Towards the Use of Mobile Agent Based Message Systems

Giacomo Cabri, Luca Ferrari, Letizia Leonardi
Dipartimento di Ingegneria dell’Informazione – Università di Modena e Reggio Emilia
Via Vignolese, 905 – 41100 Modena – ITALY
E-mail: { cabri.giacomo, ferrari.luca, leonardi.letizia }@unimo.it

Abstract

Communication is the base mechanism for coordination and collaboration in human activities, and it plays a fundamental role in enterprise environments. However current communication systems are not based on the message content, but they focus on the message transport and delivery. Furthermore the presence of several communication protocols and tools leads to a heterogeneity in communications, which can lead to difficulties in activity coordination. This paper presents an approach for communications based on the use of mobile agents, which can realize content-filtering, and can unify several message systems in a single one, leaving the user free from system details and able to interact with more than one message system at the same time.

Keywords: communication, mobile agents, messages.

1. Introduction

A lot of human activities require collaboration among involved parts in order to synchronize and better manage the task execution. Examples can be found in enterprise tasks, where often a big task is divided into smaller tasks, each one delegated to a different office or section of the enterprise itself. In such a scenario each involved part (e.g., each office) must coordinate with the others to achieve the common goal (i.e., the task execution).

Typically coordination and collaboration are achieved through communication, which means an exchange of data and information. While standard analogical communications, such as phone or fax, have done this work in past years, the common trend today is to exploit digital communications, with particular regard to computer assisted ones. In fact, thanks mainly to their fastness, easiness and the wide-spread use of computers and broadband networks, computer-aided communications are more and more powerful and used. Computer based communications allow enterprises to shrink down communication costs, while users can communicate with more than one person per time in a quite easy and immediate way. Furthermore computer aided communications allow users to exchange not only text messages, but also other kind of data such as documents, images, and more in general files, making the collaboration more efficient.

There have been a lot of communication systems so far, such as e-mail, instant messaging (IM), chat, file sharing, etc., and each user tends to use her favourite one. Even if the most important communication systems follow the standards, the fact that each user prefers one of them instead of another (e.g., e-mail instead of IM) produces an heterogeneity in communications. Furthermore the use of different messaging tools and clients between the sender and the receiver can produce a different visualization of the message (and of its content), leading to possible misunderstandings.

A communication system should allow users to worry only about message content, and not about protocols, tools and how to show the message itself. If users can concentrate only on messages, reaching a document-centric approach instead of an application-centric one [ShnP], their communication will be more efficient. In this direction, a lack of all actual digital communication systems is the incapability to distinguish among addressees of the same message. This means that each addressee will receive and see the same message content of each other one. This can lead to a waste of time in enterprise communications, since not all involved parts must receive and see the same information of others, thus several different messages must be produced (one for each class of addressee) and sent. To be more efficient the communication must allow the sender to prepare a single copy of the message, showing needed information to each addressee depending on its role inside the enterprise. Even if there are already computer based approaches which adapt to the human role inside the enterprise to better fits her requirements (see for example[ShnP]), they do not cover directly the communication aspect, that we believe it is probably the most important.

Starting from the above considerations, this paper proposes a communication system based on mobile agents, where all the capabilities of mobile agents can be exploited to enhance current communication systems in order to achieve a smarter one. Our approach changes the view of message, which becomes an active entity, able to proactively decide what to do with its content and how to show it to the final user. Furthermore our approach is compatible and can be integrated with existing ones, so to ease its exploitation by users.

This paper is organized as follows: section 2 shows the motivations and the advantages of the use of mobile agents in communication systems, section 3 details how our approach works, section 4 shows related work and finally section 5 reports conclusions.

2. Exploiting Mobile Agents as Base for Active Messages

Agents are autonomous, problem-solving, adaptable computational entities which can perform their task(s) in open and dynamic environments [LucMP03]. The most important characteristics of agents is their
autonomy, which enables them to perform actions without requiring a continue user involvement. Furthermore agents can be adaptable, that means they can react to execution environment changes recalling autonomic computing [IBM] development paradigm. Agents can be exploited to deal with complex system [Jen01], thanks to their characteristics. Mobile agents are a particular kind of agents, which extend the above definition adding the mobility, which is the capability to move from one host to another in a network. Mobility allows agents to scout the network, searching for information and data, acting as active network entities.

Our approach exploit the three main capabilities of mobile agents, autonomy, adaptability and mobility, to realize active messages, that means messages which are shipped with executable code (the agent itself) to display themselves. There are already messages that are delivered within active code, for example with JavaScript or VBScript snippets. Our approach proposes a different way to deliver messages with code: while current messages can be delivered only with code inside them (Figure 1 a), our approach proposes instead pieces of code (agents) which contain the message and deliver it. In other words, as shown in Figure 1 b, in our approach the agent embeds the message itself and is in charge of delivering it, executing some actions (such as showing) on it. Please note that, while current message systems can delivered both messages with or without code in them, our approach always delivers messages with code around them.

We believe that putting the code around the message can lead to a set of advantages, and the next subsection gives more motivations of our approach.

Figure 1. a) a standard message with code in it; b) our approach with message in the code.

2.1. Why Code Around Messages?

There are two main advantages using code around the message. The first is related to the message content showing, which can be totally controlled by the code; the second is related to the shipping mechanism, which again can be controlled by the code itself.

We remark that it is important to ease the communication so that final users can use the system to coordinate their activities without worrying too much about information displaying. Imagine for example that, in an enterprise, there is an office O1 that needs to order a thousand pieces of product X, which will be used as raw material for another product. What happens is that the office O1 sends a message to the purchase office (O2) asking to contact the supplier for that material. At this point, office O2 contact the supplier and fixes the price, then sends a confirmation message to the office O1, which does not have rights to know what the buying price is. At the same time the office O2 sends a message to the enterprise director saying that the enterprise has bought a thousand pieces of X at the price P. With current messaging systems, the only way to display different information on the base of the receiver is to compose different messages. This means that office O2 is in charge of sending a first message (without prices) to O1, and a different message (with prices) to the enterprise director. In our approach, instead, this can be simplified, since office O2 can write a single copy of the message, with all information, letting the code that ships the message (i.e., the mobile agent) decide if prices have or not to be shown to the receiver (see Figure 2).

Please note that displaying message content depending on the addressee is different from what happens with today’s message clients, which allow users to set their preferences so that a message can be displayed, for example, with highlighted colors if it is important or comes from a certain sender. While these systems cannot perform a message content selection as the above one, our approach can do this and can also display a message following user’s style preferences (colors, fonts, etc.). It is thanks to its adaptability that the agent can behave and display the message differently depending on the addressee.

The above example emphasizes how, with our approach, it is possible to write a single message content, which can be shown in different ways and behaviors, thus simplifying the communication. Furthermore the fact that our approach enables users to deal with a single message content, displayed in different ways, makes the communication more coherent, since no possible wrong (i.e., incoherent) copies of the same data (i.e., the message content) exists.

It is time now to get a closer look at the code that our approach puts around the message: why should that code be a mobile agent? First of all it is possible to note how mobile agents are apt to this kind of use, since they are mobile code and can carry with them information, such as a message. The use of mobile agents for message systems provides several advantages, thanks also to their autonomy, but the most
important one is the independency of the application protocol. Until now we have not detailed a particular message system, such as e-mail or instant messaging. The reason is that our approach can be used to support any kind of communication system. In fact, since the transport is delegated to the agent itself, each kind of communication protocol can be developed on top of it. It does not matter if the message an agent is carry with is an e-mail like message or an instant message one, since in both cases an unified protocol will be used. For example, we can use as underling protocol the Agent Transport Protocol (ATP) [ATP].

The use of an unified transport mechanism leads to the advantage that communication-involved parts can use different communication systems (e.g, e-mail and IM) with the same underlying protocol, which means that the communication system is the same (an agent based) simply used in different ways. This is more prone to a document centric approach, where applications are not important as document content (in this case documents are messages) [ShnP]). Users do not worry about application details, but only about document/message content, and this reduces the waste of time and efforts in communication producing a more efficient collaboration. Our approach achieves this thanks to the use of mobile agents, which embeds details about the use of the message they are shipping by a one-size-fits-all protocol, leaving the user free to worry about the content only.

Finally, the use of mobile agents realize the pure push technology [Chi01], since the user is not asked to open a client or a messaging tool to receive messages. In fact, in our approach, messages do reach autonomously the addressee user, so that it does not have to worry to search (i.e., receive) a message; the message will be delivered to her automatically.

### 2.2. Another Communication System?

In the previous section we have detailed advantages of our approach, which come from the application of autonomy and mobility of agents to messages. However, since there are already a lot of communication systems, it is important to clearly present the role our approach can play in this scenario. If our approach provides advantages to existing communication systems, we must take into account that a communication system must be usable by the final user, else the communication, and thus the collaboration, will not produce results but waste of time and resources.

As already written, each user tends to use her favorite communication system (e.g., e-mail) and her favorite tool (e.g., a specific e-mail client) as interface to the system itself. Instead of proposing a stand-alone system, which means a new communication system with its own tools and procedures, we propose to integrate our approach in currently available ones, so to make softer the transition from an existing communication system to our approach. This means that, for example, there will be plug-ins for e-mail clients so that agent-based messages can be displayed in the user’s favorite e-mail client. Similar considerations can be done for other tools, such as those for instant messaging, chats, etc.

The fact that, as described in section 2.1, our approach is document centric and exploits an unified transport protocol, allows both users and developers to exploit appropriate plug-ins. Plug-ins can be embedded in several tools, disregarding their nature and their purposes, leaving sender users to simply concentrate on the message content, and addressee users to simply display that content independently of their messaging client. Furthermore, since the displaying code can be carried with the message itself, the message can be displayed also on those hosts which do not have the appropriate client. This can result very useful for example on smart devices, such as PDAs, where due to memory limitations, not all applications can be installed.

The use of plug-ins for existing clients and the fact that messages can be shipped with code to display it, will cover the today’s heterogeneity in message clients, allowing for example a user to write a message within an e-mail client, while a receiver will see the message within its instant messaging client and another receiver can successfully display it on its PDA even if without any client (see Figure 3). This will enhance the ways through which human coordination and collaboration are established.

![Figure 3. Messages can be displayed stand-alone or within a client.](image)

### 3. How Our Approach Works

This section gives an overview of our whole approach, in order to describe how it works.

Since our approach is based on mobile agents, it is required to enable mobile agents on each final user host, in order to allow the exchange (i.e., send and receive) of mobile agents, and thus of messages. Mobile agent execution is supported by a mobile agent platform (platform henceforth), which is an environment in charge of managing agent execution, resource access and migration. The platform protects
the host from malicious agents, which cannot execute arbitrary operations (e.g., deleting files) without the platform allowance; furthermore the platform supplies mobility facilities to agents, thus they can migrate from one host to another simply requiring the service to the hosting platform.

Since in our approach the message must be shipped by an agent, the first step is to prepare the message and to create an agent to transport it. This means that, as shown in Figure 4 a, the user creates the message, using for example a text editor or a plug-in for her messaging client. The user can also specify a set of rules for the message, in order to decide which addressee can view which part of the message content. The message and the rules (if any) are then passed to a particular agent running in the platform: the message manager agent. This agent is a “fixed” agent, that means that it does not migrate to another platform/host, and is in charge of waiting for outgoing messages from the user. For each outgoing message, the message manager agent creates one delivery agent, that is an agent in charge of delivering the message (Figure 4 b) to a single address. The message manager agent informs then the delivery agent about the message addressee and rules to be applied on the message. After this the delivery agent can init its trip to deliver the message.

![Figure 4. The first step of our approach.](Image)

Please note that a delivery agent is created for each address the message has to be delivered, that means that each addressee will receive a copy of the message.

Once the delivery agent is ready to ship the message, it migrates to the addressee platform, where it can show the message to the addressee accordingly with the specified rules (see Figure 5). If one of the involved users uses a plug-in, there is no need of a whole mobile agent platform, but of a minimal one embedded in the user’s tool. For example, if the addressee exploits a plug-in for an e-mail client, the delivery agent is received directly by a service running within the e-mail client, which will show then the message as an integrated e-mail. Nevertheless, the code required to show the message will be embedded always in the delivery agent.

Figure 5 shows a direct connection among the sender platform and the receiver one, that means that the message delivering happens through a peer-to-peer approach. This represents another advantage of this solution, since peer-to-peer does not require a central server, which can collapse if the number of exchanged messages grows up. Instead, using peer-to-peer, each host involved in the communication is loaded proportionally to the number of messages it is exchanging with other hosts. Peer-to-peer is already exploited in several instant messaging clients (see for example [Vyp]), but not in other kinds of communication, like e-mail. Since, as already written, our approach does not distinguish among the type of communication (e-mail instead of instant messaging), we can also apply peer-to-peer to e-mail. However peer-to-peer is not always usable, in particular if the addressee host is not available. While this is not a problem in instant communications, since simply the message is not delivered, it can be a problem in delayed communications such as e-mail like. Current approaches store temporary messages in a server (e.g., POP servers), until the user decides to connect to this server downloading (i.e., receiving) all messages.

![Figure 5. The delivery agent delivers and shows a message.](Image)

To solve the above problem, our approach exploits a mechanism similar to temporary servers: first of all we group several hosts into small domains, and then we supply a service host with a running platform, called domain master. A domain can be a group of correlated hosts, such as those in a department of an enterprise or those belonging to the same research group. Each time the receiver platform is unavailable, the delivery agent is sent to the domain master, where waits until the receiver is on-line again, and then reaches the receiver host as detailed above. This solution implies that the delivery agent has the capability to find the domain master on demand and to know when a receiver platform is ready again. Both above capabilities can be achieved by use of broadcast messages to synchronize the status of each platform. When the sender platform understands that the receiver platform cannot be directly contacted, it sends a broadcast message in the domain asking for the domain master address (Figure 6 a). The domain master replies and, starting from now, each message directed to the above receiver is sent to the domain master. When the delivery agent reaches the domain master, it waits (i.e., suspends itself) until a broadcast message from the receiver platform comes; this message indicates that the receiver is now on-line. The latter message is sent as broadcast so to inform sender platforms to send (again) new messages directly to the receiver, instead of sending them to the domain master. When the domain master receives the above broadcast message, reschedules each delivery agent waiting for the receiver to come on-line, and then each agent moves to the receiver (Figure 6 b).

Splitting hosts into domains can lead to the need of delivering a message to an host external to the domain of the sender, in this case the domain master of the
addressee domain is contacted to store the delivery agent.

Figure 6. Use of a domain master.

Until now nothing has been written about addresses. The use of mobile agents and peer-to-peer requires that each platform has an unique known address. Since our approach does not distinguish between different communication types, the user have a unique identifier for each kind of communication. Thanks to this unique identifier, users can concentrate on the communicating counterpart (i.e., the message addressee), without taking into account the communication mechanism and, thus, the service required to “connect” to the addressee. For example, if user A wants to communicate with user B, which only receives e-mails, he must know the e-mail identifier of user B. If A then wants to communicate with user C, which unlike B receives instant messages, he must know the instant message identifier of C. This leads to a heterogeneous situation, since A is not thinking communication in terms of the addressee, but in terms of a couple addressee and service to reach him. Our system is simpler, since each addressee has a unique identifier for each kind of supported communication. In other words, senders are addressing an unique identifier, which means are addressing the receivers without worrying about communication details. This becomes particularly true when the same user has more than one platform, for example one at office and one at home (or on its PDA). Since agents are autonomous and able to move spontaneously, a delivery agent can decide to move from the office platform to the home one if the message appears to be unread, since it could mean that the user is no more at office but she is at home. In this situation the final user does not worry about receiving messages, since messages can reach her autonomously (depending of course on her wills), following user movements.

3.1. Other advantages

This section briefly shows other minor advantages of a mobile agent based communication system. All advantages are mainly thought to reduce time and effort waste during communication.

A first advantage can be the use of a single delivery agent, that is a particular kind of delivery agent, used for messages with multiple addressees, which deliver messages following a particular sequence. Sometimes it happens that a message must be viewed by a person, and only after it can be shown to other people. Typically it happens when the message contains a request to approve of. This situation can be managed easily by our approach: once the sender has detailed the sequence of approval, the message manager creates a single delivery agent for the message (instead of one per addressee). The above single delivery agent then moves to the first addressee, waits until she gives her approval, then moves to the second addressee, and so on. This reduces the number of messages required to synchronize a sequential task and does not require a particular involvement by the sender.

Another advantage of the use of mobile agents can be found in public boards, such as forums, newsgroup and so on. These are typically used to publish questions about certain problems, waiting then for a reply from someone. Instead of sending her own message, polling the board for answers, the user can ask to the delivery agent to automatically come back home with answers, when they are published. Once again it is emphasized the different nature of this kind of communication from the current ones, which does not require users to check for messages, but allows messages to reach users.

Since each message is embedded in an active code (the agent), message threads, such as those coming from newsgroup or individual discussions, can be managed by cooperating agents, so that each message belonging to the same thread is treated in the same way. This means, for example, that all messages belonging to a thread related to a certain research group of a factory, can be displayed with highlighted colours, etc.

The fact that the agent can carry with code tied to the messages introduces advantages also for cryptographic uses. In fact, the agent can transport an encrypted message, decrypting it at the destination and displaying it. This does not appear too much different from what can be done, for example, with PGP/GPG keys [Gpp], but the real innovation here is that there is no need to know the addressee key or, more in general, to establish a common cipher technique/algorithm between the sender and the addressee, since each required information, such as the decryption algorithm, can be transported within the delivery agent. In other words, the addressee does not worry about the cipher algorithm used, since it is the agent that decrypt the message itself.

Finally, the fact that agents are autonomous and can move spontaneously, means that routing information can be determined at run-time, depending on the network status. This means that the delivery agent can be programmed to, for example, always search for the cheapest trip to the destination. Furthermore, if the delivery agent finds some network nodes down (e.g., due to a crash), it can decide a different trip to the destination, implementing thus a more reliable
transport system, able to recover from network problems.

4. Related Work

An important work on human-to-human communication through mobile agents is Flyngware [Chi01]. Flyngware exploits mobile agents to deliver e-mails through the SMTP protocol. E-mail messages built with Flyngware are active, since they are delivered by a mobile agent, as in our approach. Nevertheless Flyngware is related only to e-mails, while our approach is more general and does not exclude other communications. Furthermore our approach is more general, since it exploits a protocol, ATP, which is not tied to any type of communication and, thus, can be exploited in each system.

Another relevant work is that in [BanP], where mobile agents are used as messenger for delivering messages between hosts of an ad-hoc network. This work has demonstrated how mobile agents can be useful to face network varying situations, deciding dynamically routing trips to reach the destination. Nevertheless, unlike our approach, this work uses mobile agents only as “messenger”, without using their capability to display messages. Furthermore their approach is not directly related to human communication, and is studied in dynamic networks as ad-hoc ones are.

In a previous work, we have already implemented a system based on mobile agents to help human to set up their communication system [CabLF03]. In this approach a mobile agent is in charge of registering an e-mail account for its owner user, configuring then her e-mail client in order to allow her to use the new account. Even if this approach is not directly related to the communication mechanism, since it involves agents only in the beginning phase (i.e., the communication account setup), it shows how mobile agents can be used to help human activities and communications.

5. Conclusions

Communication is the base mechanism for coordination and collaboration in human activities, and it plays a fundamental role into enterprises. The heterogeneity of communication systems and tools can lead to difficulties in communications, since each user tends to use the system that likes and better understands.

This paper has presented an enhanced communication system based on the use of mobile agents, which can overtake limitations and difficulties present in current systems. We believe that all advantages of exploiting the feature of agents, such as autonomy and mobility, can be applied to build a communication system that frees the user from communication details, such as the communication type. This leads to a situation oriented to document content and not to applications, where users are in charge only of defining the message and its content, without worrying about which application and protocol have to use to send/receive it. This thanks also to the possibility of using plug-ins which will allow users to continue using their favourite message client while interacting with our system.

The fact that agents can be used to build active messages, with content-filtering based on the message addressee, make this approach more interesting, especially for collaboration among enterprises and between their employees, since it simplifies the communication required to coordinate activities.

Furthermore, since agents can act proactively, they can adapt to network changes, in order to implement a more reliable system able to face network problems.

Thanks to the above advantages, our approach is very flexible and can be used to make communications more efficient reducing the time wasting of final user.

We are currently evaluating the implementation of our approach exploiting the DIET [HoiW] mobile agent platform. We have chosen such as platform because it is a platform able to support the simultaneous execution of a lot of agents, and this is a fundamental requirement for our approach, since the number of messages a person is sending/receiving can grow very rapidly. In order to improve the usability of our approach we are studying plug-ins for the Kmail e-mail client [Kml] and the Kopete instant messenger [Kpt].

References


[IBM] IBM : “A u t o n o m i c  C o m p u t i n g : IBM’s Perspective on the State of Information Technology”, manifesto available at www.ibm.com/research/autonomic


[Kml] The KMail e-mail client, web site kmail.kde.org

[Kpt] The Kopete instant messenger, web site kopete.kde.org


[Vyp] The Vypress Instant Messenger, web site www.vypress.com