Communications Systems Driven by Software Agent Technology

A. L. G. Hayzelden, J. Bigham, S. J. Poslad, P. Buckle, E. H. Mamdani

Department of Electronic Engineering,
Queen Mary and Westfield College,
University of London.
London, E1 4NS, UK.

Email {a.l.g.hayzelden, j.bigham}@elec.qmw.ac.uk

Department of Electrical and Electronic
Imperial College of Science, Technology and Medicine, Engineering,
London, SW7 2BZ, UK.

Email {s.poslad, e.mamdani}@ic.ac.uk

Advanced IP Services and Management,
Nortel Networks,
Harlow, Essex, CM17 9NA.

Email pbuckle@nortelnetworks.com

Abstract

The application of software agent technology to the management of communications' infrastructures is a challenging domain as it requires management on different time scales and has many interacting components. This paper looks at the potential benefits that may be gained from the application of agent technology to communications systems and surveys recent developments. Recent work using distributed network management by adopting co-operating and self-interested agent models of collaboration are described. The paper provides an introduction to the authors' perception of agent technology, followed by a discussion of some issues that need to be addressed for agent technology to be of practical use in the communications domain. Following sections describe how agent technology has been used for network management, legacy telecommunications systems, and telecommunications integration. The authors discuss how the emerging standards for agent technology can be used in an applied situation of providing a virtual private network. Latter sections describe mobile agent technology and some of the research that has adopted the mobile agent paradigm for fully distributed network control.

Key Phrases: Network Management, Agent Technology, Intelligent Agents, Virtual Private Networks and Mobile Agents.

1 INTRODUCTION

Telecommunications systems are entering a period of dramatic change where the next generation of telecommunications systems will need to support many more flexible services than users currently experience. Some of this change has been initiated by the impact that the Internet technologies are placing upon telecommunications providers, for example the ability to support voice communications, the ability to listen to stereo radio transmissions, and even capability for video conferencing support. Even though the deployment of Internet Protocol (IP) is virtually ubiquitous, there are technologies that were originally designed to directly support the types of services mentioned. IP was originally deployed due to its inherent robustness trait, which comes from its connectionless mechanism for communication establishment. However, technologies such as Asynchronous Transfer Mode (ATM) were originally conceived on the connection oriented paradigm and the premise that in the future, people would want services that would require guaranteed Quality-of-Service (QoS), something which, is currently proving particularly difficult to address in the IP domain. In addition to the technologies necessary to implement the communications protocol infrastructure there is the need to manage complexity issues and also to provide a scalable approach to integration of legacy communications systems. For example, another important factor in achieving inter-operability of transmissions networks is how to maintain co-ordination of the distributed system, i.e. network management.

Network management has traditionally concentrated on the physical level. However, with the advent of technologies such as, ATM [1], IP over ATM [45], mobile IP [URL 1], Intelligent Networks [4], QoS based technologies [54], Active Networks [51], mobile communications and advances in the underlying digital infrastructure [21], network management will need to focus more on the logical level to allow for more flexibility, scalability and improved integration. Other factors pushing the communications industry forward are: competition between an ever increasing number of service providers; move towards a service-oriented or application level approach; increases in computational power; advances in switching hardware; standardization of both software and hardware interfaces; and the bandwagon to provide ‘intelligence’. Decina and Trecordi [10] provide a comprehensive overview of communications and computer based convergence and review many of the underlying technologies, i.e. CORBA [37], Active Networks [51], IPv6 [45], Middleware [38] and TINA-C [URL 4].

In this paper we discuss the impact that agent technology is currently making in the communications field, including how it is used in legacy systems support, network management and how some emerging agent standards will provide a baseline for further agent technology based systems to be deployed. We use the term software agent to incorporate the two most commonly described software components that researchers have used to tackle complex decomposition problems, namely Mobile Agents (MA) and Intelligent Co-operative Agents, often referred to as just Intelligent Agents (IA). See Wooldridge and Jennings [56] for aspects of Intelligent Agent *theory, languages* and *architectures*. Probably the most important attribute that distinguishes agent software from other types of software processes is their ability to act autonomously (i.e., without the direct intervention of a human). The argument being that autonomy can provide system operation with increased robustness, as the agents can deal (to a certain degree) with unexpected event occurrences. The autonomous capability of agents is often achieved by the agent software being embedded with some mechanism for acting in a pro-active manner, which means that the agent tries to achieve high level goals rather than follow a set of pre-defined instructions in a procedural manner. The co-operative abilities of agents are also another important attribute that the agent approach brings to traditional software engineering concepts. Co-operation either allows the agent to gain a local advantage (exploiting the actions of other agents) or to add to the global ‘good’ of the system in which they operate (i.e. via collaboration). We believe that the agent approach recognizes that agents should have some ability to “deal with contingencies in operation”, a factor that is not often considered in many traditional software applications. With the advent of network systems that are owned by more than one party, agent designers will now focus attention on optimizing the individual agent's performance and making sure of its ability to interact in a more generic way, something that agent standards efforts will hopefully provide. For the multi-agent system approach to be most beneficial it has been realized that agents need to interact not only with agents constructed by the same design group but also by others. This is one way that will provide the code re-use ability that has been so instrumental in the uptake of Object-Oriented programming.

The other main contribution that agent systems development brings to the software engineering practice is a paradigm for designing systems at a higher level of abstraction than Object Oriented methodology. The main concept is that systems can be described at a conceptual level of interaction between autonomous entities that have particular run-time roles, responsibilities and commitments, which can dynamically change as the system alters. Arguably, using this approach can model the way human beings interact to solve complex problems in a more realistic manner and thus be a more natural software engineering method for system designers to adopt.

2 COMMUNICATIONS SYSTEMS

We begin this section with a general description of some of the issues and complexities involved in network management and control. Within this context we will detail how the agent approach has been applied.

The need to communicate over long distances, to transfer high definition graphics, the necessity to inter-change media formats and the need to multi-cast communications between groups of people, has meant the Public Network Operators (PNO) have been persuaded to provide more sophisticated network infrastructures to achieve and maintain their customer base. The increase in diverse service provision will cause a similar increase in the control complexity of the enabling system. Therefore, it will no longer be a case of throwing more computer memory and processing power at the problem or having larger databases to hold all of the information, there will be a need for localized, mobile and distributed problem solving to be utilized.

2.1 Communications Models

Traditionally, a telecommunications system comprises of a number of geographically distributed nodes all interconnected in some manner (e.g. star, fully meshed, minimal interconnect topology, etc.) by physical communications media to allow the transfer of bits of data. Of course the transfer of these bits have no semantic meaning unless they are translated into meaningful signals and this ability relies on sophisticated digital signal processing technologies. Adding further to the complexity of the total communications system is the network management system (NMS). The NMS makes sure that all of these interacting technologies carry out their individual tasks in a co-ordinated manner. As we will describe later, NMS were often developed specifically for the supplied hardware system and have therefore commonly relied on adopting the physical system architecture of the transmission system thereby becoming fully embedded software which is unable to be flexibly altered. To add to these issues, the ability to deal with run-time exceptions is not always feasible, particularly in high-speed network systems, (i.e. the time to react often outstrips the cause).

The complexity of inter-working existing heterogeneous systems meant that the computing arena chose a quite different approach, namely the prosperous Internet model. The Internet model is based on the concept of inter-working, where the network hardware is viewed as just a transmission platform and all the functionality lies in the software functions residing on the host servers. This model allows for greatly increased implementation of new services. We and others, including [47], [34], [17] believe that this model will eventually prevail as the enabling communications platform of the future. In later sections we will describe how management solutions from the telecommunications world, coupled with concepts from the Internet and software agent technology could provide a more integrated, easier to automate and more flexible service enabled communications system. Similarly, in the AI domain there has been considerable research on both reactive and planning strategies for dealing with these kinds of problems, but although there has been success in the application of these concepts, network management is becoming increasingly difficult to handle by global centralized mechanisms. Also, adapting to an ever changing environment is a matter that is extremely difficult to handle and cannot be efficiently modeled by *fixed* planning methods or *encoded* reactive measures, however rapid they may be. Therefore, the use of distributed problem solving coupled with AI and pro-activity may help address some of the concerns associated with the control of these complex network systems.

2.2 Network Complexity

Networks range in size and complexity from the small in-house local area networks (LAN) to the millions of nodes which, make up the Internet. The larger the network, the more heterogeneous it tends to be. As network complexity grows, traditional embedded network management approaches will be unable to cope [15], particularly when they scale to world wide communications networks. To tackle this problem the International Standards Organisation (ISO) developed the Open Systems Interconnection (OSI) standards, which are based on the idea that a manager-agent approach should be adopted for the management of complex network systems. The principle idea is that, if this approach is adopted some of the interfaces can be standardized and therefore reduce some of the network heterogeneity.

Network management in general comprises both network monitoring and control. A network management system thus performs, among other things, the tasks of network configuration guaranteeing its proper functioning, maintenance, security control, and archiving of fault management reports. See Stevenson [50] for an introduction to network management. Extending some of the concepts of the OSI network model, Network Management Systems (NMS) have encompassed the ideas of task delegation. Here, management tasks are sub-divided amongst several entities, each being assigned specific tasks, allowing the distribution of processing and the reduction of bottleneck traffic. Unfortunately, many existing network management architectures are based on outdated concepts and due to the cost of re-implementation have not tracked some of the latest developments that enable facilities such as full distribution, conflict resolution and resistance to change, which are provided by OO design and agent technology. However, TINA-C has made significant progress towards supporting modular and flexible inter-working and in some respects replaces the OSI based communications by the implementation of a ubiquitous Distributed Processing Environment (DPE), in some respects this provides a common level for service interaction. The use of services provided by the DPE allows object interactions to occur via well-defined interfaces and complexity transparencies are obtained between the services and the underlying network platforms. The typical transparencies are location, distribution and replication. In TINA the DPE is implemented through the technique of creating a standard Application Programming Interface on all the concerned hardware. This idea is similar to running Java Virtual Machine software on all computer operating systems, and therefore demonstrates how the paradigms from both Internet and Telecommunications domain are converging.

2.3 Network Management Architectures

In the telecommunications domain the decision on whether to have a centralized, hierarchical, distributed or hybrid network management architecture influences whether or not physically distributed agents will be effective or even appropriate. Rapid growth in the flexibility, diversity, and complexity of network systems has fuelled the debate over whether a centralized paradigm is scalable. Unfortunately, polling (passing monitoring information, control information or acknowledgements) from switch to management station imposes a large overhead in the basic physical layer communications mechanism. Relying solely on the centralized approach also means that the system is more vulnerable to system collapse when failure occurs, i.e. there is no localized control policy in action and control occurs via the central management system. However, centralized management provides an effective mechanism for the instance where a small redundant number of entities require control activation. This is because the centralized approach provides benefits in terms of providing global network state information, therefore reducing the probability that inconsistencies or conflicts will arise. Of course, the benefits of this must be weighed against the need to provide scalability.

An obvious disadvantage of the (completely) centralized approach is that it does not allow control data to be processed where and when it is appropriate to do so. In order to enable application and network service diversity further points need to be addressed. These points include whether there is the need:

for global information to be disseminated to local control points.

to provide common management interfaces.

to reduce the expense of geographically spreading computational power.

However, the control of telecommunications networks is inherently distributed, the management system requires the ability to deal with run-time exceptions and be robust under these circumstances. As increases in service complexity arise (particularly multi-media support networks), greater amounts of information need to be processed and partial system failures require rapid rectification; therefore, network management is becoming increasingly difficult to handle by global centralized mechanisms [6]. Having distributed localized expert problem solvers can help automate the management process. For example, by:

- Reducing communications overhead between the management system and the control activation points via a reduction in the need for continuous system state polling and signaling, i.e. higher level communications language is used.
- Distributing tasks to the most appropriate and available computational resource.
- Improving robustness, both in terms of fault recovery (less vulnerable to global system collapse when isolated failures occur) and with dealing with uncertain information.
- Improving response time - localized control can achieve more rapid responses to local exceptions.
- Using Artificial Intelligence (AI) techniques to deal with situations where there is a lack of knowledge about how to develop an optimal algorithm (i.e., when there is no well defined procedure or algorithm to deal with exceptions)
- Improving flexibility in encompassing changes (i.e., extensibility to provide new services). For example, the introduction of new complex services such as, multi-cast video-on-demand.

3 AGENT TECHNOLOGY IN COMMUNICATIONS SYSTEMS

This section will review some work carried out by researchers and projects that have used the agent approach for communications management. We discuss work which was inspired by the new concept of allowing software to be moved from node to node, and also describe research that has begun to focus on providing flexibility through the use of AI techniques.

Network Management is often used to refer to the mechanism that acts in a global manner on many entities and tends to move the system configuration (e.g., reallocation of system resources) from one equilibrium state to another. Management operations normally act over relatively longer time scales. Control is about providing localized fine-grained actions necessary to maintain the system in a stable condition [8]. However, the distinction is becoming blurred [26] with the advent of mobile control entities and localized agent expertise enabling adaptation of node control logic and distributed co-ordination, e.g. where fine grained alterations to node functionality is invoked from a geographically distant location.

Magedanz, Rothernel and Krause [29] have provided a broad taxonomy of agents in the telecommunications domain and identified the following groupings; local agents, networked agents, DAI based agents and mobile agents. These classifications apply to the vast array of work that is commonly grouped under the heading of ‘agents in telecommunications’. However, one of the most promising classifications of using agent technology in communications control is that of adopting the distributed multi-agent systems (MAS) approach. As mentioned previously a central

MAS concept is that of a society of agents sharing information. The problem solving that the MAS can achieve is often determined by the amount of co-operation or collaboration that individual agents actively participate in and to what extent information is shared. There is an important distinction between *co-operative* agents, *self-interested* agents. Self-interested behavior is most appropriate when the agents are developed for competing parties operating in an open environment. Co-operation is often essential in critical system control as the equilibrium operational state of the system must be explicitly known (i.e. network fault control, nuclear power station control, etc.).

There has been a considerable amount of work on distributed planning, scheduling, resource allocation and control problems for co-operative agents. The principle reasons for distribution have been either,

- to reduce communication costs associated with a central problem solver, or
- to gain speed and computational resource through parallelism, or
- to gain reactivity through not needing to consult a central agency, or
- to gain robustness through lack of dependence on a single computational node.

It is becoming understood that the case for distribution of the problem solving in the co-operative agent case is not always strong, e.g. [44], “to date, the arguments in favor of distributed co-operative scheduling systems have mostly been weak”. The arguments certainly need to be examined in the context of network management problems.

End-to-end service management and resource provisioning across a multi-vendor heterogeneous environment is now mission critical for carriers in the competitive open market for network services. In this type of situation the self-interested approach may be more applicable. Agent researchers have focused upon the integration of control and management and have applied economic and game theoretic concepts to resource allocation, e.g. [3]. Economic systems constitute one well-studied class of mechanisms for allocating resources among distributed decision makers [7]. The models and techniques, employed in ensuring that efficient resource allocation is achieved, are a major factor to consider when applying agent technology in the search for solutions to telecommunications problems. There are also many other non-functional, functional, physical and conceptual issues to address, including:

- What communication does the agent have with other agents - try to minimize inter-agent communication so that action resolution is reached more rapidly.
- Where should the agent(s) be located (the agent location granularity)? One agent per node, one per physical link, one to represent each connection?
- Should the agents be stationary or mobile?
- What view of the network does the agent have (the agent control granularity)? Is it beneficial to have the agent aware of other indirectly related events occurring in the agent's locality.
- What degree of dependency should there be between the agents in the system? If an agent ‘dies’, is the system still robust?
- What co-ordination model should be adopted to make sure coherent group behavior is achieved when large and complex interactions occur between distributed reasoning agents.

One management technique that has focussed on some of the issues mentioned above is the Management by Delegation (M_bD) [18] approach. Goldszmidt and Yemini [19] describe the features of M_bD with an emphasis on the relationship to the intelligent agent paradigm. M_bD is a decentralized network management architecture that is consistent with the incorporation of agents in the DAI sense. Each node in the network is able to run distributed script processes. This enables the network management system to consist of “agents” operating at individual nodes that can receive scripts, which specify the tasks to do: hence the term Delegation. As Meyer et al. [31] state “At the highest level of abstraction, the decentralized M_bD paradigm and centralized Simple Network Management Protocol (SNMP) paradigm appear to be the same, as both have a Network Management System (NMS) communicating with agents via a protocol”. However, there is a definite shift in paradigms, where M_bD is more concerned with the delegation of more high level goals than is obtained in the SNMP approach. The delegation of scripts as opposed to simple management messages is often more efficient in terms of bandwidth utilization.

The adoption of this type of model, therefore, leads to the question of how to co-ordinate the control interactions of the many entities that are able to make alterations to the operation of the network system. There is a large volume of research on co-ordination techniques in the agent domain. However, the most influential co-ordination technique for task and resource allocation among a society of agents is the Contract-Net Protocol (CNP) [49]. Its generic nature and robustness is acknowledged by its adoption into the FIPA (Foundation for Intelligent Physical Agents) specifications [URL 3] as a high level co-ordination mechanism for allocating tasks or resources within an agency. In the CNP a market structure is assumed and agents can take on either of two roles, a manager or a contractor. The basic premise of this form of co-ordination is that, if an agent cannot solve an assigned problem using local expertise, it will decompose the problem into sub-problems and try to find other agents with the necessary resources and expertise to solve these sub-problems. The assigning of the sub-problems is solved by a contracting mechanism. This consists of the manager creating a contract that is announced to other agents in the system (effectively the contract is broadcast). The contract agents each submit a bid for the contract on offer. The bids are evaluated. The contracts are awarded to the contractors with the most appropriate bid. If the bid is accepted the contract agent cannot revoke. The CNP has been used in many applications, such as [5] and [55] for service level network control. Some of the CNP advantages include the ability to allocate tasks dynamically and to allow agents to be introduced and removed at will. Its limitations involve the fact that it does not detect or resolve conflicts and that the broadcasting of messages can sometimes incur high communication costs, e.g. cause or add to network congestion. Skarmeas and Clark [48] describe a system where communication and co-ordination occurs using a central message-board and the use of the CNP. Advertisements for services (e.g., routing information sets) are posted to the message-board, which then forwards (or broadcasts) them onto the relevant parties. Bidders respond directly to the sender (not through the message board), who then selects the best offer. Again, this is the coupling of a central co-ordination facility to a community of fully distributed control agents, where the robustness of the latter is dependent on the functioning of the former.

3.1 Software Agents for Adaptive Network Configuration

The broadband communications environment is probably one of the most compelling application areas for agent technology. This is due to the need for it to provide a fast switching ability and the potential for it to enable diverse services, both factors that are difficult to achieve when employing traditional network management paradigms. Furthermore, the provisioning of broadband facilities naturally leads to increases in network control and therefore induced management complexity.

In the broadband environment resource allocation is a difficult problem to address, primarily because network operational states are so dynamic, and this is particularly true in ATM. Solutions to these types of problems tend to be based on centrally controlled optimization algorithms and operational research formalisms. Many papers do not directly focus on the ability to adapt the network configuration to the evolution of operational network behavior. Whilst much research work has been conducted on, for example, optimal strategies for Connection Admission Control (CAC) [39], route topology, or optimal capacity configurations, there is very little literature on the problems associated with successive topology reconfiguration requiring co-ordination of different aspects of network functionality. Hardwicke and Davison [22] have conducted research into ATM topology performance management using a peer-to-peer agent oriented approach. In contrast, Frei and Faltings, [14] have chosen to rely on a hierarchical decomposition of the agent system framework and [23], [24] a layered approach. Gaïti and Boukhatem [16] have chosen a co-operative agent approach for dealing with congestion control in ATM networks. They use agents located at switch buffers to provide feedback information to the source node agent so that connections that are additionally added to the network do not add to the congestion being measured on that region of the network.

3.1.1 Agents for Service Provision in Virtual Private Networks

FIPA ([URL 3], and also Section 3.2.3), has defined an informative specification document that addresses the issue of agent technology in network management and network provisioning. They define a scenario where importance is placed on combining the service elements from different network providers in order to obtain a single end-to-end service for the user. FIPA states that “Intelligent Agent technology is promising in the sense that it will facilitate automatic negotiation of appropriate deals and configuration of service at different levels” (FIPA’97 Document 7). The scenario in the specification uses co-operating specialized negotiating agents to generate a dynamic Virtual Private Network (VPN). We have placed the FIPA VPN description in this section as it is mainly concerned with the provision of multi-media services and resources in a future broadband environment. As a first step, the roles of customer, service provider and network provider were mapped to different types of software agents. The following assumptions about the future telecom environment were made:

1. There will be typically multiple competing network providers (ISPs or telephone operators) each responsible for their own network domain, which allow subscribing to their connectivity services, possibly during a relatively short period of time.
2. Service providers will take care of negotiating with and choosing between the different network providers. Their role could be taken by new players, but also by ISPs or operators themselves.

There are three application specific agents, which together render this service. A Personal Communication Agent (PCA) will represent the interests of its end-user as well as possible, using information present in user’s profiles and preference files. The VPN will be used to run one or more end-user applications, each controlled by a different Application Agent. A VPN Service Provider Agent (VPN SPA) can be considered as some kind of VPN service broker between PCA and NPAs. It will search for the offers, most interesting to the PCA. A Network Provider Agent (NPA) represents a Network Provider responsible of managing its own network domain. Different NPAs can also co-operate.

There is a specific VPN Ontology specified that maps the agent ACL messages into the control parameters. See FIPA Doc. 7 for more information on the syntax and semantics [URL 3].

An agent based VPN service has been developed as part of the EU FACTS project (FIPA Advanced Communication and Transport Systems project [URL 15]). The service provision comprises three stages:

1. User negotiation stage:
2. VPN Service negotiation stage
3. VPN Service deployment stage

Each of these is now described in more detail (it is assumed that the standard interaction with the middle agents has already taken place, i.e., agents have already registered themselves with the FIPA Directory Facilitator (DF) agent, see Section 3.2.3):

During the User Negotiation Stage (Figure 1), the Personal Communication Agents (PCAs) act as personal organisers for the users, arranging a mutually convenient time for the multimedia application to be started. This task is in effect a 'meeting scheduling' task as described in the FIPA 97 Developer's Guide [URL 3]. An initiating PCA will start a negotiation process with other participating PCAs about parameters such as time, date, duration, cost, etc. In this stage there is also interaction between the PCA and one or more Application Agents, responsible for the management of a specific type of application or service accessible to the user. In FACTS we considered two application types: a Video Conferencing tool and a Calendar Management tool. Those applications will correspond to a Video-conference Application Agent (VCAA) and Calendar management Application Agent (CMAA). The Application Agents also interact with the Application Wrapper Agents (in our case VCWAs, CMWAs respectively) which provide the interface with the actual application (each application requires its own Application Wrapper Agent). This stage results in a scheduled video-conference meeting.

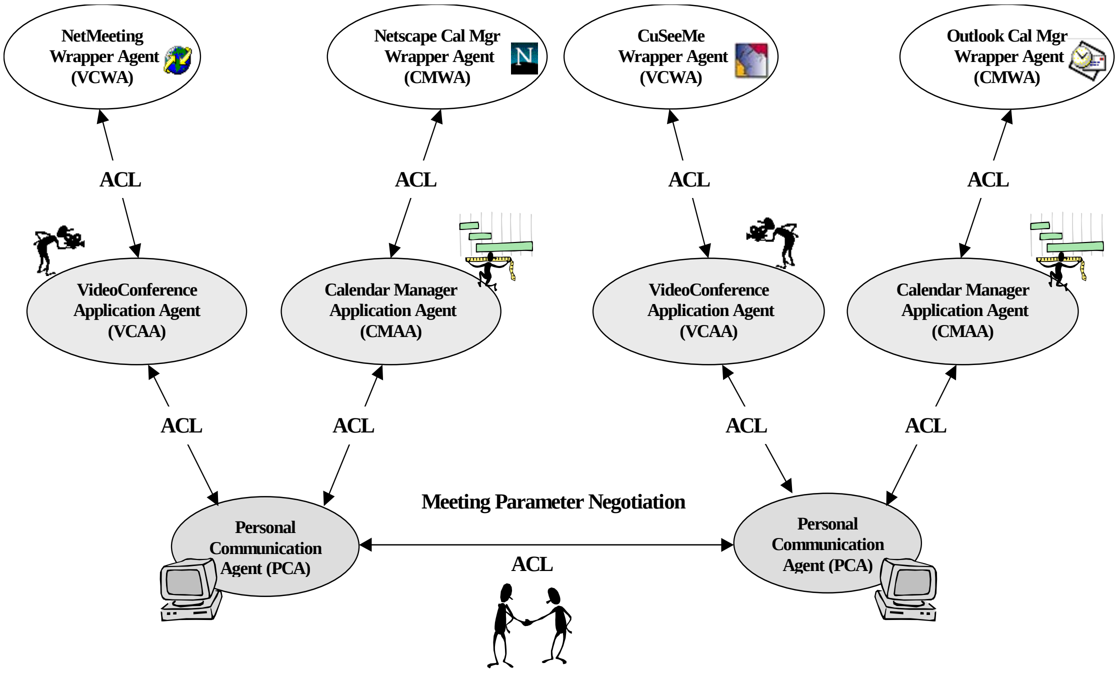

Figure 1: FACTS VPN agent application, stage 1, user negotiation

Note that all agents in Figure 1, interact via FIPA ACL messages (as discussed in the next section), combined with content following different vocabularies or ontologies. The use of wrapper agents will allow us to switch between different end-user applications. So even if one user has for example NetMeeting and the other one uses CuSeeMe, interoperability will be ensured by the definition of a common vocabulary for videoconferencing. The User Negotiation Stage is not dependent on calendar or video conference agent implementations as all communication in this stage is handled directly between the PCA (Personal Communication Agents). Further, the videoconference application must support an interoperability standard itself (e.g. H.323) for the data streaming. Similarly, different calendar management applications can be used via a calendar management vocabulary.

When the User Negotiation Stage is finished, the Service Negotiation Stage starts (Figure 2). One of the PCAs will communicate the details of the required service to a selection of SPAs. Each SPA must then negotiate with a set of NPAs to arrange the provision of the required VPN connection (Network Provisioning stage). This results in the selection of an NPA, which meets best the requirements of the SPA. Following this the SPAs which were contacted by the PCA will send their service offer to the PCA, which will select the SPA whose offer matches the service requirements best.

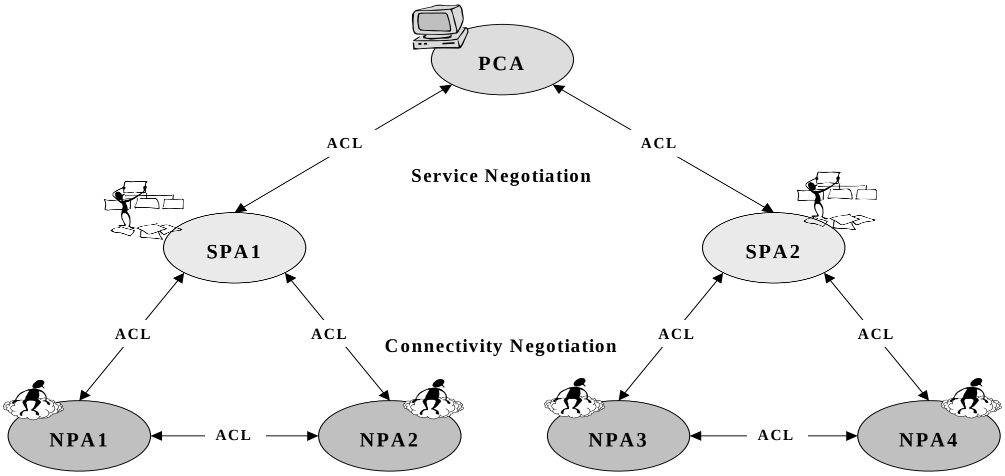

Figure 2: FACTS VPN agent application, stage 2, VPN service negotiation

The last stage is the actual VPN deployment (Figure 3). The service provider requests the network providers to set up the network connections. When these are in place the involved end-users are informed that the VPN is available for use. During the VPN life-time, management information released by any of the parties is passed on to relevant parties and appropriate actions can be taken. The service provider controls the decommissioning of the VPN (triggered by the initiating end-user or by a reservation system controlled by the provider). Participating users are then requested to terminate their VPN link and network providers are informed the VPN is terminated. Finally, network providers will charge the service provider for the use of their network. The service provider can add charges for additional services and will propagate them to the end-user

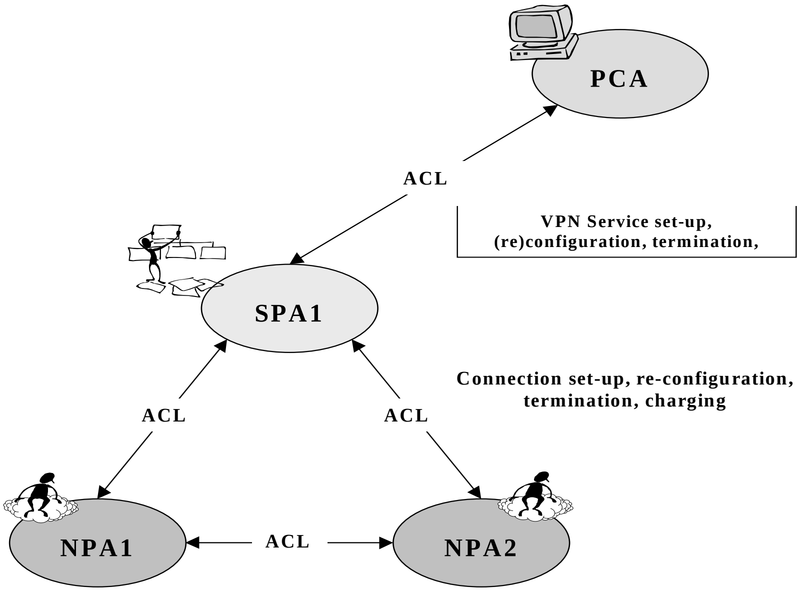

Figure 3: FACTS VPN agent application, stage 3, VPN service deployment

3.2 Agent Middleware and Network Integration

Traditionally, remote applications were and are still predominantly accessed over separate networks and service support infrastructures, dedicated to voice, data or video. The application, service [support] and the network functions were vertically integrated. Although, standards have been developed to enable a particular service from different vendors to interoperate with each other, there is a growing demand to use a single portal to access any service from any network. In the Telecom domain Messerschmitt [30] has identified a shift away from vertical integration, where a single provider offers a dedicated infrastructure embracing networks, services and applications, to horizontal integration. This horizontal integration is characterized by 3 layers: a set of integrated network and transport mechanisms, a set of generic support services (middleware) including operating system services, and a diverse set of applications and services (Figure 4). The separation of applications from transport and common functions encourages application diversity; any service provider can supply these applications.

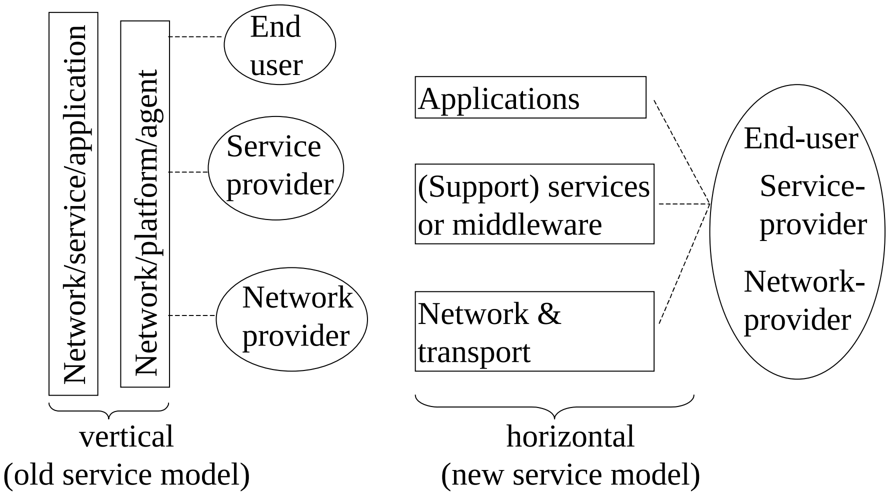

Figure 4. Network services are moving from vertically integrated to horizontally integrated service model.

Note that there is no exact correspondence between this model and the ISO Open Systems Interconnection Reference Model as Figure 4 is a logical service model not a network protocol model. Note also until the advent and use of agent standards (including de facto standards) agent services may also tend to follow a vertical service model.

In order to support transparent application access and interoperability, horizontal integration is required at both the network layer and the service layer. To access services from any network requires heterogeneous networks to be inter-linked. To combine heterogeneous services requires service interoperability. In addition, it may also be potentially advantageous, to introduce loose coupling, vertically, between levels. For example, software agents can enable lower level entities at the network level to make use of higher-level end-user and application profiles to allocate network bandwidth.

The roles between different users and providers of applications, services and networks will become more dynamic and inter-linked in future communication systems (see Figure 1). Traditionally, networks and services were usually managed by the network and service provider on behalf of the customer and offered as a single virtual network service but customers themselves are becoming more empowered to integrate and manage their own services. For example, a network customer in the guise of a service provider to its own customers, may wish to link inquiries to a sales database so that customers associated with the highest sales are always dealt with first.

3.2.1 Network Integration

A variety of forces are driving diverse sets of networks to become inter-linked. These forces include factors such as deregulation, increased competition in the marketplace and the penetration of Internet applications within the home and business. Technically, there are several different approaches which could lead to achieving the goal of interoperability within a heterogeneous network environment. Firstly, heterogeneous networks can be unified, for example, non-IP networks can be enhanced to carry IP packets perhaps by adding a ‘shim’ over another network layer protocol or by the use of tunneling [42]. Secondly, heterogeneous networks can be homogenized by simply replacing each network with one of the same type as the partner networks, i.e., it was envisaged initially that ATM would pervade the whole network from desktop to WAN (ATM everywhere). Thirdly, heterogeneous networks can be linked at discrete points in the network with devices such as bridges and gateways where different network protocols can be translated [42].

As mentioned previously, it is anticipated that the Internet, will dominate the network access infrastructure. It may also dominate the core or carrier network ("IP-everywhere") with the addition of support from new standards to achieve end-to-end QoS such as MPLS (Multi-Protocol Label Switching), RSVP (Resource Reservation Setup Protocol) and through initiatives to support both Integrated Services and Differentiated Services [URL 18].

3.2.2 Service Integration and Differentiation

To access any service, via any network, requires high-level service [infrastructure] abstractions which will enable users to browse the services available, to combine simple services to form more complex services, to set-up Service Level Agreements (SLAs) and to configure, initiate and control (possibly) multiple concurrent services.

This service integration may not only be user-driven but may also occur within the service user domain rather than the service provider domain. There are a variety of reasons for this. Service integration often requires access to information, which users may wish to keep private from any other organization. Users may spot niche opportunities and wish to synthesize value-added services quickly whilst competition is weak. If users own the domain expertise, they may not find it easy to impart it to others to act on their behalf.

Service integration and differentiation may occur in a phased evolution, passing from control by the provider to control by the user. In the first phase, service interfaces will be coalesced on the user or consumer-side and connected to a service infrastructure, such as middleware, inside the customer domain. In the second phase, the unified interface will be expanded with the customer being able to use the interface to secure online reconfiguration of existing services and provisioning of new services. Furthermore, these services will not be just those offered by a single provider: predefined service interfaces will enable third party service providers to define, configure and supply services. Access to these services will be mediated by an “intelligent” intermediary deployed by the operator which will also be able to ‘push’ new services pro-actively to customers. In a third phase consumer systems may adapt and evolve, perhaps automatically, independently of a network or service provider further up the chain.

A primary means of achieving service integration and differentiation is via the use of middleware. We define middleware to be any software entity that is interposed between a client (service user) and a server provider, a peer and other peers (one or more users or providers), or an application and a platform. This entity provides some kind of additional services that we would intuitively associate with a human middleman, broker, arbiter, facilitator, and so on, although obviously realized in computational terminology of types and services. Further these services are organized to be readily accessible by any user, yet to be independent of specific end users and services - this organization has arisen through a process of heterogeneous service generalization and rationalization.

There exists a variety of commercial-off-the-shelf object-oriented middleware including CORBA (OMG, 1992)], Java [URL 11], Common Object Model or COM [URL 12] and the Web [URL 13]. These support service integration at a low level of abstraction via a shared application programmer's interface using procedure calls or object invocation.
CORBA hides the transfer of data between service users and service providers. Optimally CORBA users must have detailed a priori knowledge, albeit in a language independent manner, of their service invocation interfaces before the clients are commissioned. There are several components of Java, which can be regarded as performing some sort of middleware role. These include: the Java Virtual Machine and a Remote Method Invocation interface to standardize service invocation, insulated from a particular hardware platform; Java Beans (and similarly COM) standardizes the packaging of services to support dynamic service selection and aggregation. The Web provides a generic invocation service using a standard address format, URL or Uniform Resource Locator but there is more important middleware functionality implicitly offered by the Web model of distributed computing. URLs are analogous to object references, the common currency of, for example, CORBA systems. But, to access the server, information, or program referenced by the URL, first the client has to know the URL. The way that a client discovers what URLs are available is via a search engine. A search engine therefore actually provides location services in the manner of a trader or yellow-page directory service.

Whilst such object-oriented middleware could provide many of the facilities to support the service integration, their use is not ideal when used within a federated, open, distributed environment. Object-oriented middleware can require too much detailed knowledge to be acquired before service invocation can occur. Some object-oriented middleware is difficult to extend at run-time, is not very re-configurable or adaptive and is unable to broker services intelligently within an open service environment. Although information is shared between service users and providers, the semantics of how to interpret and act on the information is not shared at this low level of abstraction.

In addition, the way object-oriented parts interact may naturally be very complex to reconfigure and maintain, depending on how the objects and object interaction is designed. The use of meta-objects, which contain information, to constrain the temporal object interaction sequence, to vet replacements and extensions of parts of the system, and to co-ordinate the actions between different autonomous object aggregates, becomes essential. Traditionally, these meta-objects exist in the form of an object-oriented application framework but these are not geared towards dynamic service re-configuration at run-time or towards intelligent re-configuration in response to unusual events.

Software agents offer many potential advantages over object-oriented middleware for the provision of enhanced service integration. Agents can support much more flexible means of interaction such as farming out parts of problems to others via collaborative problem solving; using inherently richer semantics for discovering the capabilities and requirements of service providers; use of intelligence to reason about alternatives, incomplete knowledge and past experience. Agents can share the semantics for interpreting the information. Collaborative agents could pervade the whole of the service architecture not just the application layer of the service model (Figure 1), for example, agents in the network layer could interoperate with application layer and user agents, enabling the delivery of particular applications to take into account service-independent personal preferences. Finally, (intelligent) agents can use knowledge representation and (logical) reasoning. This provides a more sophisticated mechanism for service allocation and provision that transcends the syntactic matching used in conventional object brokering [40].

3.2.3 Architectures and Models for Service Provision in Multi-agent Systems

The highly interactive nature of multi-agent systems points to the need for consensus on agent interfaces in order to support interoperability between different agent systems. One of the first and perhaps most well-known models for agent communication arose out of the DARPA funded Knowledge Sharing Effort (KSE) at the University of Maryland, circa 1990, [13]. This developed a three layer model to define the pragmatics, syntax and semantics for sharing knowledge and hence supporting agent communication. This model was implemented using three independent encodings: KQML, KIF and Ontolingua respectively. In practice, this model requires the addition of third-party agents called middle agents [11], which facilitate communication by providing traffic in meta knowledge about the capabilities of service providers and the preferences of service user agents. This meta knowledge is supplied to, or extracted from, middle agents using a sub-set of communicative primitives such as Broker-one, Recommend-one and Recruit-one. As KQML developed, diversification occurred and different dialects and enhancements arose, - there is no longer a single standard.

In 1996, a second initiative, a non-profit association, FIPA, the Foundation for Intelligent Physical Agents [36], arose to develop and ratify agent standards by common census (in contrast to the KSE program). The completion and adoption of such a standard is a prerequisite to the widespread commercialization and successful exploitation of intelligent agent technology. At this time, FIPA has around 50 member organizations (commercial and academic) committed to achieving the required consensus for interoperability.

In some respects, FIPA can be regarded as a second-generation architecture for agents. FIPA shares some similarities with the KSE model, it has standardized the pragmatic layer encoding (ACL) and the [content] syntactical and semantic layer encodings, see Labrou, Finin and Peng [27] for further details. However, FIPA differs from KSE in terms of the functionality, which is standardized, and how this functionality is modeled. Firstly, FIPA has standardized several types of middle-agent (Figure 5) such as a yellow page agent (directory facilitator) and a message-forwarding agent (Agent Communication Channel). Secondly, the first set of FIPA specifications, FIPA 1997, (FIPA delivers specifications annually) has standardized some aspects of agent management by providing another middle agent called the Agent Management System. Note that the facilitation and management communicative primitives are not part of the agent pragmatic layer. Note also that an existing transport protocol IIOP has been mandated as a transport protocol to solve the bootstrap problem to allow messages from agents outside the platform to be delivered to middle agents, and hence service and user agents, within the platform. Thirdly, FIPA has specified an abstract model to allow agents to access non-agent services via ‘wrappers’. These three specifications form the normative and core set of specifications. In addition to the normative specifications, there is a set of domain, or application dependent, agent abstractions. This additional, supporting, material is included in informative sections, which are configured to help developers implement application agents on the FIPA platform. The agent abstractions address specific task oriented service domains including travel, audio-visual entertainment and VPN services. N.B. FIPA 97 specifications do not prescribe particular implementations of application agent, or middle agent internals, nor do they explicitly dictate how agent communication is to be implemented. The specifications, merely prescribe the minimum interoperability standards, upon which developers must implement their own solutions

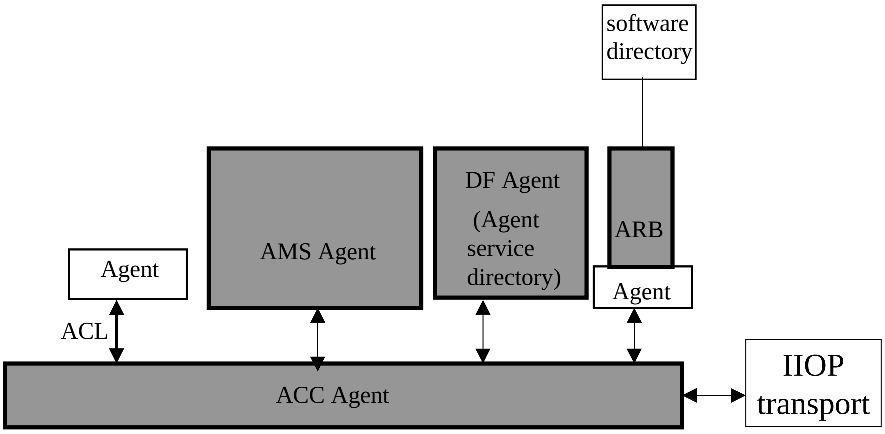

Figure 5: The FIPA Agent Platform

The FIPA middle agents which comprise its ‘Agent Platform’ (AP) consist of: Agent Management System (AMS)

Agent; Directory Facilitator (DF) Agent; Agent Communication Channel (ACC) and an agent playing a role as an Agent Resource Broker (ARB) to interface to non-agent software. Agents outside the platform must be able to communicate using IIOP protocol. The shaded areas indicate the boundary of the AP.

The VPN application described earlier in Section 3.1.1 is an example of how (FIPA) agents can integrate services. Here agents are used to integrate a meeting scheduler service, a video conference service and a VPN (Virtual Private Network) service application.

3.3 Mobile Agents

Mobile agents were originally concerned with the ability to move executable code from one computer to another. The main benefit obtained from adopting this approach was that available computational resources on another computer could be utilized when resources on the original computer became scarce. To date, researchers have become more interested in dealing with the idea of moving so called ‘intelligence’ from one place of execution to the next.

There has been a lot of development and general excitement in the area of mobile agent technology, much of which has evolved from the platform independence of the Java language. A Mobile agent overview paper by Morreale [33] provides some interesting applications for mobile agent technology, similarly [20] contains a section on mobile agent technology that describes some of the mobile agent platforms that are currently available. For a mixed intelligent and mobile agent perspective on agents in communications systems we refer the reader to [25].

The concept of mobile agents further blurs the distinction between client and server type models, however it still remains the case that the passing of messages is often more efficient than moving an agent from node to node and therefore there has to be careful analysis of the benefits that each approach can provide in the particular application domain. However, since mobile agents provide a suitable mechanism for dynamically placing new control software within a telecommunications environment, its adoption will have an impact on the actual network architecture and the related signaling protocols of the telecommunications system [29], [41]. For example, the metaphor itself - mobile, encapsulated programs provides a new framework for thinking about computer networks and the modularity of mobile agent approach encourages the design of more flexible and adaptive network architectures [41].

Appleby and Steward [2] describe the application of mobile agents to congestion control in a circuit switched environment [2]. Their work demonstrated the feasibility of using agents to autonomously manage a network.. Control is provided by using an “ensemble of mobile agents”. The “ensemble” came from the authors’ requirements that for robust control “agents should be present in reasonably large numbers”[2]. In fact the number of agents used to control the network dynamically changed in population number so as to adjust to the dynamics of the network state. The two different kinds of agents that they described were *load agents,* to provide the lowest level of control in the system and *parent* agents. The *load agents* were responsible for distributing the communications traffic evenly through the available circuits. The parent agent’s tasks were to manage the population dynamics of the *load agents*. The *parent agent* possessed the ability to “launch” new mobile *load agents* when it detected nodes that were in high states of utilization. The experiments demonstrated that the mobile load agents could be used to measure the nodal utilization (the amount of traffic throughput in a single network node) and balance some of the traffic when demand became high. This behavior was achieved via the use of a distributed implementation of Dijkstra’s shortest path algorithm [12]. Some of the disadvantages involved in this method of node searching is that sometimes cyclic routes were generated (see [46]).

3.3.1 Mobile Agent based Routing

Schoonderwoerd et al. [46] focussed on using the ant analogy for the control of route path finding in networks via stigmergy. The basic idea of stigmergy is to utilize the agent's "world" as a memory store. The type of routing considered in this work is known as hop-by-hop routing. Hop-by-hop routing occurs when each node has a routing table, which gives information about the next nearest neighbor nodes. The routing table, in its most basic form, consists of a table of node identifiers each with an associated priority. To route a call, information is given to the source node, which then checks for a node that is consistent with getting to the designated destination node. Once a node is chosen, the destination information is passed onto the chosen node and the process continues until the destination node is reached. Schoonderwoerd's research considered simulating ants that randomly roamed the network laying a trail of “pheromones” at each node. The amount of pheromone deposited depended on the amount of time each of the ants were delayed in the nodes that they traversed and was directly dependent on the utilization of the node (amount of total capacity occupied by existing calls). As the ants aged, the amount of pheromone that they deposited decreased. The amount of pheromone became in effect the priority of that route. Further ants followed the routes where higher levels of pheromones had been deposited, therefore re-enforcing that path. Paths with high levels of pheromone indicated a less congested route, because congested nodes would delay the ants and therefore prevent the ants from depositing a greater amount of pheromone. Schoonderwoerd et al. [46] extended Appleby and Steward's work by paying attention to the elimination of unnecessarily long routes and also the prevention of cyclic routes. The first problem was overcome by summing the squared utilization of the nodes from the actual node back to the source node. The later problem was overcome by making sure that the ants only updated the routes in the direction of their source node and limiting the number of ants that could update the node routing table, therefore eliminating inconsistencies.

Minar, Kramer and Maes [32] use mobile agents to discover information about network connectivity. The agents build a map of the links and node configuration so that dynamic routing decisions can be made. Within their scenario the nodes are radio-frequency transceivers distributed through two dimensional space. Each Agent has a three step control cycle, which consists of learning about the edges (links) at the executing node; learning about node information from any other agents operating at that node; and learning about the choice of the next node to re-locate to. The main objective of the research is to test different "movement" strategies. The first strategy is a control mechanism where the agents choose an adjacent node at each time event. The strategies increase in sophistication. The second strategy (denoted as "Conscientious") chooses adjacent nodes that have not been visited before or have been visited least recently. The agents using the Conscientious mechanism do not base their choice on any information learned from the other agents. The final strategy ("Super Conscientious"), bases the choice of node visits on information learned from other peer agents in the system. Results obtained from the experiments showed that agents which co-operate by passing network topology information could map out the network configuration in less time than that taken where there was limited or no co-operation [URL 6].

MAGENTA is an environment for the execution of mobile agents [URL 9]. It consists of "lieus" which are static programs that are capable of creating, sending, receiving and saving agents (cf. with "Places" in IBM Aglets) [URL 7]. A lieu has an unique identifier. The agents can hop from lieu to lieu and can interact with the system and other agents when executing at a lieu. Security for the executing platform is supported, as each agent has a global identifier that acts as a permit. The agent is only allowed to carry out system resource changes once the permit is checked and verified.
In MAGENTA backup copies of agents and lieus are maintained so that agents can be re-created when they disappear due to site failure. This is achieved via a directory mechanism [43].

MAGNA stands for Mobile AGeNt Architecture. The project is developing a mobile agent approach for enhancing and extending the developments made in the TINA-C environment. A long term objective of the project is to develop a generic architecture that will aid the needs of future communications systems. MAGNA has recognized the possible benefits that mobile agent technology may inject into the telecommunications industry, particularly with respect to the service scalability problem inherent in Intelligent Networks. They have, however, also realized that the value of intelligent mobile agents for the provision of telecommunication services may be questionable for some application domains in which traditional RPC mechanisms might be more adequate. Therefore, they are currently developing a mobile agent architecture (MAGNA), which considers Remote Programming and Remote Procedure Call as potentially complementary and not mutually exclusive technologies. It is their belief that this technology will enable telecommunication services to be provided instantly and customized directly at the locations where the intelligence is needed, namely it will enable "Intelligence on Demand" [29].

3.4 Toolkit based Construction of Agent Systems

In the last year or so there has been particular proliferation in the market for the production of agent-based frameworks or software toolkits. Building an agent system is time consuming and in some cases can be expedited by using an infrastructure provided by an agent toolkit. Many of the agent toolkits provide support to handle the multi-threading, communications and basic interface functionality such that the agent system designer can concentrate more on the modeling of the particular application domain. Therefore, these toolkits enable software developers to create agent based systems whilst removing some of the tedious and often complex tasks. Of particular interest is the number of commercial templates and toolkits that are available for download on the WWW, including AgentBuilder [URL 8], IBM Aglets [URL 5], Grasshopper [URL 2]. Of course, there is also a fair share of agent construction toolkits that are for in-house development, e.g., Zeus toolkit [35].

JAFMAS [URL 16] is a Java based agent building framework that is made up from sixteen Java classes, although only four abstract classes need to be extended. JAFMAS allows agent system designers to develop speech-act based MAS, supporting both directed and broadcast communication between agents. Co-ordination support is provided through automata modeling, where agent plans can be described as rule based conversations. To achieve overall co-ordination of the agency, a Petri Net modeling tool is used. This enables the automata models to be checked against possible conflicts and deadlock situations. More information on JAFMAS can be found at [URL 16]. Similarly JATLite [URL 17] is also a Java based framework for the development of speech-act based agents.

Aglet technology (IBM Japan) [URL 5], has re-opened the interest in mobile agent technologies and provides a Java implemented autonomous mobile software agent toolkit. Aglets extend the model of network-mobile code by allowing the class files to maintain state when migrating from node to node. A major concern with MA technology is the lack of security on the agent side of the relationship; however, Aglets still rely on the presence of a Java Virtual Machine to execute their state and actions, so the potential for security breeches is limited [53].It is often the case that the agent requires a specific piece of software in order to gain access to the resources on a host computer. In the IBM Aglet framework this software is called “places” and provides some security mechanisms to prevent the server from being violated by the executing agent. However, there is considerable concern about users delegating confidential information (such as credit card details) to their MA and releasing them into an open network system where, without proper security considerations, server computers could gain access to confidential information [52].

4 SUMMARY

This paper has provided an introduction to the research literature on software agents for telecommunications systems. We have briefly described the differing classifications of software agents, including mobile agents and so called intelligent agents. We provided a description of our perception of agent technology and the benefits of agent-based approaches have over existing technologies, including how the agent approach provides a system developer with a new metaphor for describing the interactions that occur between interacting software components.

Latter sections considered some of the problems that are found in the communications industry with some details of how the agent approach has been applied. For example, we have described work performed as part of the FACTS project, which has applied agents to manage service provisioning within virtual private networks. We have also described the change from the more traditional vertical service model to a horizontal service integration model in the telecommunications market. The horizontal model allows for easier integration of various applications and services in a multiple service provider environment.

We conclude that with the rapid changes that are occurring in the telecommunications industry, including the proliferation in the number of service providers, their need to deploy ever more value added services across multiple platforms and increased competition, the multi-agent systems approach appears to be one such suitable management solution.

ACKNOWLEDGEMENTS

The authors at Queen Mary and Westfield College thank EPSRC and BT Labs for partly funding this work. The work of the authors at Imperial College is part the CASBAh project jointly funded by EPSRC, under grant GR/L34440 and Nortel Networks and their support is gratefully appreciated. The VPN example is taken from work done as part of the EU FACTS project. The authors would like to thank the people that provided comments on early versions of this paper, especially Michael Wooldridge and Steve Robertshaw.

REFERENCES

[1] J. L. Adams, Asynchronous transfer mode - an overview. BT Technology Journal Vol. 13 No. 3. pp 9-14. 1995

[2] S. Appleby, S. Steward. Mobile software agents for control in telecommunications networks. BT Technology Journal, Vol.12 No. 2 pp.104-113. 1994.

[3] J. Bigham, A. Hayzelden, Subsumption and Survivability in Bidding for Bandwidth. In Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS-98), Paris, France, July 3-7, 1998. Published by IEEE. ISBN 0-8186-8500-X. Available at http://www.agentcom.org/agentcom/

[4] U. Black, Emerging Communications Technologies. Prentice Hall Series. ISBN 0 13 742834 0. 1997.

[5] M. Busuioc, Distributed Intelligent Agents - A solution for the Management of Complex Services. IATA’96 Workshop, August 1996. Held at ECAI 96, Budapest, Hungary. 1996.

[6] M. C. Chan, G. Pacifici, R. Stadler, Managing Multimedia Network Services. Journal of Network and Systems Management, Vol. 5 No. 3. 1997.

[7] S. H. Clearwater, Market-Based Control: A Paradigm for Distributed Resource Allocation. World Scientific, 1996.

[8] R. Davison, M. Azmoodeh, Performance Management of Public ATM networks - A Scaleable and Flexible Approach. Proceedings of the 5th IFIP/IEEE International Symposium on Integrated Network Management. 1997.

[9] R. Davison, J. Hardwicke, M. Cox, Applying the Agent Paradigm to Network Management. BT Technology Journal, Vol. 15. No. 3. 1998.

[10] M. Decina, V. Trecordi, Convergence of Telecommunications and Computing to Networking Models for Integrated Services and Applications. Proceedings of the IEEE. Vol. 85, No. 12. December 1997.

[11] K. Decker, K. Sycara, M. Williamson, Middle-agents for the Internet. In Proceedings of the 15th International Joint Conference on Artificial Intelligence, Nagoya, Japan, August 1997, pp578-583.

[12] E. W. Dijkstra, A Note on Two Problems in Connection with Graphs. Numerical Mathematics 1. pp. 269--271. 1959.

[13] T. Finin, Y. Labrou, J. Mayfield, KQML as an agent communication language. In Software agents, Bradshaw JM (ed.), MIT Press, 1997. pp.291-316.

[14] C. Frei, B. Faltings, Intelligent Agents for Network Management. AI for Network Management Systems 1997, IEE Digest No. 97/094.

[15] A. Fuggetta, G. P. Picco, G. Vigna, Understanding Code Mobility. IEEE transactions on Software Engineering. Vol. 24, No. 5. May 1998. Pp.342-362. 1998.

[16] D. Gaïti, N. Boukhatem, Cooperative Congestion Control Schemes in ATM Networks. IEEE Communications Magazine, Vol. 34. No. 11, pp.102-110, 1996.

[17] D. Gaïti, A proposal for integrating intelligent management in the intelligent network conceptual model. Computer Networks and ISDN Systems, Vol.28. No.4. 1996.

[18] G. Goldszmidt, Y. Yemini, Distributed Management by Delegation. In 15th International Conference on Distributed Computing Systems. IEEE Computer Society, June. 1995.

[19] G. Goldszmidt, Y. Yemini, Delegated Agents for Network Management. IEEE Communications Magazine, Vol. 36, No. 3. March, 1998.

[20] S. Green, L. Hurst, B. Nangle, P. Cunningham, F. Somers, R. Evans. Software Agents: A review. Technical Report of Trinity College, University of Dublin. TCD-CS-1997-06. Available at http://www.cs.tcd.ie/research_groups/aig/iag/. 1997.

[21] F. Halsall, Data Communications, Computer Networks and Open Systems. Addison-Wesley. ISBN 0 201 42293 X. 1996.

[22] J. Hardwicke, R. Davison, Software agents for ATM performance management., NOMS'98, 1998.

[23] A. Hayzelden, J. Bigham, Management of ATM Resource Configuration, Automated by Distributed Agent Control. In Proc. of the IEE Fourth Network Communications Symposium, Manchester, 7-8 July 1997. Available at http://www.agentcom.org/agentcom/. 1997.

[24] A. Hayzelden, J. Bigham, Heterogeneous Multi-Agent Architecture for ATM Virtual Path Network Resource Configuration. In Intelligent Agents for Telecommunications Applications (IATA’98). LNAI 1437. LNAI 1437. Edited by Albayrak, S., and Garijo, F. J. ISBN 3-540-64720-1. Published by Springer Verlag. pp. 45-59. http://www.agentcom.org/agentcom/. 1998.

[25] Hayzelden, A. & Bigham, J. (Eds.) (1999) ‘Software Agents for Future Communications Systems’. Published by Springer-Verlag. ISBN - 3-540-65578-6. 1999.

[26] IATA, Intelligent Agent for Telecommunications Applications. Workshop Proceedings at ECAI’96, Budapest., Hungary.

[27] Y. Labrou, T. Finin, Peng, Yun, (1999) ‘The current landscape of Agent Communication Languages’. IEEE Intelligent Systems. Vol. 14, no. 2., 1999.

[28] E. C. Lupu, M. Solman,. Towards a Role-Based Framework for Distributed Systems Management. Journal of Network and Systems Management, Vol. 5. No. 1, 1997. pp 5-30. 1997.

[29] T. Magedanz, K. Rothernel, S. Krause, Intelligent Agents: An Emerging Technology for Next Generation Telecommunications? INFOCOM '96, March 24-28, 1996, San Francisco, CA, USA.

[30] D. Messerschmitt, The Future of Computer-Telecommunications Integration. IEEE Communications Magazine Vol. 34 No. 4. 1996. pp 66-69. 1996.

[31] K. Meyer, M. Erlinger, J. Betser, C. Sunshine, G. Goldszmidt, Y. Yemini. Decentralising Control and Intelligence in Network Management. Integrated Network Management IV, Chapman & Hall, 1995. Ed. Sethi et al. pp. 4-15. ISBN 0412715708.

[32] N. Minar, K. H. Kramer, P. Maes, Cooperating Mobile Agents for Mapping Networks. Proc. First Hungarian National Conference on Agent Based Computing, May, 1998. http://www.media.mit.edu/~nelson/research/routes-coopagents/. 1998.

[33] P. Morreale, Agents on the Move. IEEE Spectrum. Vol. 35, No. 4. pp34-4. 1998.

[34] H. S. Nwana, D. T. Ndumu, Agents of Change in Future Communication Systems. In A. Hayzelden and J. Bigham (eds), Software Agents for Future Communications Systems. Chapter 2. ISBN 3-540-65578-6. 1999.

[35] H. S. Nwana, D. T. Ndumu, L. Lee, ZEUS: An Advanced Tool-Kit for Engineering Distributed Multi-Agent Systems. Proceedings of PAAM'98, Westminster Central Hall, London, UK. 23rd-25th April 1998. ISBN 0 9525554 8 4. 1998.

[36] P. D. O'Brien, R. C. Nicol, FIPA towards a standard for software agent's. BT Tech. J 1998. Vol. 16, No. 3. pp.60-68. 1998.

[37] J. Pavón, J. Tomás, Y. Bardout, L. H. Hauw. CORBA for Network and Service Management in the TINA Framework. IEEE Communications Magazine, Vol. 36, No. 3, March, 1998. Pp.72-79.

[38] J. V. Pitt, E. H. Mamdani, R. G. Hadinham, A. J. Tunnicliffe, Agent-Oriented Middleware for Telecommunications Network & Service Management. AI for Network Management Systems 1997, IEE Digest No. 97/094. 1997.

[39] J. M. Pitts, J. A. Schormans, Introduction to ATM Design and Performance. John Wiley & Sons Ltd, 1996. ISBN 0-471-96340-2. 1997.

[40] A. Puder, A Declarative Extensions of {IDL}-based Type Definitions with Open Distributed Environments. Int. Conf. Object-Oriented Information Systems. Springer-Verlag, 1994.

[41] B. Quendt, An agent-based Resource-Control in a Signaling System for the Open Telecommunication Market. Proceedings of PAAM '97, Westminster Central Hall, London, UK. 21st-23rd April 1997.

[42] M. T. Rose, The Open book: a practical perspective on OSI. Prentice Hall. 1990. ISBN 0136430163.

[43] A. Sahai, C. Morin, Mobile Agents for Enabling Mobile User Aware Applications. Proceedings of the Second International Conference on Autonomous Agents (Autonomous Agents '98), Minneapolis/St. Paul, MN USA, May 9-13, 1998.

[44] T. Sandholm,. V. Lesser. On Automated Contracting in Multi-enterprise Manufacturing'. Proceedings of the Improving Manufacturing Performance in a Distributed Enterprise: Advanced Systems and Tools Conference, Edinburgh, Scotland, pp. 33-42, July 13-14. 1995.

[45] R. M. Schmid, R. Beller, S. Giordano, H. Flinck, IP and ATM - a position paper. EXPERT deliverable, AC094, WP 2.1. 1997.

[46] R. Schoonderwoerd, O. Holland, J. Bruten, Ant-like agents for load balancing in telecommunications networks. Proceedings of First International Conference on Autonomous Agents. http://www-uk.hpl.hp.cpm/people/ruud/. 1997.

[47] D. Sincoskie, Broadband ISDN is happening - except it's spelled IP. IEEE Spectrum January 1997. Pp32-33. 1997.

[48] N. Skarmeas, K. L. Clark, Intelligent Routing based on Active Patterns as the basis for the integration of distributed information systems. http://www-lp.doc.ic.ac.uk/~klc/. 1997.

[49] R. G. Smith, R. Davis, Frameworks for Cooperation in Distributed Problem Solving. IEEE Transactions on System Man and Cybernetics, 1981, Vol. 11 No.1. pp. 61-70. Also in 'Readings in Distributed Artificial Intelligence'. Morgan Kaufman Publishers, 1988. Bond, A. and Gasser, L. (Editors)

[50] D. Stevenson, Network Management: What it is and what it isn't. White Paper (1995) Available at: http://netman.cit.buffalo.edu/Doc/Dstevenson

[51] D. Tennenhouse, J. Smith, W. D. Sincoskie, D. Wetherall, A Survey of Active Network Research. IEEE Communications Magazine, January 1997.

[52] B. Venners, Under the Hood: The architecture of aglets. April 1997. http://www.javaworld.com/javaworld/jw-04-1997/jw-04-hood.html

[53] B. Venners, Solve real problems with aglets, a type of mobile agent, May 1997. http://www.javaworld.com/javaworld/jw-05-1997/jw-05-hood.html

[54] Z. Wang, J. Crowcroft, Quality-of-Service Routing for Supporting Multimedia Applications. IEEE Journal on Selected Areas in Communications, Vol.14, No. 7, September 1996.

[55] R. Weihmayer, R. Brandau, Cooperative distributed problem solving for communication network management. Computer Communications, Vol.13, No. 9. November 1990. pp.547-556.

[56] M. Wooldridge, N. R. Jennings, Intelligent Agents: Theory and Practice. The Knowledge Engineering Review. Vol. 10. No. 2. 1995. pp.115-152. 1995.

Universal Resource Locator (URL) addresses

[URL 1] Perkins, C. E. ‘Mobile Networking Through Mobile IP’. http://www.computer.org/internet/v2n1/perkins.htm
[URL 2] Grasshopper platform: http://www.ikv.de
[URL 3] FIPA Specifications: http://www.fipa.com
[URL 4] TINA-C homepage: http://www.tinac.com
[URL 5] IBM Aglet Framework: http://www.trl.ibm.co.jp/aglets
[URL 6] Nelson Minar ‘s Web page: http://www.media.mit.edu/~nelson/research/routes-coopagents/
[URL 7] Programming Mobile Agents in Java - With the Java Aglet API. Danny B. Lange and Mitsuru Oshima, IBM Research. http://www.trl.ibm.co.jp/aglets/aglet-book/index.html
[URL 8] AgentBuilder Framework: http://www.agentbuilder.com
[URL 9] Akhil Sahai homepage: http://www.irisa.fr/prive/sahai/
URL 10] IETF http://www.ietf.org/html.charters/
[URL 11] Java home-page, http://www.javasoft.com
[URL 12] Microsoft COM Home page, http://www.microsoft.com/com/default.asp
[URL 13] World Wide Web Home page, http://www.w3.org
[URL 14] Federation of Intelligent Physical Agents home page, http://www.fipa.org.
[URL 15] FIPA Agent Communication Technologies and Services (FACTS) ACTS project home page, http://www.labs.bt.com/profsoc/facts.
[URL 16] JAFMAS Toolkit: http://www.ececs.uc.edu/~abaker/JAFMAS
[URL 17] JATLite Toolkit: http://java.stanford.edu/java_agent/html
[URL 18] IETF Differentiated Services, http://www.ietf.org/html.charters/ diffserv-charter.html
[URL 19] ACTS IMPACT Project http://www.acts-impact.org/

Biographies

Alex Hayzelden is a lecturer in the department of Electronic Engineering, Queen Mary and Westfield College. His interests include the application of agent technology techniques to the management of communications systems. He is a co-editor of a book entitled "Software Agents for Future Communications Systems", Springer-Verlag, 1999.

John Bigham researches into intelligent agents applied to telecommunications and the applications of uncertainty management techniques to engineering problems. He currently participates in the ACTS Intelligent Agents project IMPACT.

Stefan Poslad is a research fellow in the Intelligent and Interactive Systems Group, Department of Electrical & Electronic Engineering, at Imperial College, London. He currently participates in the UK EPSRC / Nortel Network CASBAh project. His research interests include distributed systems, communication systems, self-configurable systems and software agents.

Philip Buckle is currently the Agent Technology Prime in the Advanced IP Services and Management Group at Nortel networks, Harlow, England. He currently participates in the EU FACTS project. His current research interests include negotiation protocols, biological metaphors in computing and their application to enhancing agent technology.

Abe Mamdani holds the Nortel Network/RAE Chair of Telecommunications Strategy and Services in the Department of Electrical & Electronic Engineering, Imperial College. He is world-renowned for his research into fuzzy logic. His recent research has focussed on uncertainty in Artificial Intelligence and intelligent agents applied to telecommunications related problems.