improves the exchange of files in situations where the location is not a final host information known at the start of implementation process.

The use of services which location is not defined at the time the system was designed and the capability of adaptation of mobile agents facilitate the use of generic services provided by third parties.

The automatic serialization of objects, available in numerous programming languages, is a feasible option for small document sizes or when "steps" in performance are not numerous. The enhanced performance obtained by autonomous controls in the serialization process ensures the reduction of transfer times between origin-destination locations, so optimizing available resources.

We have achieved, in the same way as other existing architectures such as ESB, an independence of the platforms, ensuring scalability and applying, as far as possible, a unique intelligence for each transfer problems arising.

Regarding the SOA systems mentioned above [7], the following conclusions can be made by returning to the principles first formulated by Thomas Erl:

- Re-usable services. All services provided by Genesis can be used by any (other) agents operating the system.
- It provides a formal agreement. A formal interface has been defined for both the execution of agents and the services used by agents when they reach a location.
- Weak coupling since there is no link between mobile agents and services
- It allows the composition. It allows creation of complex systems as a composition of basic services.
- They must be autonomous. Services are autonomous, but more mobile agents are involved.
- They should not have been. The state is not associated with the mobile agent to the input.
- They should be able to be discovered. SNS Agent-based service that enables the discovery of available services in other locations.

Some more items can be added as follows:

- No dependence on the environment of implementation. The system should not be dependent on a single location either in services nor agents, increasing the availability, fault tolerance and horizontal scalability of the systems as a result.

Future Work

As discussed early on, security is one of the characteristics that should be take more care of in a system allowing the exchange of information, The implementation of security systems better adapted to this reality should not be forgotten. This text has not explored such security systems.

The implementation of a prototype does not allow the evaluation of a fully functional system. Consequently, to develop their potential fully, based on the ideas raised above, functional and scalable software should ensure the provision of similar performance to the proposal made.

The study and comparison with other mobile agent systems allow the assessment of improvements already obtained and the most optimistic scenarios where it is possible to use an idea based on the transmission of files.

The ease of exchange of information seems especially subject to machine-machine relationships, but the relationship man-machine has been intentionally neglected. It would be necessary to carry out some tasks to foresee the behavior of the agents environment genesis in man-machine relations.

In the proposed model of mobile agents, only the basic lines of communication between agents and their interaction have been sketched out. Its definition and study would allow the advancement in the application of the theories discussed in this paper.

Bibliography

[1] Aglets. <http://www.trl.ibm.com/aglets/>
[2] Ametller, J.; Robles, S; Borrel, J.: "Agents Migration over FIPA ACL Messages". Lecture Notes in Computer Science (LNCS). Vol. 2881. Pages. 209-2019. 2003.
[3] Bellifemine, F.; Caire, G.; Poggi, A.; Rimassa, G.: "JADE: A software framework for developing multi-agent applications. Lessons learned.". Information and Software Technology. Vol. 50. Issue 1-2 Pages 10-21. Jan 2008.
[4] Bellifemine, F.; Caire, G.; Poggi, A; Rimassa, G.: "JADE: A white paper". Exp. Vol. 3 – n.3 Sept. 2003. Web: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.115.9030&rep=rep1&type=pdf>.
[5] Bellifemine, F.; Rimassa, G.; Poggi, A.: "JADE - A FIPA-compliant Agent Framework". Proceedings of the 4th International Conference and Exhibition on The Practical Application of Intelligent Agents and Multi-Agents. 1999.
[6] Bray, T.; Paoli, J; Sperberg-McQueen (Eds), C.M.; Maler,Eve: "Extensible Markup Languaje (XML) 1.0 (Fifth Edition)" W3C Recommendation. http://www.w3.org/TR/2008/REC-xml-20081126/
[7] Erl, T.: "Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services" . Prentice Hall. 2004.
[8] Escrivá, M; Palanca, J; Aranda, G: "A Jabber based MultiAgent System Platform". International Conference on Autonomous Agents. Pages: 1282 - 1284 . 2006 ISBN:1-59593-303-4
[9] Ewing, D.J.; Hall, R.S.; Schwartz, M.F.: "A Measurement Study of Internet File Transfer Traffic". Citeseer. 1992.
[10] Fielding, R.; Gettys, J; Mogul, J.C.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T.: "Hypertext transfer protocol – HTTP/1.0" , RFC 2616 . Junio 1999. Web: <http://tools.ietf.org/pdf/rfc2616.pdf>
[11] FIPA. Web: <http://www.fipa.org>.
[12] Greenberg, M.S.; Byington, J.C.; Harper, D.G. : "Mobile Agents and Security". IEEE Communications Magazine. July 1998.
[13] Heery, Karl:"Agent-based Workflow Management using XMPP and CARTAGO ". Thesis degree of MSc Advanced Software Engineering . School of Computer Science and Informatics University College Dublin. 2009.
[14] Hormat, D.; Cvetkivic, D.; Milutinovic, V.; Kocovic, P.; Kovacevic,V.: "Mobile Agents and Java Mobile Agents Toolkits.". Lecture Notes in Computer Science (LNCS). Vol. 18. Pages 271-287. September, 2001.
[15] Jabber Software Foundation (2003), Jabber Instant Messaging User Base Surpasses ICQ, Web: <http://xmpp.org/xsf/press/2003-09-22.shtml>
[16] Jabber Software Foundation (2005), Web: <http://www.jabber.org>
[17] Jabber Software Foundation: "Extensible Messaging and Presence Protocol (XMPP): Core". 2004. Web: <http://www.ietf.org/rfc/rfc3920.txt>
[18] Jabber Software Foundation: "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence". 2004. Web: <http://www.ietf.org/rfc/rfc3921.txt>
[19] Java Community Process. Web: <http://jcp.org/en/home/index>
[20] Kangasharju, J.; Tarkoma, S. : "Benefits of Alternate XML Serialization Formats in Scientific Computing". SOCP '07: Proceedings of the 2007 workshop on Service-oriented computing performance: aspects, issues, and approaches. California. EEUU. 2007.

[21] KarJoth, G; Lange D.B.; Oshima, M: “A security model for aglets”. IEEE Internet Computing. 1997.
[22] Keogh, Jim: “J2EE. Manual Referencia”. Edit. McGraw-Hill. 2002.
[23]Krol, D.; Lupa, A.: “Agent Migration: Framework for Analisys”. Journal of Universal Computer Science. Vol.15 no.4 . 2009.
[24]Lange, D.B.; Oshima, M.: ”Seven Good Reason for Mobile Agents”. Comunications of the ACM. Volume 42 Issue 3. Pag. 88-90. March 1999. ISSN:0001-0782
[25] Lingnau, A.; Drobnik , O.; Dömel , P. : “An HTTP-based Infrastructure for Mobile Agents”. Fourth International World Wide Web Conference ``The Web Revolution'' December 11-14, 1995, Boston, Massachusetts, USA . Web: <http://citeseekx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.3296&rep=rep1&type=pdf>
[26] Lua, E.K.; Crowcroft, J.; Pias, M.; Sharma, R.;Lim,S.: “A survey and Comparison of Peer-to-peer Overlay Network Schemes”. IEEE Communications survey and Tutorial. 2007.
[27] Merge, Falco: “Enterprise Service Bus”. Free and Open Source Software Conference 2007.
[28] Milojivic, D.; Breugst, M.: “MASIF. The OMG Mobile Agent System Interoperability Facility”. Lecture Notes in Computer Science (LNCS) Vol. 2. Pages 117-129. June 1998
[29] Nielsen, H.; Leach, P; Lawrence, S: “ An HTTP Extension Framework” . Especificación HTTP 1.2. Febrero 2000. RFC 2774. Web : <http://tools.ietf.org/pdf/rfc2774.pdf>
[30] OMG Home Page.<http://www.omg.es>.
[31] Papazoglou, M.P.; Traverso, P.; Durstdar,S.; Leymann, F.: “Service-Oriented Computing: State of the art and research challenges”. IEEE Computer. Vol. 40. Pages 38-45. 2007.
[32] Sun's Java EE Web: <http://java.sun.com/javaee/index.jsp>.
[33] Tanenbaum, A.S.; Herder, H.N.; Bos, H.: “File size distribution on UNIX systems: then and now “ . ACM SIGOPS Operating Systems Review. 2006.
[34] Thompson, H. S., Beech, D., Maloney, M., and Mendelsohn (Eds),. “XML Schema Part 1: Structures”. W3C Recommendation. 2001. Web: <http://www.w3c.org/TR/xmlschema-1>
[35] Trillo, R; Ilarri, S.; Mena, E.: “ Comparison and Performance Evaluation of Mobile Agents Platforms”. In Proc. 2007 the Third International Conference on Autonomic and Autonomous Systems (ICAS'07), Athens, Greece.
[36]Van Engelen, R.; Zhang, W.; Gobindaraju, M.: “Toward Remote Object Coherence with Compiled Object Serialization for Distributed Computing with XML Web Services”. In the proceedings of Compiler for Parallel Computing (CPC) pages 441-455. 2006.
[37] W3C XML . Web : <http://www.w3c.org/XML/>
[38] W3C. Web : <http://www.w3.org/TR/xml-infoset/>