Intelligent Software Agents - a .NET Example

Adrian Kosmaczewski

2006-05-25

Software Example

In the February 2006 issue of MSDN Magazine (http://msdn.microsoft.com/msdnmag/), Matt Neely describes a .NET implementation of mobile agents:

> "The term agent originates in artificial intelligence and describes a logical entity that has some level of autonomy within its environment or host. A mobile agent has the added capability to move between hosts. In a computing context, a mobile agent is a combined unit of data and code that can move between different execution environments."

(Neely, 2006)

The idea described in the article is that of a small family of .NET classes that literally "jump" from a computer to another, performing tasks in the host computer, through a mechanism called Remoting:

> "An example of a traveling agent app could perform operations control. An agent is sent out with a list of machines on the local network it should traverse to inventory hardware and software (…) built-in services that facilitate the componentization and mobility of code, namely object remoting and serialization.(…) Mobile agents have their uses and their pros and cons. The autonomous and mobile nature of mobile agents can lead to reduced network traffic, decentralization, increased robustness and fault-tolerance, and easy deployment."

(Neely, 2006)

(Source: Neely, 2006)

Such an agent could be used for administrative purposes (for example, a system administrator could use it to perform complex tasks directly in the target host computer), for inventory purposes (using WMI, such an agent could "collect" information about the computer), and even, and most dangerously, to inject viruses or trojans (such a mechanism could open security backdoors to external attacks in a computer!).

The author goes on describing three archetypical use cases for such agents:

- A simple travelling agent ("Travelling Pattern");

- A task agent ("Task Pattern", handling distributed workload in a network, similar to the approach used by the SETI@home project);
- A buyer/seller scenario ("Interactive Pattern") with a buyer that looks for the best price in several sellers running in different computers (this example shows the biggest complexity).

For more information about .NET Remoting, please refer to this article: http://msdn.microsoft.com/library/en-us/dndotnet/html/hawkremoting.asp. For the moment, suffice to say that Remoting allows to instantiate and execute objects and methods in computers connected by any communication medium, in a protocol-independent way (that is, using HTTP, TCP, etc). For instance, the interaction with remote SOAP web services can be seen as a particular type of Remoting.

Comparison

In the case of the hardware agent described by Brookshear, he describes the following characteristics as those defining an intelligent agent:

- Rational reaction to stimuli
- Intelligent behavior
- Learning capabilities

Let's see how the software example of the MSDN Magazine article compares to a hardware agent in these three specific fields.

Rational Reaction to Stimuli

"An agent is a"device" that responds to stimuli from its environment. Most agents have sensors by which they receive data from their environments. Example of sensors such include microphones, cameras, and air sampling devices."

(Brookshear, 2005)

In the case of the software agent described in the article, the "sensors" are the complete API exposed by the .NET Framework. The agent can use any of the methods exposed in it to get information about the host where the agent is being executed. For example, in the case of the task agent (second example), the agent gets the list of logical drives in the current host computer:

"My child agent will merely list the system's logical drives (with a call to Directory.GetLogicalDrives) and again output the results to the host's console".

(Neely, 2006)

Intelligent Behavior

"The goal of artificial intelligence is to build agents that behave intelligently. This means that the actions of the agent's actuators must be rational responses to the data received through its sensors."

(Brookshear, 2005)

In the article, the author describes the “Interaction Pattern” of a buyer/seller scenario:

“MyBuyingAgent (…) is a bit more complex as it needs to maintain more state. The buying agent is given the name of an item to buy, a value for the maximum amount it can spend for that item, and an array of host URLs that the agent needs to visit in order to find the item at the lowest price. As the buying agent travels, it will need to enumerate each seller on each host, determine if that seller has the item desired, and then find the item’s price. Since the agent wants to find the best price, it will have to track which seller it is going to buy from. I call this the winning seller. The agent will have to store the location of the winning seller, the ID of the winning seller, and the price the winning seller is offering for the desired item.”

(Neely, 2006)

Here we have clearly a description of an intelligent behavior of the buyer agent, following the description given by Brookshear.

Learning Capabilities

“In some cases an agent’s responses may improve over time as the agent learns. This could take the form of developing procedural knowledge (learning”how”) or storing declarative knowledge (learning “what”).”

(Brookshear, 2005)

This last characteristic is not explicitly mentioned in the article, and with a good reason; it is by far the most complex of all, and would require a more complex design. The author of the MSDN Magazine article has purposedly kept the article simple, to introduce the major concepts and the general idea.

Conclusion

The article in MSDN Magazine does not delve into the learning mechanisms, that would make this “agent” a complete one in the AI sense. It rather focuses more into communication and security issues, giving at the same time the pros and cons of such an implementation.

My conclusion is that, given the proper learning instructions, the software agent would be fully compliant with the requirements set by the AI discipline.

References

J. Glenn Brookshear, “Computer Science, An Overview, Eighth Edition”, ISBN 0-321-26971-3, Addison Wesley, 2005

Matt Neely, “Write Mobile Agents In .NET To Roam And Interact On Your Network”, MSDN Magazine, February 2006 (Available here:

http://msdn.microsoft.com/msdnmag/issues/06/02/MobileAgents/default.aspx
- Accessed March 18th, 2006)