IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITP4918

Framework for Mobile Agents to take Advantage of the .NET Environment

Eder, Shiu Ming Tsang
School of Multimedia Systems, Monash University, Australia, eder_tsang@hotmail.com

Raymond Koon-Ying Li
School of Multimedia Systems, Monash University, Australia, raymond.li@infotech.monash.edu.au

ABSTRACT

Microsoft released .NET a few years ago and since then, its support from software developers around the world has been rapidly gaining momentum. Microsoft has included XML Web Services within its .NET framework. As XML Web Services are standard based, vendor neutral, discoverable through service registries, and essentially oriented toward program-to-program interaction, they are therefore ideal ingredients for building mobile agent systems. Despite mobile agent technology is currently attracting a lot of attention from both industry and academia; a literature survey indicated that Microsoft has not yet released a framework to guide the deployment of mobile agents within its .NET environment. In this paper, a framework for the deployment of mobile agent is presented, showing how XML Web Services within the .NET environment can be utilized to provide the required security, autonomy, mobility and agent discovery functions.

PREAMBLE

According to Vinaja (2001), there are three types of mobility in the computing world:

- User mobility
- Hardware mobility
- Software mobility

Under *user mobility*, using multiple "user profiles", a computer can be shared by a number of users and a user can also move between computers without the fear of their data being exposed to other users. By maintaining a separate user profile for each user on the same computer, each user can have its own protected set of data, applications and preferences. With same profile maintained on different computers, a user, when moving from one computer to another, will be greeted by the same familiar interface.

Laptops and handheld mobile devices such as Plam Pilot and Pocket PC, facilitate hardware mobility. Today a handheld device not only provides daily personal assistance functions such as scheduling and reminder but also has built-in word processor, database functionality and multimedia capabilities. Some of the latest ones even support Bluetooth and Wireless Fidelity (Wi-Fi) and Third Generation (3G) technologies. It allows its owner to continue to be productive while on the road and at home.

Under *software mobility*, the definition is the software that moves from one computer to another computer in a remote location. Mobile agent technology typifies software mobility. A mobile agent is a piece of software which travels from one computer to another within a network of heterogeneous computers. Its itinerary can be defined by its owner or "self" determined. When an agent completes its work at a location, it will migrate to a new host after its current state was saved. At the new location, its state will then be restored, allowing the agent to continue where it left off.

Microsoft Windows 2000 and XP offer multiple user-profiles with a secured login function and therefore provide a framework to support *user mobility*.

Both Microsoft Windows XP and Pocket PC 2003 have Wi-Fi support and therefore provide a framework to support *hardware mobility*.

Despite mobile agent technology is currently attracting a lot of attention from both industry and academia; a literature survey indicated that Microsoft has not yet released a framework to guide the deployment of mobile agents within its .NET environment. Hence, the main objective of this research is to develop a framework that would support the deployment of mobile agents within the Microsoft .NET environment.

Microsoft released .NET few years ago and since then, its support from software developers worldwide has been rapidly gaining momentum. Microsoft has included XML Web Services under its .NET framework. Web Services are one of the latest and the most powerful tool that is currently offered to software developers.

Web Services are *standard based and vendor neutral*. Interaction with Web Services is often achieved using Internet standards such as *Transmission Control Protocol/Internet Protocol* (TCP/IP), *Hypertext Transfer Protocols* (HTTP) and *Extensible Markup Language* (XML). XML Web Services expose functionality through a new standard known as *Web Services Description Language* (WSDL) and another new standard known as *Universal Description, Discovery, and Integration* (UDDI) facilitates registry services for Web Services discovery purposes.

The concept of Web Services is oriented towards program–to-program interaction. *The Simple Object Access Protocol* (SOAP) is the preferred mechanism used to send messages between client programs and XML Web Services. SOAP is not tied to any programming language and operating system. To consume Web Services, the only requirements are the ability to communicate over TCP/IP and the ability to process XML. Web Services are therefore *language and platform independent*.

Given the above, Web Services therefore provide good candidates for the deployment of mobile agents.

In this paper, the benefits and issues of using mobile agents are first discussed to explore the necessary and essential functions and conditions of mobile agent systems. The components of .NET framework which are appropriate for the deployment of the mobile agent technology are then described. Based on the above, a framework, showing how Web Services and other components within the .NET environment can be used to provide the required security, autonomy, mobility and agent discovery functions for the deployment of mobile agents, is derived and presented.

The salient features of the framework are:

1. Use of the *brokering agen*t to enhance security.
2. Use of *directory services* for Web Services (remote host functionalities) publishing and discovery.
3. Use of the *chatting server* to isolate all incoming agents from corporate resources.

BENEFITS AND ISSUES OF USING MOBILE AGENT

The followings are some of the benefits of using mobile agent: (Lange and Oshima, 1999)

Network connection

Mobile agent reduces the bandwidth requirement as all interactions between an agent and a target host, are done locally. Besides the agent migration process, no data is required to be transmitted between the target host and the client machine, which launches the agent.

Heterogeneous environment

An agent is expected to move among computers and therefore should be able to perform under all different hardware and software configurations at any target hosts.

Mobile service and information retrieval

Portable devices generally do not connect to a network permanently and may not always be connected to the same network at the same place. A mobile agent should support such a requirement. A mobile agent should be able to tolerate any transient disconnection of its launching machine from the network after it has been dispatched. After an agent completes its task at a remote host, it will return to its owner's machine whenever the network connection is re-established.

Host push system

To avoid frequent scanning of a host machine for any new information, an agent can also be programmed to be resident in a target host to discover new information at the host and only return home when new information is made available and acquired. This also conserves network bandwidth.

Dynamic Deployment

A mobile agent can travel to a remote host, where it can aid in the dynamic deployment of new components there. For example, an owner can order its agent to travel to a remote server and install a particular ODBC driver there.

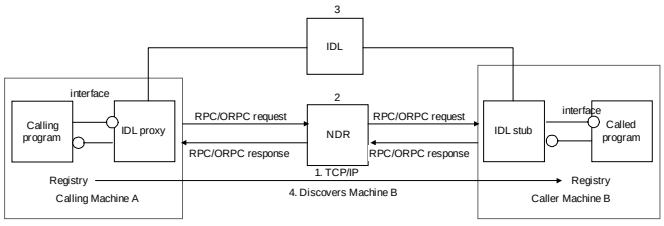
This virus-like behaviour can raise security concerns. The above discussions highlight the services, such as mobility, intelligence, automation and agent discovery, which are required to support mobile agent functions. Moreover, it also raises issues such as security, reliability, language independence and cross platform considerations.

To make it easier to understand the .NET framework, some of the reasons why Microsoft abandoned the traditional DNA and moved onto .NET will first be discussed.

MOTIVATION FOR MICROSOFT TO MOVE FROM TRADITIONAL DNA TO .NET

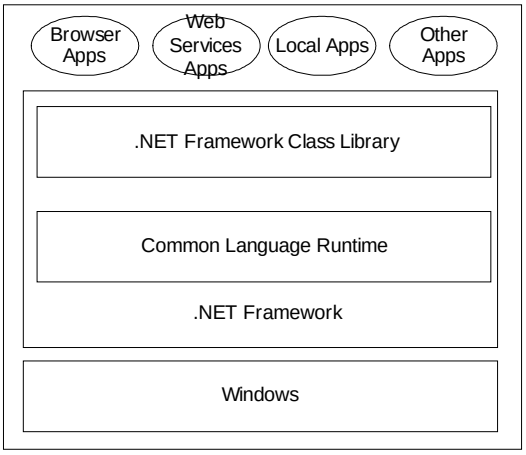
Under Microsoft *Distributed interNet applications Architecture* (DNA), *Remote Procedure Call* (RPC) and *Object Remote Procedure Call* (ORPC) are used to facilitate communications between two computers in separate locations under the *Distributed Common Object Model* (DCOM). Figure 1 illustrates an example of DCOM calling. (Skonnard, 2002)

Figure: 1 DCOM Example. Source: Skonnard (2002)



1. *Communication protocol*
2. *Message format*
3. *Description language*
4. *Discovery mechanism*

Figure: 2 .NET framework Source: Chappel (2002)



Under the DCOM model, objects on both sides communicate with each others using TCP/IP protocol to transfer a specific message encapsulated using an efficient binary format called *Network Data Representation* (NDR). The *Interface Description Language* (IDL) is used to shield developers from the complexities of the communication and handles all stub-related issues. The registry of the calling machine needs to be pre-configured in order to discover the location of the caller machine as well as the location for its DCOM objects. Furthermore, both NDR and IDL are Microsoft specific. Therefore, when traditional DNA is used, there are language dependent and platform specific issues.

Applications built on the .NET framework do not have as many problems as when comparing to those built under the traditional DNA. Microsoft .NET framework uses the *Common Language Runtime* (CLR) to define a common approach to all interfaces and data types (see Figure 2). As a result, the need to resolve problems that exist when different programming languages are used under DNA, are no longer required under the .NET framework. Furthermore, many firewalls block non-HTTP requests such as NDR whereas such a problem does not exist when XML Web Services are used under .NET framework.

Web Services, in essence use a collection of standards and protocols that allow users to make processing requests to remote systems by speaking a common, non-proprietary language and using common transport protocols (such as HTTP, SMTP) (Waterhouse, 2002). The *Simple Object Access Protocol* (SOAP), an XML-based protocol (Refer to Figure 3), provides intra-process communications across machines through the firewall. The .NET framework therefore resolves the language dependence and platform specific issues that are associated with the DNA environment.

COMPONENTS OF WEB SERVICES

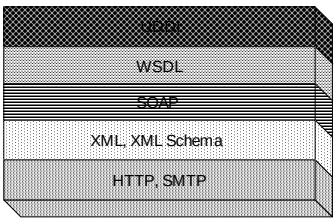
Figure 3 shows the Web Services stack.

Web Services were designed to expose their functionality to the entire world and to any devices. All web enabled (using standard protocols such as HTTP) applications will be able to access Web Services even within the .NET environment. For example, an Internet browser of a handheld device such as a mobile phone and Pocket PC can now access Web Services any time and from any place.

Extensible Markup Language (XML) is the key component for Web Services. The SOAP, WSDL, and UDDI are all built on top of XML. SOAP uses XML as the data-encoding format. WSDL uses an XML schema to describe the structure of a Web Services. UDDI uses an XML schema to define the structure of the registry and uses SOAP to specify the communication with registry. (Basiura 2002)

Using Web Services for the deployment of mobile agents can take advantage of the existing network infrastructure and standard communication protocols provided by the Internet. This approach is not only convenient for developers, but also ensures that an open framework is being used. Web Services is not a proprietary technology. It is supported

Figure: 3 Web Services stack. Source: Basiura (2002)



by IBM, the W3C, Microsoft, Sun, The Stencil Group, Oracle, Borland, BEA and Hewlett-Packard (Myerson, 2002).

Since SOAP, WSDL and UDDI are all utilizing XML. An agent implemented using Web Services will be language independent and can function across different platforms.

MOBILE AGENT TECHNOLOGY CAN BE ENHANCED BY WEB SERVICES

Extensible Markup Language (XML) can represent both structured and semi-structured data and has a number of characteristics that have caused it to be widely adopted as a data representation format. XML is extensible, platform-independent, and supports internationalization by being fully Unicode compliant. XML is the common language for the .NET framework.

When a mobile agent is deployed using Web Services, XML will ensure that agents can be interoperated across machines and platforms. Moreover, *Common Language Specification* (CLS) under .NET environment provides a Java-like virtual machine which can confine the running of a mobile agent within a particular memory space and thereby protects the system from a hostile agent. (Wigley and Roxburgh, 2002)

OVERVIEW OF THE PROPOSED FRAMEWORK

Within the framework, there will be four groups of computers which communicate and collaborate amongst themselves using XML on top of TCP/IP protocol. These four groups of computers are:

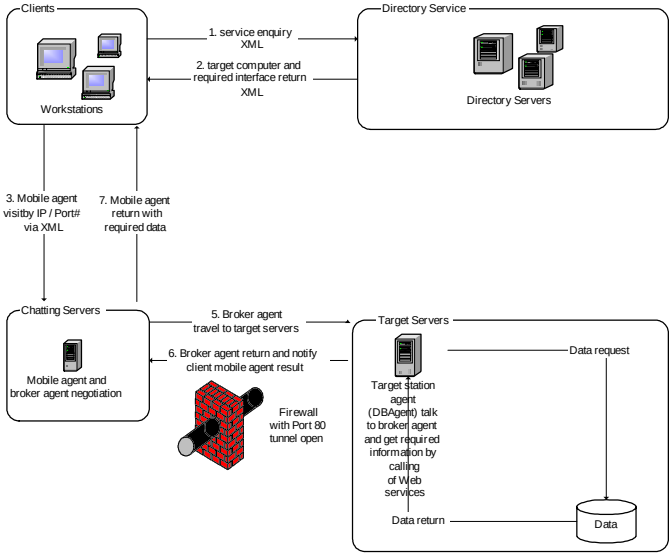
- Client workstation
  A client machine can be a stationary desktop, laptop or a handheld device such as Pocket PC. A user can use one of the client machines to launch a mobile agent.

- Directory Services Server
  These are a group of servers (or clusters of servers) which offer the Universal Discover Description and Integration (UDDI) service for Chatting Servers discovery. Companies can 'adver tise' their services on the server for any mobile agents to visit and discover the services that they offer, and the interfaces that are required to access the services.

- Chatting Servers
  These servers offer chatting services which provide a chatting room where an incoming mobile agent and a broker agent of the corporation can negotiate. After authenticated the incoming agent and validated its requests, the broker agent will migrate to the Database Services servers located behind the corporate firewall for the requests to be processed.

- Database Service Server
  A stationary agent at the Database Service server will receive the request from a broker agent and retrieve the required information from the corporate database and hand them over to the broker agent for passing on.

Figure: 4 Mobile agent and .NET environment



THE ARCHITECTURE FOR THE PROPOSED FRAMEWORK

Figure 4 presents the proposed framework. This framework is based on the research by Xue and Li (2002) but is now extended to take advantage of the .NET framework.

Under the framework, a mobile agent is activated by its owner and launched from a client machine (Home). The mobile agent travels to a Directory Server for directory consultation. The Directory Server will provide a list of possible target machines' locations in terms of IP addresses, port numbers and whatever interfaces (including parameters required) that the mobile agent needs to access a particular service at the remote machines. The client agent will then return home and prepare for sending itself to visiting the target corporations. At this point, the owner may be required to supply further information if the mobile agent is programmed in such a way.

The mobile agent will then travel to a target corporation. At the corporation, a Chatting Server will accept the mobile agent. The Chatting Server provides a chatting room service that will facilitate XML negotiation between the visiting mobile agent and a resident broker agent. The broker agent will authenticate the visiting agent for its identity, its owner's identity, validation of the supplied interfaces and the rights to request information. After the authentication and validation processes, the broker agent will migrate itself through the firewall to a Database Server. In the Database Server, upon requested by the broker agent, a stationary agent (DB agent) will invoke the appropriate Web Services to retrieve the required information from the corporate database and hand over the information to the broker agent. The broker agent will then moves through the firewall and hand over the acquired information for the visiting agent to return home.

FURTHER DISCUSSIONS ON THE PROPOSED FRAMEWORK

Discovery Services

Under the framework, the Directory Service utilizes the Universal Discover Description and Integration (UDDI) (Refer to Figure. 2). UDDI services act like the Yellow Pages. For example, a user can use the Yellow Pages to find a suitable hotel along St. Kilda beach. Under the framework, the owner will dispatch a mobile agent who will have the intelligence to consult UDDI services to find which hosts (hotels) it should contact to achieve the objective of finding the best hotel. UDDI services are just like Yellow Pages where businesses can advertise and increase their business opportunity by posting the details of their servers, their services and the interface required to access the required

information. With regards to the interfaces, a UDDI service can also provide a mapping of functions across businesses. For example, when a user submit *getprice()*, it may map to *getfare()* in Qantas airline and map to *getrequiredprice()* in Cathy Pacific airline.

On the server which provides the UDDI services, a database or catalog will be maintained by a database administrator to provide function mapping. Other details in the database such as IP address and communication port numbers will be maintained and updated by the corresponding subscribed business partners.

Negotiation

A Chatting Server will provide a negotiation environment for an incoming mobile agent and a resident broker agent. For Example, in the real world, a customer visiting a company will need to go to the reception. He/she will have to produce his/her identity and announces to the receptionist who he/she wanted to visit. The same principal should be applied in our model. When a mobile agent visits a corporation, it will only be accepted into a Chatting Server- where the resident broker agent will check its identity, whether it is friendly or not, and whether it has a valid request in terms of the function calls and interface that it acquired from a UDDI server. When everything is in order, the broker agent will then move through the corporate firewall and present the acquired XML request to the database agent at the Database Server. The stationary database agent will retrieve the requested information for the broker agent to pass them onto the visiting agent which is waiting in the Chatting room.

Security and Scalability

The Web Services will be used both in the chatting room and at the database server.

The separation of the chatting room from the database server using a firewall provides the required security and preventing hostile agents causing damages to the mission critical corporate system and protecting important corporate data. Further security can be enhanced by applying encryption for all data exchanges and utilizing a Secure Sockets Layer protocol (SSL) (Dournaee, 2002).

Further services

Under such circumstances, a group of client agents, which possess similar knowledge and with the same intention to do business can come together and negotiate. For instance, a group of client agents may have different air ticket prices after negotiating with different broker agents at different target servers. They can exchange their information and then pick the best price and return home to report to their owner. Alternatively, the client mobile agents can conduct an auction section before returning home.

The chatting room can be hosted on an independent server and a corporation can have multiple chatting rooms. Since this framework is a multi-tier system, scalability and load balancing can be easily achieved by adding additional hardware resources without affecting the existing infrastructure.

CONCLUSION AND FUTURE RESEARCH

With the growing popularity of handheld devices and mobile technologies, mobile agent technology provides a unique solution to address the transient connection nature of mobile devices. Microsoft .NET framework supports both user mobility and hardware mobility, but the support for mobile agents is missing. The proposed framework should provide researchers and enterprise solution providers the missing framework to support mobile agents within the Microsoft .NET environments. The proposed framework takes the advantages of Web Services and other non-proprietary communication protocols and services to provide functions such as mobility, security, inter-operability, reliability and automation for the mobile agents.

The proposed framework should provide an infrastructure for future research such as:

1. Deployment of the framework using Recursion Voyager ORB,
2. Formulation of a specification for agent negotiation by stipulating a common interface and the rules governing its mapping to third party interfaces for the services offered by the discovery servers as outlined in the proposed framework
3. Validation of framework by implementing the framework in a real industrial situation.
4. Investigation into the possibility of incorporating the new components of the .NET 2003 (such as .NET *remoting* and mobile information server) into the proposed framework.

REFERENCES

Basiura, R. (2002) Professional ASP.NET Web Services with VB.NET, Wrox Press p.21-23.

Chappel, D. (2002) Understanding .NET - A Tutorial and Analysis p.44-65.

Dournaee, B. (2002) XML Security, RSA Press p.227-229.

Lange, D. B. and Oshima, M (1999) Seven Good Reasons for Mobile Agents, Communications of the ACM, Vol. 42, No. 3, March 1999

Myerson, J. H. (2002) Web Services Architectures, How they stack up, Tect Press, p.5-13. http://www.webservicesarchitect.com/content/articles/webservicesarchitectures.pdf, Accessed on 16/8/03

Skonnard, A. (2002) The Birth of Web Services, Microsoft MSDN http://msdn.microsoft.com/msdnmag/issues/02/10/xmlfiles, Accessed on 10/7/03.

Vinaja, R. (2001) Mobile Agents, Mobile Computing and Mobile Users in Global E-Commerce, IRMA 2001 Conference Proceedings, IRMA International Conference 2001, p.173-175.

Waterhouse, M. (2002) Web Services Architect Review http://www.webservicesarchitect.com/content/articles/mark04print.asp, Accessed on 23/7/03

Wigley, A. and Roxburgh, P. (2002) Building .NET Applications for Mobile Devices, Microsoft Press p.11.

Xue, F. and Li, K.Y.R. (2002) Java Mobile Agent a*nd Project Management*, IRMA 2002 Conference Proceedings, Information Resources Management Association International Conference, 2002.