Agent Technology Roadmap *Draft*

A Roadmap for Agent Based Computing

agent based computing

For comment

Draft

Compiled, written and edited by

Michael Luck

Peter McBurney

Onn Shehory

Steve Willmott

and the AgentLink Community

Status of this document

This roadmap has been prepared as part of the activities of AgentLink III, the European Coordination Action for Agent-Based Computing (IST-FP6-002006CA).

The corresponding editor of this document, Michael Luck, is based in the School of Electronics and Computer Science, University of Southampton, Southampton, SO17 1BJ, and may be contacted via email at mml@ecs.soton.ac.uk.

Acknowledgements

This report is a collaborative effort, involving numerous different contributors. We have tried to list most in the appendix at the end of the report. We are grateful to all who contributed, including those not named, and welcome further inputs.

Special thanks go to Catherine Atherton, Roxana Belecheanu, Becky Earl, Adele Maggs, Steve Munroe and Serena Raffin, who all contributed in essential ways to the production of this document.

Editors

Michael Luck
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
United Kingdom
mml@ecs.soton.ac.uk

Peter McBurney
Department of Computer Science
University of Liverpool
Liverpool L69 3BX
United Kingdom
p.j.mcburney@csc.liv.ac.uk

Onn Shehory
IBM - Haifa Research Labs
Haifa University
Mount Carmel, Haifa
31905 Israel
onn@il.ibm.com

Steven Willmott
Dept Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Jordi-Girona 1-3E-08034
Barcelona, Spain
steve@lsi.upc.edu

AgentLink III

AgentLink III is an Information Society Technologies (IST) Coordination Action for agent-based computing, funded under the European Commission's Sixth Framework Programme (FP6), running through 2004 and 2005. Agent-based systems are one of the most vibrant and important areas of research and development to have emerged in information technology in recent years, underpinning many aspects of broader information society technologies.

The long-term goal of AgentLink is to put Europe at the leading edge of international competitiveness in this increasingly important area. AgentLink is working towards this by seeking to achieve the following objectives.

- To gain competitive advantage for European industry by promoting and raising awareness of agent systems technology.
- To support standardisation of agent technologies and promote interoperability.
- To facilitate improvement in the quality, profile, and industrial relevance of European research in the area of agent-based computer systems, and draw in relevant prior work from related areas and disciplines.
- To support student integration into the agent community and to promote excellence in teaching in the area of agent-based systems.
- To provide a widely known, high-quality European forum in which current issues, problems, and solutions in the research, development and deployment of agent-based computer systems may be debated, discussed, and resolved.
- To identify areas of critical importance in agent technology for the broader IST community, and to focus work in agent systems and deployment in these areas.

Further information about AgentLink III is available from the AgentLink website at www.agentlink.org

Purpose of the Roadmap

In trying to raise awareness and to promote take-up of agent technology, there is a need to inform the various audiences of the current state-of-the-art and to postulate the likely future directions the technology and the field will take. This is needed if commercial organisations are to best target their investments in the technology and its deployment, and also for policy makers to identify and support areas of particular importance. More broadly, presenting a coherent vision of the development of the field, its application areas and likely barriers to adoption of the technology is important for all stakeholders. AgentLink is undertaking this technology roadmapping study in order to develop just such a strategy for agent research and development.

The roadmap is a living document, and the roadmapping process is ongoing. This draft report reflects the overall efforts of the roadmapping team in developing a framework and outlining key directions and messages that have been elicited to date. It is now being published as a public draft for comment from the community in advance of formal publication. Feedback and contributions on all aspects of this document are welcome, but must be received by 15 August 2005 if they are to be incorporated into the final published version in the autumn of 2005; send email to the roadmap coordinator, Michael Luck, at mml@ecs.soton.ac.uk. **In particular, we are keen to receive comments and contributions on Sections 7 and 8 (Technology Roadmap and Challenges) for inclusion in the final roadmap.**

Contents

1 What is agent technology?

1.1 Agents as design metaphor

Agents provide designers and developers with a way of structuring an application around autonomous, communicative components, and lead to the construction of software tools and infrastructure to support the design metaphor. In this sense, they offer a new and often more appropriate route to the development of complex systems, especially in open and dynamic environments. In order to support this view of systems development, particular tools and techniques need to be introduced. For example, methodologies to guide analysis and design are required, agent architectures are needed for the design of individual components, tools and abstractions are required to enable developers to deal with the complexity of implemented systems, and supporting infrastructure (including more general, current technologies, such as web services) must be integrated.

1.2 Agents as a source of technologies

Agent technologies span a range of specific techniques and algorithms for dealing with interactions in dynamic, open environments. These address issues such as balancing reaction and deliberation in individual agent architectures, learning from and about other agents in the environment, eliciting and acting upon user preferences, finding ways to negotiate

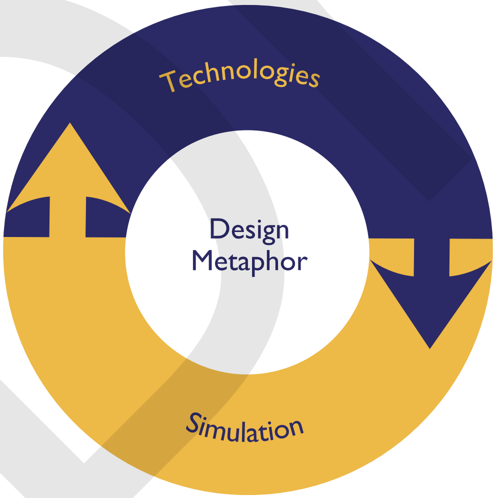


Figure 1.1: Agent-based computing spans technologies, design and simulation

and cooperate with other agents, and developing appropriate means of forming and managing coalitions. Moreover, the adoption of agent-based approaches is increasingly influential in other domains. For example, multi-agent systems have already provided faster and more effective methods of resource allocation in complex environments, such as the management of utility networks, than previous approaches.

1.3 Agents as simulation

Multi-agent systems offer strong models for representing real-world environments with an appropriate degree of complexity and dynamism. For example, simulation of economies, societies and biological environments are typical application areas.

The use of agent systems to simulate real-world domains may provide answers to complex physical or social problems that would be otherwise unobtainable, as in the modelling of the impact of climate change on biological populations, or modelling the impact of public policy options on social or economic behaviour. Agent-based simulation spans: social structures and institutions to develop plausible explanations of observed phenomena, to help in the design of organisational structures, and to inform policy or managerial decisions; physical systems, including intelligent buildings, traffic systems and biological populations; and software systems of all types, currently including eCommerce and information agency.

In addition, multi-agent models can be used to simulate the behaviour of complex computer systems, including multi-agent computer systems. Such simulation models can assist designers and developers of complex computational systems and provide guidance to software engineers responsible for the operational control of these systems. Multi-agent simulation models thus effectively provide a new set of tools for the management of complex adaptive systems, such as large-scale online resource allocation environments.

Agent systems are not simply panaceas for these large problems; they have been demonstrated to provide concrete competitive advantages such as:

- improving operational robustness with intelligent failure recovery;
- reducing sourcing costs by computing the most beneficial acquistion policies in online markets; and
- improving efficiency of manufacuring processes in dynamic environments.

2 Technological Context

Until recently, one of the major hurdles for the widespread adoption of agent technologies within the wider IT context has been the lack of infrastructure able to support the creation of dynamic and heterogeneous networks of devices and services that is central to the support of significant agent-based systems.

However, recent years have seen the development of a vast array of middleware technologies to support emerging enterprise level (and quality of) systems. Such technological infrastructure ranges from low-level wireless communication protocols like Bluetooth to higher-level web services technologies. In addition, they also span the range of devices supported from limited-capability devices such as mobile phones and PDAs to workstations and high-performance computing. Technologies such as Jini, UPnP and Salutation, for example, define discovery and registration protocols that allow for dynamic discovery of components in networks; peer-to-peer frameworks such as JXTA provide simple to deploy messaging and service advertisement for non-client server style applications. For distributed applications, the Object Management Group's CORBA specification defines a broad set of features for the development of distributed object based applications, while web services technologies based on XML and HTTP are now setting the standard for new generations of intra-organisational and inter-organisational distributed applications. In the eBusiness domain, frameworks such as RosettaNet and ebXML also add business rules and templates to define interactions and agreements between trading partners, while for scientific applications, grid computing architectures are making it increasingly possible to access remote resources and execute complex computational workflows. Finally, markup languages such as XML and RDF, along with standardised ontologies, provide a means for explicit resource description and manipulation of application data at a richer semantic level than previously possible.

An underlying trend across each of these areas is the move towards so called Service Oriented Architectures (SOAs) for distributed applications; that is, the design of systems based on components providing well defined computational services that can be aggregated dynamically at runtime to create new applications.

This technical context impacts on the development of agent systems in two major ways:

- Each of these technologies provides implementation methods for building multi-agent systems never before possible, providing for underlying infrastructural needs of agent-based systems, such as a standardised means for discovery and communication between heterogeneous services.
- Applications now enabled by these technologies are becoming increasingly agent-like, with many of the more complex technical problems similar to those that have

been addressed in the context of multi-agent systems, including issues of trust, reputation, obligations, contracts and analysing the dynamics of large-scale open systems.

In terms of providing potential infrastructures for the development of agent systems, technologies of particular relevance include:

- Base Technologies:
  - The Extensible Markup Language (XML) is the universal format for structured documents and data on the Web. It was designed for ease of implementation and for interoperability with both SGML and HTML.
  - The Resource Description Format (RDF) is a framework for describing and interchanging metadata.

- eBusiness:
  - ebXML aims to standardise XML business specifications by providing an open XML-based infrastructure enabling the global use of electronic business information in an interoperable, secure and consistent manner.
  - RosettaNet is a consortium of major technology companies working to create and implement industry-wide eBusiness process standards. RosettaNet standards offer a robust non-proprietary solution, encompassing data dictionaries, an implementation framework, and XML-based business message schemas and process specifications for eBusiness standardisation.

- Universal Plug & Play:
  - Jini network technology provides simple mechanisms that enable devices to plug together to form an emergent community in which each device provides services that other devices in the community may use.
  - uPnP offers pervasive peer-to-peer network connectivity of intelligent appliances and wireless devices through a distributed, open networking architecture to enable seamless proximity networking in addition to control and data transfer among networked devices.

- Web Services:
  - UDDI is an industry initiative aimed at creating a platform-independent, open framework for describing services, discovering businesses, and integrating business services using the Internet. It is a cross-industry effort driven by platform and software providers, marketplace operators and eBusiness leaders.

- SOAP provides a simple and lightweight mechanism for exchanging structured and typed information between peers in a decentralised, distributed environment using XML.
- WSDL/WSCL: WSDL provides an XML grammar for describing network services as collections of communication endpoints capable of exchanging messages, thus enabling the automation of the details involved in applications communication. WSCL allows the abstract interfaces of web services (that is, the business level conversations or public processes supported by a web service) to be defined.

Of course, there are other important developments that are not mentioned, simply because it is not possible to be exhaustive. For example, in addition to the areas mentioned above, ubiquitous infrastructures such as UMTS, GPRS and other 3G wireless technologies and beyond are increasingly prevalent.

Conversely, agent-related activities are already beginning to inform development in a number of these technology areas, including contributions to W3C Semantic Web standardisation, and OMG CORBA via an OMG special interest group on agent technology. Contributions have also come through the Foundation for Intelligent Physical Agents (FIPA), which defines a range of architectural elements directly related to work in the W3C Web Services Architecture specifications and elsewhere.

These positive moves with regard to the technological context for agents, are illustrated in Figure 2.1, where the years in which the main technologies that can facilitate agent-based systems development are noted. While research in agent technologies has been active for over a decade now, the figure shows that it is only as of 1999, with the appearance of effective service-oriented technologies (Jini) and pervasive computing technologies (Bluetooth) that truly dynamic networked systems could be built without big investments in establishing the underlying infrastructure. In particular, as of 2002, with the emergence of Grid computing and calls for adaptive wide-scale web service-based solutions there is now a real need to provide attractive solutions to the higher-level issues of communication, coordination and security.

In general, it is clear that current technological developments are increasingly addressing problems that have long been posed within the agent research community. In particular, they are responding to some of the underlying infrastructural needs for agent-based systems, such as a standardised means for discovery and communication between heterogeneous services. This suggests two clear and inter-related trends. First, supporting technologies are emerging very quickly. Thus, to some extent, the research problem has moved from one of infrastructure to one of the higher-level issues relating to effective coordination and cooperation between services. Second, a very large number of systems are being built and designed using these emerging infrastructures, and are becoming ever more agent-like — their developers face the same problems that the agent community has encountered in the past.

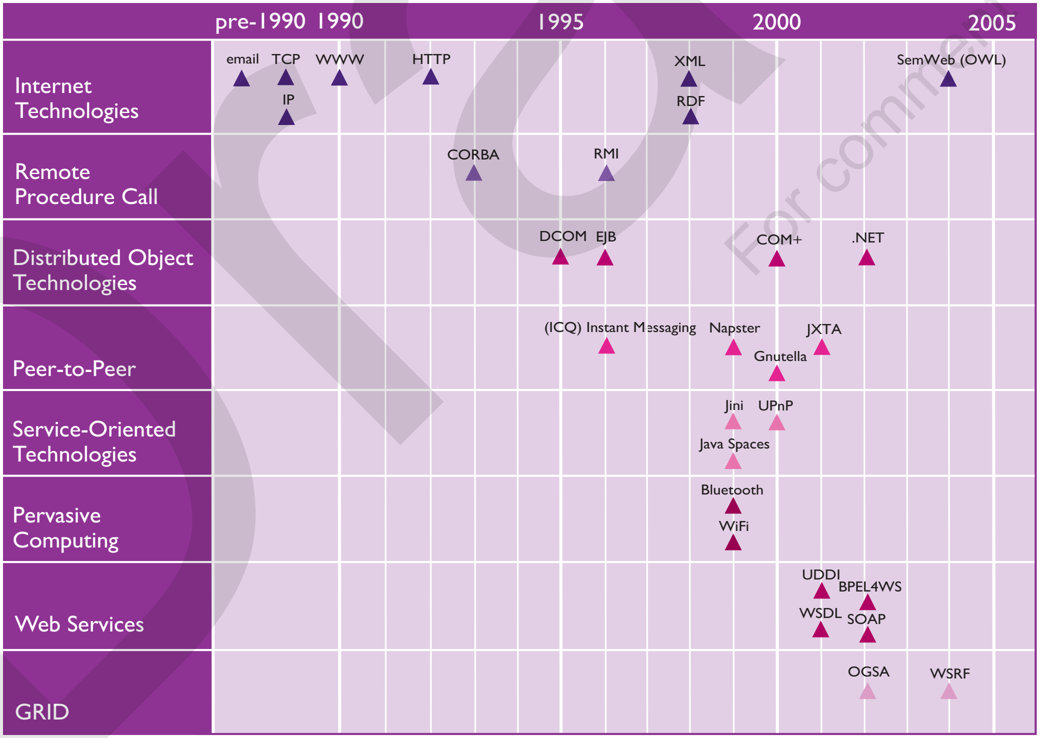


Figure 2.1: Agent-related technologies for infrastructure support

3 Emerging Trends and Critical Drivers

Against the background of the technological context described above, there are a number of visions of future computing paradigms that are relevant to the introdcution of agent technologies, both in demanding their use, and in providing an environment in which to operate. In this section, we consider the emergence of several key trends and drivers that suggest agents and agent technologies will be vital. The discussion is not intended to be exhaustive, but instead indicative of the current impetus for use and deployment of agent systems.

3.1 Semantic Web

Since it was first developed in the early 1990s, the *world wide web* has rapidly and dramatically become a critically important and powerful medium for communication, research and commerce. However, the Web was designed for use by humans, and its power is limited by the ability of humans to navigate the data of different information sources.

The *semantic web* is based on the idea that the data on the web can be defined and linked in such a way that it can be used by machines for the automatic processing and integration of data across different applications (Berners-Lee et al., 2001). This is motivated by the fundamental recognition that, in order for the Web to scale, programs must be able to share and process data, particularly when they have been designed independently. The key to achieving this is by augmenting web pages with descriptions of their content so that it is possible to reason about that content.

Among the particular requirements for the realisation of the semantic web vision are: rich descriptions of media and content to improve search and management; rich descriptions of web services to enable and improve discovery and composition; common interfaces to simplify integration of disparate systems; and a common language for the exchange of semantically-rich information between software agents.

It should be clear from this that the semantic web demands effort and involvement from the field of agent-based computing, and is intimately tied up with it. Indeed, it offers a rich breeding ground for both further fundamental research and a whole range of agent applications that can (and should) be built on top of it. In the first stage of web development, all web pages were created by humans and read by humans. In the second (current) stage, most web pages are created by machines (for example, automatically generated from database queries) but still read by humans. In the next stage, of the semantic web, most web pages will be created by machines and read by machines.

3.2 Web Services and Service Oriented Computing

Web services technologies provide a standard means of interoperating between different software applications, running on a variety of different platforms. Specifications cover a wide range of interoperability issues, from basic messaging, security and architecture, to service discovery and the composition of individual services into structured *workflows*. Standards for each of these areas, produced by bodies such as W3C and OASIS, provide a framework for the deployment of component services accessible using HTTP and XML interfaces. These components can subsequently be combined into loosely coupled applications that deliver increasingly sophisticated added-value services.

In a more general sense, web services standards serve as a potential convergence point for diverse technology efforts such as eBusiness frameworks (ebXML, rosettaNet and others), Grid architectures (which are now increasingly based on web services infrastructures) and others towards a more general notion of *service oriented architectures* (SOA) in which distributed systems are increasingly viewed as collections of service provider and service consumer components interlinked by dynamically defined workflows. Web services must be realised by concrete agents that send and receive messages, while the services themselves are the resources characterised by the functionality provided. In the same way as agents may perform tasks on behalf of a user, a web service provides this functionality on behalf of its owner, a person or organisation.

Web services thus provide a ready-made infrastructure that is almost ideal for use in supporting agent interactions in a multi-agent system. More importantly, perhaps, it is widely accepted, standardised, and likely to be the dominant base technology over the coming years. Conversely, an agent-oriented view of web services is gaining increased traction and exposure as provider and consumer web services environments are naturally seen as a form of agent (Booth et al., 2004).

3.3 Peer-to-Peer Computing

Peer-to-peer (P2P) computing covers a wide range of infrastructures, technologies and applications that share a single characteristic: they are designed to create networked applications in which every node (deployed system) is in some sense equivalent to all others, and application functionality is created by potentially arbitrary interconnection between these *peers*. The consequent absence of the need for centralised server components to manage P2P systems makes them highly attractive in terms of robustness against failure, ease of deployment and maintenance (Milojicic et al., 2002).

The best known P2P applications include hugely popular file sharing applications such as Gnutella and Bit Torrent, AKAMI content caching, groupware applications (such as Groove

Networks office environments) and Internet telephony applications such as Skype. While the majority of these well-known systems are based on proprietary protocols and platforms, toolkits such as Sun Microsystem's JXTA provide a wide array of networking features for the development of P2P applications, such as messaging, service advertisement and peer management features. Standardisation for P2P technologies is also underway within the Global Grid Forum, which now includes the P2P working group established by Intel in 2000.

P2P applications display a range of agent-like characteristics, often applying self-organisation techniques in order to ensure continuous operation of the network and relying on protocol design to ensure correct behaviours of clients. (For example, many systems include simple credit-reputation systems to reward *socially beneficial* behaviour). As P2P systems become more complex, an increasing number of agent technologies may become relevant. These include, for example: mechanism design to provide a rigorous basis for economic mechanisms applied to incentivise rational behaviour among clients in P2P networks; agent negotiation techniques to improve the level of automation of peers in popular applications; increasingly advanced theories of trust and reputation; and the application of social norms, rules and structures, as well as social simulation, in order to better understand the dynamics of populations of independent actors.

3.4 Grid Computing

The Grid is the high-performance computing infrastructure for supporting large-scale distributed scientific endeavour that has recently gained heightened and sustained interest from several communities (Foster and Kesselman, 2004). It provides a means of developing eScience applications such as those demanded by, for example, the Large Hadron Collider facility at CERN, engineering design optimisation, bioinformatics and combinatorial chemistry. Yet it also provides a computing infrastructure for supporting more general applications that involve large-scale information handling, knowledge management and service provision. Typically, Grid computing is abstracted into several layers, which might include: a data-computation layer dealing with computational resource allocation, scheduling and execution; an information layer dealing with the representation, storage and access of information; and a knowledge layer, which deals with the way knowledge is acquired, retrieved, published and maintained.

The Grid thus refers to an infrastructure that enables the integrated, collaborative use of high-end computers, networks, databases, and scientific instruments owned and managed by multiple organisations. Grid applications often involve large amounts of data and computing and often require secure resource sharing across organisational boundaries; they are thus not easily handled by today's Internet and Web infrastructures.

The UK's eScience programme has allocated £230M to Grid-related computing, while Germany's D-Grid programme has allocated €300M, and the French ACI Grid programme nearly €50M.

Utility Computing

The Internet has enabled computational resources to be accessed remotely. Networked resources such as digital information, specialised laboratory equipment and computer processing power may now be shared between users in multiple organisations, located at multiple sites. For example, the emerging Grid networks of scientific communities enable shared and remote access to advanced equipment such as supercomputers, telescopes and electron microscopes. Similarly, in the commercial IT arena, shared access to computer processing resources has recently drawn the attention of major IT vendors with companies such as HP ("utility computing"), IBM ("on-demand computing"), and Sun ("N1 Strategy") announcing initiatives in this area. Sharing resources across multiple users, whether commercial or scientific, allows scientists and IT managers to access resources on a more cost- match between demand and supply of resources. Ensuring efficient use of shared resources in this way will require design, implementation and management of resource-allocation mechanisms in a computational setting.

The key benefit of Grid computing more generally is flexibility – the distributed system and network can be reconfigured on demand in different ways as business needs change, in principle enabling more flexible IT deployment for more demanding problems and more efficient use of computing resources (Information Age Partnership, 2004). According to BAE Systems (Gould et al., 2003), while the technology is already in a state in which it can realise these benefits in a single organisational domain, the real value comes from cross-organisation use, through virtual organisations, which require ownership, management and accounting to be handled within trusted partnerships. In economic terms, such virtual organisations provide an appropriate way to develop new products and services in high value markets; this facilitates the notion of service centric software, which is only now emerging because of the constraints imposed by traditional organisations. As the Information Age Partnership (2004) suggests, the future of the Grid is not in the provision of computer power, but in the provision of information and knowledge in a service-oriented economy. Ultimately, the success of the Grid will depend on standardisation and productisation, and efforts in this direction are already underway through a range of vendors, including Sun, IBM, HP, and so on.

3.5 Ambient Intelligence

The notion of *ambient intelligence* has largely arisen through the efforts of the European Commission in identifying challenges for European research and development in Information Society Technologies (IST Advisory Group, 2002). Aimed at seamless delivery of services and applications, it relies on the areas of ubiquitous computing, ubiquitous communication and intelligent user interfaces. The vision describes an environment of potentially thousands of embedded and mobile devices (or software artefacts) interacting to support user-centred goals and activity. This suggests a component-oriented view of the world in which the artefacts are independent and distributed. The consensus is that autonomy, distribution, adaptation, responsiveness, and so on, are key characterising features of the components, and in this sense they share the same characteristics as agents.

Ambient intelligence requires these agents to be able to interact with numerous other agents in the environment around them in order to achieve their goals. Such interactions take place between pairs of agents (in one-to-one cooperation or competition), between groups (in reaching consensus decisions or acting as a team), and between agents and the infrastructure resources that comprise their environments (such as large-scale information repositories). Interactions like these enable the establishment of electronic institutions or virtual organisations, in which groups of agents come together to form coherent groups able to achieve overarching goals.

The environment provides the infrastructure that enables ambient intelligence scenarios to be realised. On the one hand, agents offering higher-level services can be distinguished from the physical infrastructure needed to support effective interaction through sensors and actuators, and the physical connectivity for supporting quick and efficient interactions, for example. On the other, they can also be distinguished from the virtual infrastructure needed to support resource discovery, large-scale distributed and robust information repositories (as mentioned above), and the logical connectivity needed to enable effective interactions between large numbers of distributed agents and services, for example.

In relation to pervasiveness, it is important to note that scalability (more particularly, device scalability), or the need to ensure that large numbers of agents and services are accommodated, and heterogeneity of agents and services, is facilitated by the provision of appropriate *ontologies* to enable effective interactions. Addressing all of these aspects will require efforts to provide solutions to issues of operation, integration and visualisation of distributed sensors, ad hoc services and network infrastructure.

3.6 Self-* Systems and Networks and Autonomic Computing

Computational systems able to manage themselves have been part of the vision for computer science since the work of Charles Babbage. With the increasing complexity of advanced information technology systems, and the increasing reliance of modern society on these systems, attention in recent years has returned to this. Such systems have come to be called *self-** systems and networks, with the asterisk indicating that a variety of attributes are under consideration. While an agreed definition of self-* systems is still emerging, aspects of these systems include properties such as: self-awareness, self-organisation, self-configuration, self-management, self-diagnosis, and self-repair.

Such systems abound in nature, from the level of ecosystems, through large primates (such as man) and down to processes inside single cells. Similarly, many chemical, physical, economic and social systems exhibit self-* properties. Thus, the development of computational systems that have self-* properties is increasingly drawing on research in biology, ecology, statistical physics and the social sciences. Recent research on computational self-* systems has tried to formalise some of the ideas from these different disciplines, and to identify algorithms and procedures that could realise various self-* attributes, for example in peer-to-peer networks. One particular approach to self-* systems has become known as *autonomic computing*, considered below.

Computational self-* systems and networks provide an application domain for research and development of agent technologies, and also a contribution to agent-based computing theory and practice, because many self-* systems may be viewed as involving interactions between autonomous entities and components.

More specifically, in response to the problems relating to the explosion of information and integration of technology into everyday life, and the associated problems of complexity in managing and operating computer systems, *autonomic computing* draws on the autonomic function of the human central nervous system, which controls key functions without conscious awareness or involvement. First proposed by IBM [2], autonomic computing is an approach to self-managed computing systems with a minimum of human interference. Its goal is a network of sophisticated computing components that give users what they need, when they need it, without a conscious mental or physical effort. Among the defining characteristics of an autonomic system are the following: it must automatically configure and reconfigure itself under varying (and unpredictable) conditions; it must seek to optimise its operation, monitoring its constituent parts and fine-tuning its workflow to achieve system goals; It must be able to discover problems and recover from routine and extraordinary events that might cause malfunctions; it must act in accordance with its current environment, adapting to best interact with neighbouring systems, by negotiating for resource use with other systems; it must function in a heterogeneous world and implement open standards; and it must marshal resources to reduce the gap between its (user) goals and their achievement, without direct user intervention.

Ultimately, the goal is to realise the promise of IT: increasing productivity while minimising complexity for users. The key message to be drawn from this vision is that it shares many of the goals of agent-based computing, and agents offer a way to manage this complexity.

3.7 Complex Systems

Modern software and technological systems are among the most complex human artefacts, and are ever-increasing in complexity. Some of these systems, such as the Internet, were not designed but simply grew organically, with no central human control or even understanding. Other systems, such as global mobile satellite communications networks or current PC operating systems, have been designed centrally, but comprise so many interacting components and so many types of interactions, that no single person or even team of people could hope to comprehend the detailed system operations. This lack of understanding may explain why such systems are prone to error as, for example, in the large-scale electricity network failures in North America and in Italy in 2003.

Moreover, many systems that affect our lives involve more than just software. For example, the ecosystem of malaria involves natural entities (parasites and mosquitos), humans, human culture, and technological artefacts (drugs and treatments), all interacting in complex, subtle and dynamic ways. Intervening in such an ecosystem, for example by providing a new treatment regime for malaria, may have unintended and unforeseen consequences due to the little-understood nature of these interactions. The science of complex adaptive systems is still in its infancy, and provides little yet in the way of guidance for designers and controllers of specific complex systems.

Whether such complex, adaptive systems are designed or not, their management and control is vitally important to modern societies. Agent technologies provide a way to conceptualise these systems as comprising interacting autonomous entities, each acting, learning or evolving separately in response to interactions in their local environments. Such a conceptualisation provides the basis for realistic computer simulations of the operation and behaviour of the systems, and of design of control and intervention processes (Bullock and Cliff, 2004). For systems that are centrally designed, such as electronic markets overlaid on the Internet, agent technologies also provide the basis for the design and implementation of the system itself. Indeed, it has been argued that agent technologies provide the only way to cope with the increasing complexity of modern software systems (Zambonelli and Parunak, 2003): pervasive devices, ambient intelligence, continuous operation (allowing no downtime for upgrades or maintenance), and open systems.

3.8 Summary

It is natural to view large systems in terms of the services they offer, and consequently in terms of the entities or agents providing or consuming services. The domains discussed here reflect the trends and drivers for applications in which typically many agents and services may be involved, and spread widely over a geographically distributed environment. Figure 3.1 depicts the emergence of these driver domains over time, illustrating their currency, and suggesting that their maturity, which will demand the use of agent technologies, is likely to be some years away.

Most importantly perhaps, the environments that have been identified here are open and dynamic so that new agents may join and existing ones leave. In this view, agents act on behalf of service owners, managing access to services, and ensuring that contracts are fulfilled. They also act on behalf of service consumers, locating services, agreeing contracts, and receiving and presenting results. In these domains, agents will be required

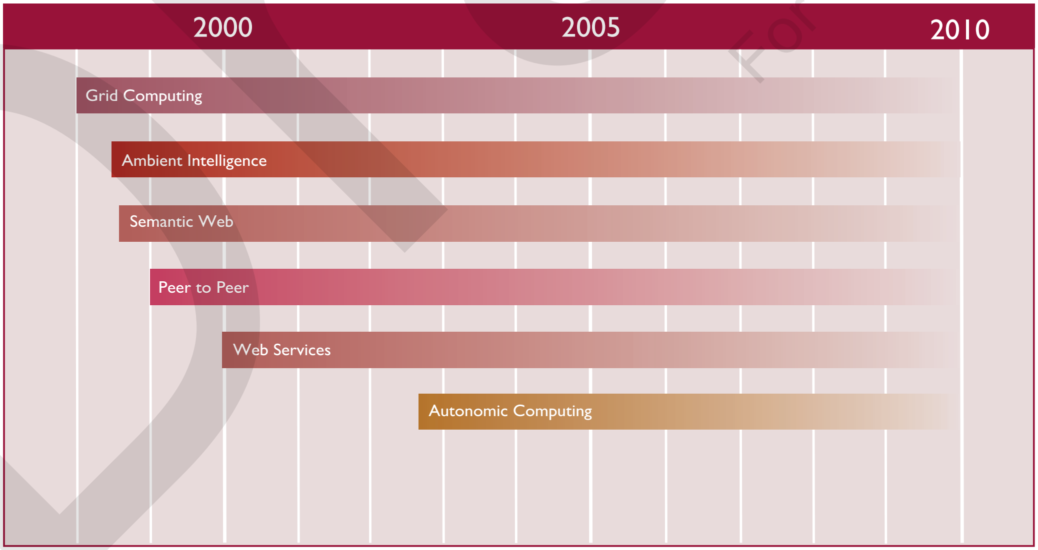


Figure 3.1: The emergence of agent-related domains over time.

to engage in interactions, to negotiate, and to make pro-active run-time decisions while responding to changing circumstances. In particular, agents will need to collaborate and to form coalitions of agents with different capabilities in support of new virtual organisations.

Of course, these drivers do not address a whole range of specific areas within the field of agent-based computing. For example, there is a need for systems that can behave intelligently and work as part of a community, replacing humans in environments that are dirty, dull or dangerous. There are also drivers relating to human-agent interfaces, learning agents, robotic agents, and many others, but those identified here provide a context that is likely to drive forward the whole field.

4 Agent Technologies, Tools and Techniques

In seeking to realise the visions above, of a world comprising large numbers of interconnected devices and services, many technological areas within the field of agent-based computing are relevant. Although it isn't yet obvious whether the label of agent-based computing will persist through the next 10 or 20 years, it is clear that different subsets of the specific technologies within the field will be needed. Below, we review several of the key agent technologies that are likely to be enablers for these next generation computing systems, as part of complete agent systems, or as individual components contributing to some other paradigm. Those discussed here have been identified by the AgentLink community as being particularly important over the next 3 to 15 years.

4.1 Organisations

Dynamic agent organisations that adjust themselves to gain advantage in their current environments are likely to become increasingly important over the next five years. This will arise in dynamic (or emergent) agent societies, which will require that agents can adapt to function effectively in uncertain or hostile environments. Some work has already started on the development of systems that can meet this challenge, but it is fundamental to realising the power of the agent paradigm, and its relevance will remain at the forefront of R&D efforts over the next 10-15 years, especially in relation to commercial efforts at exploitation. In particular, building dynamic agent organisations (including, for example, teamwork, coalition formation, and so on) for dealing with aspects of the emerging visions of the Grid and the Web, as well as aspects of ubiquitous computing, will be crucial.

Social factors in the organisation of multi-agent systems will also become increasingly important over the next 10 years as we seek ways to structure interactions in an open and dynamic world. This relates to the need to properly assign roles, (institutional) powers, rights and obligations to agents in order to control security and trust-related aspects of multi-agent systems at a semantic level as opposed to current developments, which deal with them at the infrastructure level.

4.2 Coordination

Coordination is defined in many ways but in its simplest form it refers to ensuring that the actions of independent actors (agents) in an environment are coherent in some way. The challenge therefore is to identify mechanisms that allow agents to coordinate their actions automatically without the need for human supervision, a requirement found in a wide variety of real applications. Cooperation in turn refers to coordination with a common goal in mind.

Research to date has identified a huge range of different types of coordination and cooperation mechanisms, ranging from emergent cooperation (which can arise without any explicit communication between agents), coordination protocols (which structure interactions to reach decisions), coordination media (or distributed data stores that enable asynchronous communication of goals, objectives or other useful data), to distributed planning (which takes into account possible and likely actions of agents in the domain).

4.3 Negotiation

Future research in multi-agent resource allocation should emphasise the interdisciplinary character of the field and bring together ideas from computer science and artificial intelligence on the one hand, and the socio-economic sciences on the other. A typical objective in multi-agent resource allocation is to find an allocation that is optimal with respect to a suitable metric that depends, in one way or another, on the preferences of the individual agents in the system. A wide range of concepts studied in social choice theory can (and should) be utilised to assess the quality of resource allocations. Of particular importance are concepts such as envy-freeness and equitability that can be used to model fairness considerations (Arrow et al., 2002; Brams & Taylor, 1996; Endriss & Maudet, 2004). These concepts are relevant to a wide range of applications. A good example is the work on the fair and efficient exploitation of Earth Observation Satellite resources carried out at ONERA, the French National Aeronautics Research Centre in Toulouse (Lemaître et al., 2003).

While much recent work on resource allocation has concentrated on centralised approaches, in particular combinatorial auctions (Cramton et al., 2005), many applications are more naturally modelled as truly distributed systems where allocations emerge as a consequence of a sequence of local negotiation steps (Chevaleyre et al., 2005). The centralised approach has the advantage of requiring only comparatively simple communication protocols. Furthermore, recent advances in the design of powerful algorithms for combinatorial auctions have had a strong impact on the research community (Fujishima et al., 1999; Sandholm, 2002). A new challenge in the field of multiagent resource allocation is to transfer these techniques to distributed resource allocation frameworks, which are not only important in cases where it may be difficult to find an agent that could take on the role of the auctioneer (for instance, in view of its computational capabilities or in view of its trustworthiness), but which also provides a test-bed for a wide range of agent-based techniques. To reach its full potential, distributed resource allocation requires further fundamental research into agent interaction protocols, negotiation strategies, formal (e.g. complexity-theoretic) properties of resource allocation frameworks, and distributed algorithm design.

4.4 Agent Communication

Agent communication and interaction is the study of how two or more software entities may interoperate and communicate with one another. As in human communications theory, issues of syntax (the form of communication signals), semantics (the context-independent meaning of signals) and pragmatics (how signals are used in interaction contexts) have been considered. Agent researchers have therefore drawn on human linguistic theory and the philosophy of language to design and study agent communication languages and protocols. One important source has been speech act theory, from the philosophy of language, in which utterances are understood in terms of their impact on states of the world, including the mental states of any agent who hears the utterance. The agent communications language of the Foundation for Intelligent Physical Agents (FIPA ACL), for example, has been given a speech-act based semantics formalised in modal logic. Another rich source of ideas has been argumentation theory and the study of dialogue games from the philosophy of argument, in which rule-governed interactions between autonomous entities are designed and studied. Such interactions may, for example, involve negotiations over the division of a scarce resource, or persuasion dialogues over beliefs, etc. In addition, because software agents are computational entities, their utterances may be viewed as statements in computer programs, and so the theory of programming languages and their semantics has also been influential in this area.

The challenges in the domain are long-standing and deep. One challenge is the difficulty of assigning meaning to utterances, since the precise meaning of a statement will depend upon: the context in which it is uttered; its position in a sequence of previous utterances; the nature of the statement (for example, a proposition, a commitment to undertake some action, a request, etc); the objects referred to in the statement (such as a real world object, a mental state, a future world-state, etc); and the identity of the speaker and of the intended hearers. Another challenge, perhaps insurmountable, is semantic verification: how to verify that an agent means what it says when it makes an utterance. In an open agent system, one agent will not normally be able to view the internal code of another agent in order to verify an utterance by the latter; even if this were possible, a sufficiently-clever agent could always simulate any desired mental state when inspected by another agent.

A key research challenge in this area is to map the relevant theories in the domain, and to develop a unifying framework for them. In particular, a formal theory of agent languages and protocols is necessary, so as to be able to study language and protocol properties comprehensively, and to compare rigorously one language or protocol with another. In addition, progress towards understanding the applicability of different agent communication languages and protocols in different application domains is necessary for wider adoption of research findings.

4.5 Complex Systems and Self Organisation

Self-organisation refers to the process by which a system changes its internal organisation to adapt to changes in its goals and environment without explicit external control, often resulting in emergent behaviour that may or may not be desirable. Due to the dynamism and openness of contemporary computing environments, understanding the mechanisms that can be used to model, assess and engineer self-organisation and emergence in multi-agent systems is an issue of major interest.

A self-organising system functions without central control, and through contextual local interactions. Components aim to individually achieve simple tasks, but a complex collective behaviour emerges from their mutual interactions. Such a system modifies its structure and functionality to adapt to changes to requirements and to the environment based on previous experience. Nature provides examples of self-organisation, such as ant food foraging, molecule formation, or antibody detection. Similarly, current software applications are driven by social interactions (negotiations, transactions), based on autonomous entities or agents, and run in highly dynamic environments. The issue of engineering applications, based on the principles of self-organisation, to achieve robustness and adaptability, is thus gaining increasing interest in the software community. This trend originates from the fact that current software applications need to cope with requirements and constraints stemming from the increased dynamism, sophisticated resource control, autonomy and decentralisation inherent in contemporary business and social environments. The majority of these characteristics and constraints are the same as those that can be observed in natural systems exhibiting self-organisation.

Self-organisation mechanisms provide the decision-making engines based on which system components process input from software and hardware sensors to decide how, when and where to modify the system's structure and functionality. This enables a better fit with the current requirements and environment, while preventing damage or loss of service. It is therefore necessary to characterise the applications in which existing mechanisms, such as stigmergy, can be used, and to develop new generic mechanisms independent of any particular application domain.

In some cases, self-organisation mechanisms have been modelled using rule-based approaches or control theory. Furthermore, on many occasions the self-organising actions have been inspired from biological and natural processes, such as the human nervous system and the behaviour observed in insect species that form colonies. Although such approaches to self-organisation have been effective in certain domains, environmental dynamics and software complexity have limited their general applicability. More extensive research in modelling self-organisation mechanisms and systematically constructing new ones is therefore needed. Future self-organising systems must accommodate highdimensional sensory data, continue to learn from new experiences and take advantage of new self-organisation acts and mechanisms as they become available.

A phenomenon is characterised as emergent if it has not been exactly predefined in advance. Such a phenomenon can be observed at a macro system level and it is generally characterised by novelty, coherence, irreducibility of macro level properties to micro-level ones and non-linearity. In multi-agent systems, emergent phenomena are the global, not-exactly-predetermined overall system behaviours, which are collective results originating from the local agent interactions and individual agent behaviours. Emergent behaviours can be desirable or undesirable; building systems with desirable emergent behaviour capabilities increases their robustness, autonomy, openness and dynamism.

To achieve desired global emergent system behaviour, local agent behaviours and interactions should comply with some behavioural framework dictated by a suitable theory of emergence. Unfortunately, too few theories of emergence are currently available and existing ones still require improvement. Therefore, new theories of emergence need to be developed based on inspiration from natural systems, for example.

An important open issue in self-organising systems relates to modelling the application context and environment. In this respect, a key question is the definition of the relevant environmental parameters that need to be considered in determining the evolving structure and functionality of self-organising software. Additional open questions relate to how context can be captured, processed and exploited for adjusting the services provided by the application in a given situation, how the self-organising effects occurring from participation of the application in different contexts can be synchronised, how to effectively model user preferences and intentions and the amount of historical information that should be recorded by the system and considered in determining its evolution over time.

4.6 Trust and Reputation

Many applications involving multiple individuals or organisations must take into account the relationships (explicit or implicit) between participants. Furthermore, individual actors may also need to be aware of these relationships in order to make appropriate decisions. The field of trust, reputation and social structure seeks to capture human notions such as trust, reputation, dependence, obligations, permissions, norms, institutions and other social structures in electronic form.

By modelling these notions, engineers can borrow strategies commonly used by humans to resolve conflicts that arise when creating distributed applications, such as regulating the actions of large populations of actors using financial disincentives (or punishments) for breaking social rules or devising market mechanisms that are proof against certain types of malicious manipulation. The theories are often based on insights from different domains including economics (market based approaches and mechanism design), other social sciences (social laws, social power) or mathematics (game theory).

The complementary aspect of this social perspective relating to reputation and norms is a traditional concern with security. Although currently deployed agent applications often provide good security, for agents autonomously acting on behalf of their owner, several additional factors need to be addressed. First, considerable effort must still be put into issues of security in open agent systems. Efforts by other communities are tackling some aspects here, but more on specific agent security concerns needs to be done. Second, collaboration of any kind, especially in situations in which computers act on behalf of users or organisations, will only succeed if there is trust. Ensuring this trust requires, for example, the use of: reputation mechanisms to assess prior behaviour; norms (or social rules) and the enforcement of sanctions; and electronic contracts to represent and enforce agreements.

Whereas assurance deals primarily with system integrity, security addresses protection from malicious entities: preventing would-be attackers from exploiting self-organisation mechanisms that alter system structure and behaviour. In addition, to verify component sources, a self-organising software system must protect its core from attacks. Various well studied security mechanisms are available, such as strong encryption to ensure confidentiality and authenticity of messages related to self-organisation. However, the frameworks within which such mechanisms can be effectively applied in self-organising systems still require considerable further research.

In addition, the results of applying self-organisation and emergence approaches over long time periods lead to concerns about the privacy and trustworthiness of such systems and the data they hold. The areas of security, privacy and trust are critical components for the next stages of research and deployment of open distributed systems and as a result of self-organising systems. New approaches are required to take into account both social and technical aspects of this issue to drive the proliferation of self-organising software in a large range of application domains.

4.7 Reasoning and Learning

Reasoning is a critical faculty of agents, but the extent to which it is needed is determined by the context. While reasoning in general is important, in open environments there are some specific concerns relating to heterogeneity of agents, trust and accountability, failure handling and recovery, and societal change. Work must be continued on the representation of computational concepts for the norms, legislation, authorities, enforcement, and so forth, that can underpin the development and deployment of dynamic electronic institutions. Similarly, current work on coalition formation for virtual organisations is limited, with such organisations largely static. The automation of coalition formation may be more effective at finding better coalitions than humans in complex settings, and is required, for example, for Grid applications.

One enabler for this is negotiation, yet research into negotiation mechanisms that are more complex than auctions and game-theoretic mechanisms is still in its infancy. Research into argumentation mechanisms, for example, and the strategies appropriate for participants under them, is also needed before these mechanisms will achieve widespread deployment. In addition, many virtual organisations will be required to make decisions collectively, aggregating in some fashion the individual preferences or decisions of the participants. Research on the application to agent societies of social choice theory from political science and sociology is in its infancy, and considerably more work is needed here.

Even though learning technology is clearly crucial for open and scalable multi-agent systems, it is still in early development. While there has been progress in many areas, such as evolutionary approaches and reinforcement learning, these have still not made the transition to real-world applications. Reasons for this can be found in problems of scalability and in user trust in self-adapting software. In the longer term, learning techniques are likely to become a central part of agent systems, while the shorter term offers application opportunities in areas such as interactive entertainment, which are not safety-critical.

4.8 Infrastructure

Any infrastructure deployed to support the execution of agent applications, such as those found in ambient and ubiquitous computing must, by definition be long lived and robust. In the context of self-organising systems, this is further complicated, and new approaches supporting the evolution of the infrastructures and facilitating their upgrade and update at runtime, will be required. Given the potentially vast collection of devices, sensors, and personalised applications for which agent systems and self-organisation may be applicable, this update problem is significantly more complex than so far encountered. More generally, middleware, or platforms for agent interoperability, will be crucial for the medium terms development of agent systems.

4.9 Interoperability

At present, information agents exist in academic and commercial laboratories, but are not widely available in real world applications. The move out of the laboratory is likely to happen over the next ten years, but a much higher degree of automation than is currently available in dealing with knowledge management is needed for information agents. In particular, this demands new web standards that enable structural and semantic description of information; and services that make use of these semantic representations for information access at a higher level. The creation of common ontologies, thesauri or knowledge bases play a central role here, and merits further work on the formal descriptions of information and, potentially, a reference architecture to support the higher level services mentioned above.

Distributed agent systems that self-organise to adapt to their environment must both adapt individual agent components and coordinate adaptation across system layers (i.e. application, presentation and middleware) and platforms. In other words interoperability must be maintained across possibly heterogeneous agent components during and after the self-organisation actions and outcomes. Furthermore, agent components are likely to come from different vendors and hence the developer may need to integrate different self-organisation mechanisms to meet an application's requirements. The problem is further complicated by the diversity of self-organisation approaches applicable at different system layers. In many cases, even solutions within the same layer are often not compatible. Consequently, developers need tools and methods to integrate the operation of self-organising agent components across the layers of a single system, among multiple computing systems, as well as, between different self-organisation frameworks.

4.10 Software Engineering

Despite a number of languages, frameworks, development environments, and platforms that have appeared in the literature, implementing multi-agent systems is still a complex task. In part, to manage multi-agent systems complexity, the research community has produced a number of methodologies that aim to structure agent development. However, even if practitioners follow such methodologies during the design phase, there are difficulties in the implementation phase, partly due to the lack of maturity in both methodologies and programming tools. There are also difficulties in implementation due to a lack of specialised debugging tools, skills needed to move from analysis and design to code, the problems associated with awareness of the specifics of different agent platforms, and in understanding the nature of what is a new and distinct approach to systems development.

In relation to open and dynamic systems, new methodologies for systematically considering self-organisation are required. These methodologies should be able to provide support for all phases of the agent-based software engineering lifecycle, allowing the developer to start from requirements analysis, identify the aspects of the problem that should be addressed using self-organisation and design and implement the self-organisation mechanisms in the behaviour of the agent components. Such methodologies should also encompass techniques for monitoring and controlling the self-organising application or system once deployed.

In general, Integrated Development Environment (IDE) support for developing agent systems is rather weak, and existing agent tools do not offer the same level of usability as state-of-the-art object oriented IDEs. One main reason for this is the current unavoidable tight coupling of agent IDEs and agent platforms, which results from the variety of agent models, platforms and programming languages.

With existing tools, multi-agent systems often generate a huge amount of information related to the internal state of agents, messages sent and actions taken, yet there are not yet adequate methods for managing this information in the context of the development process. This impacts both dealing with the information generated in the system and obtaining this information without altering the design of the agents within it. Platforms like JADE provide general introspection facilities for the state of agents and for messages, but they enforce a concrete agent architecture that may not be appropriate for all applications. Thus, tools for inspecting any agent architecture, as can be done with remote debugging in current developments, are needed. Extending this to address other issues related to debugging for organisational features, and for considering issues arising from emergence in self-organising systems will also be important in the longer term. The challenge is relevant now, but will grow in importance as the complexity of installed systems increases further.

With self-organising agent applications, the inherent complexity also demands a new generation of CASE tools to assist application designers in harnessing the large amount of information involved by providing reasoning at appropriate levels of abstraction, automating the design and implementation process as much as possible and allowing for the calibration of deployed multi-agent systems by simulation and run-time verification and control.

More generally, there is a need to integrate existing tools into IDEs rather than starting from scratch. At present there are many research tools, but little that integrates with generic development environments, such as Eclipse; such advances would boost agent development and reduce implementation costs. Indeed, developing multi-agent systems currently involves higher costs than using conventional paradigms due to the lack of supporting methods and tools. Today, most of the cost of agent systems development lies in the implementation stage, as opposed to standard software engineering where the effort applied to implementation can be as little as 20% of the total effort.

The next generation of computing system is likely to demand large numbers of interacting components, be they services, agents or otherwise. Current tools work well with limited numbers of agents, but are generally not yet suitable for the development of large-scale (and efficient) agent systems, nor do they offer development, management or monitoring facilities able to deal with large amounts of information or tune the behaviour of the system in such cases.

We also need metrics for agent-oriented software: engineering always implies some activity of measurement, and traditional software engineering already uses widely applied measuring methods to quantify aspects of software such as complexity, robustness and mean time between failures. However, the dynamic nature of agent systems, and the generally non-deterministic behaviour of *self-organising* agent applications deem traditional techniques for measurement and evaluation inappropriate. Consequently, new measures and techniques for both quantitatively and qualitatively assessing and classifying self-organising multi-agent systems applications are needed.

4.11 Programming Languages

Most research in agent-oriented programming languages is based on declarative approaches, mostly logic based. Imperative languages are not common, since they are applied mainly to build frameworks, not agent languages. From the research perspective, the design and development of agent-based languages is also important. Currently, real agent-oriented languages (such as BDI-style ones) are limited, and used largely for research purposes; apart from some exceptions, they remain unused in practice.

Current research emphasises the role of multi-agent systems development environments to assist in the development of complex multi-agent systems, new programming principles to model and realise agent features, and formal semantics for agent programming languages to implement specific agent behaviours.

A programming language for multi-agent systems should respect the principle of separation of concerns and provide dedicated programming constructs for implementing individual agents, their organisation, their coordination, and their environment. However, due to the lack of dedicated agent programming languages and development tools, the construction of multi-agent systems is still a time-consuming and demanding activity.

One key challenge in agent-oriented programming is to define and implement some truly agent-oriented languages that integrate concepts from both declarative and object programming, to allow the definition of agents in a declarative way, yet supported by serious monitoring and debugging facilities. These languages should be highly efficient, provide interfaces to existing mainstream languages for easy integration with code and legacy packages. This challenge is already relevant, but it will become more so in the next 2 or 3 years; such agent-oriented languages are expected in 5-7 years.

In addition to languages for single agents, we also need languages for high-level programming of multi-agent systems. In particular, the need for expressive, easy-to-use, and efficient languages for coordinating and orchestrating intelligent heterogeneous components is already pressing and, although much research is already being done, the development of an effective programming language for coordinating huge, open, scalable and dynamic multi-agent systems composed of heterogeneous components will require 8-10 years.

At present, agent programming languages are tied to concrete theoretical models of how a multi-agent system operates. This is an advantage when performing automated analysis methods to determine if the system is functioning properly, but it has drawbacks in that these models force developers to create systems in ways that may not be natural in the view of traditional programming practice, for example by successively refining the code of the agent system.

Ideally, a developer would create such a system using a language that incorporated specific agent primitives but without overly constraining the developer. During implementation, or testing, a developer might then choose those models most relevant to the application and focus on the specific agent aspects. However, achieving this is not trivial, since it suggests a decoupling between theory and language that would then be difficult to recouple.

4.12 Formal Methods

While the notion of an agent acting autonomously in the world is intuitively simple, formal analysis of systems containing multiple agents is inherently complex. In particular, to understand the properties of systems containing multiple actors, powerful modelling and reasoning techniques are needed to capture possible evolutions of the system. Such techniques are required if agents and agent systems are to be modelled computationally.

Research in the area of formal models for agent systems attempts to represent and understand properties of the systems through the use of logical formalisms describing both the mental states of individual agents and the possible interactions in the system. The logics used are often higher order logics of belief or other modalities, along with temporal modalities, all of which require efficient algorithms when applied to problems of significant scale. Recent efforts have used logical formalisms to represent interactive properties, such as coalitions of agents and game-type properties.

It is clear that formal techniques such as model checking are needed to test, debug and verify properties of implemented multi-agent systems. Despite progress, there is still a real need to address the issues that arise from differences in agent systems, in relation to the paradigm, the programming languages used, and especially the design of self-organising and emergent behaviour. For the latter, a programming paradigm that supports automated checking of both functional and non-functional system properties may be needed. This would lead to the need to certify agent components for correctness with respect to their specifications. Such a certification could be obtained either by selecting components that have already been verified and validated offline using traditional techniques such as inspection, testing and model checking or by generating code automatically from specifications. Furthermore, techniques are needed to ensure that the system still executes in an acceptable, or safe, manner during the adaptation process, for example using techniques such as dependency analysis or high level contracts and invariants to monitor system correctness before, during and after adaptation.

4.13 Simulation

As mentioned above, agent-based computing provide a means to simulate both natural and artificial systems, including agent-based computational systems themselves. Such simulation modelling is increasingly providing guidance to decision-makers in areas of medicine, social policy and industrial engineering, and assisting in the design, implementation and management of artificial and computational systems. However, for the full potential of agent-based (or individual-based) simulation models to be realised, a number of research and development challenges need to be met. First among these is the development of a rigorous theory of simulation. When should one stop refining a simulation model, for example? How many iterations of a randomised simulation model or scenarios are required in order to have confidence in the results? How much detail is required to be simulated in a model? How much trust should be placed in the results? The answers to these questions are likely to depend on the application domain, so a single, unified theory may be impossible to achieve. But efforts towards this goal are needed, not least because of the increasing reliance placed on simulation models in important public policy decisions, such as those of the Kyoto Treaty on Climate Change.

Another major challenge relates to the development of agent-based simulation models involving cognitive and rational agents. In economic systems, for example, it has long been known that the expectations of individual actors may influence their behaviour, and thus the global properties of the system. How may these anticipatory and reflective aspects of real-world societies be modelled by agent based simulation models? The rapid growth of online resource allocation systems, such as Grid systems, makes this an important issue. If a computational Grid comprises intelligent computational users, many of whom base their decisions on their own economic models of the Grid operation itself, then the task of management is complicated immensely: statements and actions by the system manager may impact the beliefs and intentions of the participants, and thus impact system operations and performance. The challenge of managing user expectations in this way is well-known to governors of central banks, such as the European Central Bank, as they try to manage national monetary policy. The theory and practice of agent simulation models are not sufficiently mature to provide guidance to managers in this task.

4.14 User Interaction Design

In future complex environments, human involvement in complex systems is likely to become more important, yet this requires the exploration and understanding of several new possibilities, including: autonomy and improvisation (to deal with unforeseen events, like behaviour of human users); a standardised agent communication language with a powerful semantics to drive some of agent behaviour and facilitate integration of human users; social and organisational models for multi-agent systems, in which programs and humans can naturally interact. In addition, as software becomes self-organising to fit in a variety of contexts, a new set of issues concerning the interaction with users is created. A key question here is how people can interact with continuously changing software. Additional questions concern whether it would be valuable to try to design implicit interaction with applications operating on indirect sensor-based input and in that case how could users migrate from traditional explicit to future implicit interaction.

5.1 Diffusion of Innovations

In order to understand the current commercial position of agent technologies it is useful to know something about the diffusion of new technologies and innovations. This is a subject long-studied by marketing theorists (Rogers, 1962; Midgley, 1977) drawing on mathematical models from epidemiology and hydrodynamics. We begin by considering some relevant concepts.

5.2 Product Life Cycles

Most marketers believe that all products and services are subject to life cycles: sales of a new product or service begin with a small number of customers, grow to a peak at some time, and then decline again, perhaps to zero, as shown in Figure 5.1 (Levitt,1965). Growth occurs because increasing numbers of customers learn about the product and perceive that it may satisfy their needs (which may be diverse). Decline eventually occurs because the market reaches saturation, as potential customers have either decided to adopt the product or have found other means to satisfy their needs, and/or because the needs of potential customers change with time. Most high-technology products are adopted

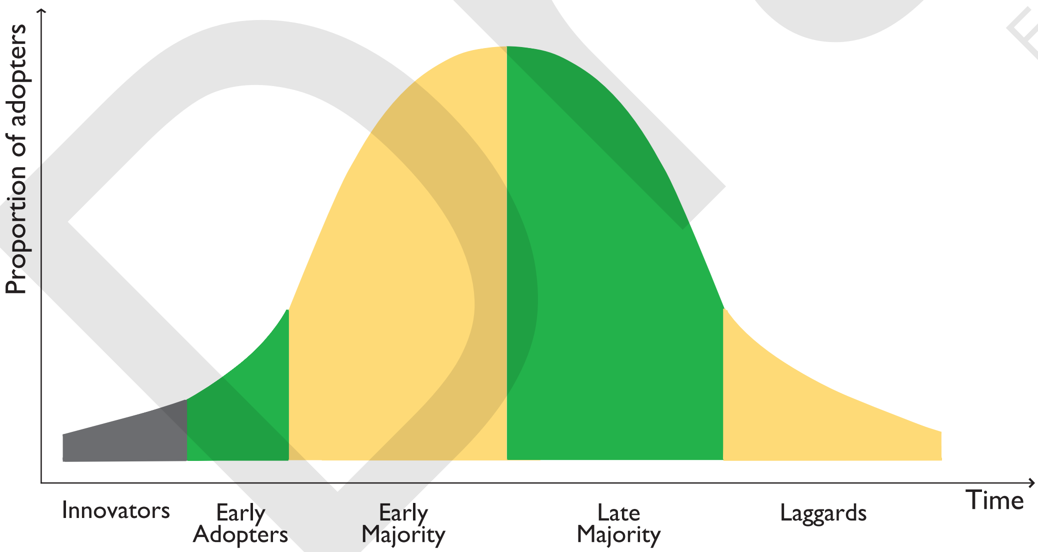


Figure 5.1: The technology adoption life cycle

initially only by people or companies with a keen interest in that type of new technology and the disposable income to indulge it. Thus, early adopters are often technologically sophisticated, well-informed, wealthy and not averse to any risks potentially associated with use of a new product.

Why does a product life cycle exist? In other words, why is it that all the companies or people who will eventually adopt the technology, product or process do not do so immediately. There are a number of reasons for this:

- Potential adoptees have to learn about the new technology before they can consider adopting it. Thus, there needs to be an information diffusion process ahead of the technology diffusion process.
- In addition, for non-digital products and services, the supplier needs to physically distribute the product or service. Establishing and filling sales channels may take considerable time and effort, and thus delay uptake of the product or service.
- Once they learn about a new technology, not all eventual adoptees will have the same extent of need for the product. The early adopters are likely to be those with the most pressing needs, which are not currently satisfied by competing or alternative technologies. The early adopters of supercomputers, for instance, were organisations with massively large-scale processing requirements, such as research physicists, meteorologists, and national census bureaux; later users included companies with smaller, but still large-scale processing requirements, such as econometric forecasting firms and automotive engineering design studios.
- Of those potential adoptees with a need, not all will have the financial resources necessary to adopt the new technology. Most new technologies, products and processes are expensive (relative to alternatives) when first launched. But prices typically fall as the base of installed customers grows, and as new suppliers enter the marketplace, attracted by the growing customer base. Thus, later adopters typically pay less than do early adopters for any new technology. Likewise, the total costs of adoption also typically fall, as complementary tools and products are developed in tandem with a new technology. If a company's needs are not pressing, it may benefit by waiting for the price and other adoption costs to fall before adopting.
- Similarly, not all potential adoptees share the same attitudes to technological risk. The risks associated with adopting a new technology also typically fall as bugs are eliminated, user-friendly features added, and complementary tools and products developed. Each subsequent release of an operating system, such as Windows or Linux, for example, has entailed lower risks to users of unexpected losses of data, obscure hardware incompatibilities, exception conditions, etc.
- Finally, for many advanced technologies and products, the value to any one adopter

depends on how many other adopters there are. These so-called *network goods* require a critical mass of users to be in place for the benefits of the technology to be fully realisable to any one user. For example, a fax machine is not very useful if only one company purchases one; it will only become useful to that company as and when other companies in its business network also have them.

These reasons for the existence of product life cycles mean that companies or people who adopt a new technology or purchase a new product later in its lifecycle may do so for very different reasons than do the early adopters; later adopters may even have different needs being satisfied by the product or technology. For example, in most countries the first adopters of mobile communications services were mobile business and tradespeople, and wealthy individuals. Only as prices fell have residential consumers, non-mobile office workers, and teenagers become users, and their needs are very different from those earlier into the market. The changing profile of adoptees creates particular challenges for marketers (Moore, 1991). This has led to the notion of a "chasm" between one adopter segment and the next as in Figure 5.2.

How quickly do new products and technologies reach saturation? If one considers an innovation such as written communication, which began several thousand years ago, diffusion has been very slow. Perhaps as many as half the world's population have still to learn to read and write. In contrast, cellular mobile telephones are now used by almost

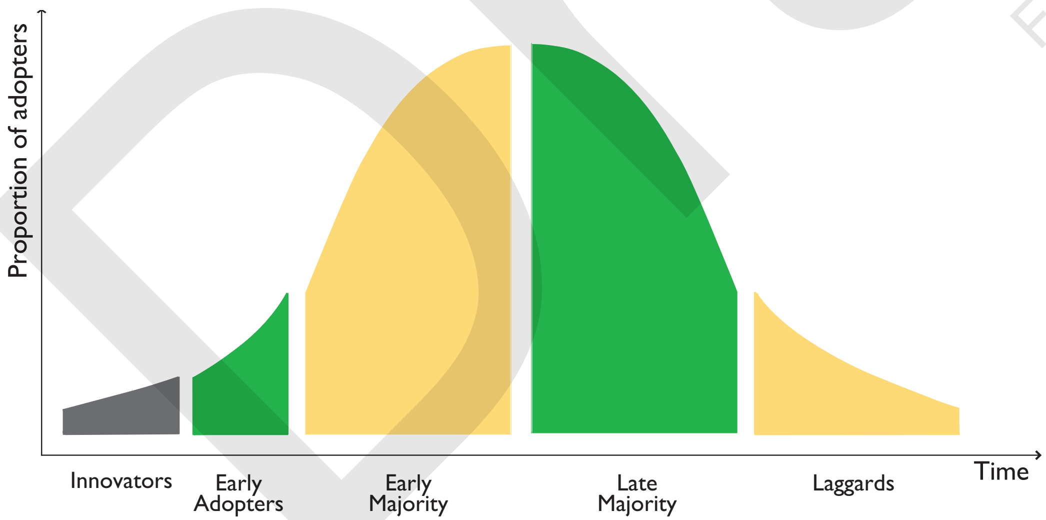


Figure 5.2: The revised technology adoption life cycle

1.7 billion people[1], a position reached in just over two decades from the launch of the first public cellular networks.

5.3 Standards and Adoption

The fact that many technology products and processes are network goods means that the presence or otherwise of technology standards may greatly impact adoption. If a standard exists in a particular domain, a potential adopter knows that choosing it will enable access to a network of other users. The greater the extent of adoption of the standard, the larger this network of users will be. Thus, one factor inhibiting adoption of Linux as an operating system (OS) for PCs was the fact that, until recently, most users had adopted the *de facto* standard of Microsoft Windows; while the user of a stand-alone machine could use any operating system they desire, installing an uncommon OS would mean not having access to the professional services, software tools and applications which support or run on the operating system. If adopting a technology is viewed as akin to choosing a move in a multi-party strategic game, where the potential adopter wishes to select the technology option that will be also chosen by the majority of their peers, then the existence of a standard may weight the payoffs in favour of a particular option and against others (Weitzel, 2004).

Where do standards come from? Standards may be imposed upon a user community by national Governments or international organisations, as with the adoption of GSM by all European and many other nations, for second-generation mobile communications networks; the communications regulatory agencies of the United States, in contrast, decided not to impose a particular technology standard in this domain. Or, standards may be strongly recommended to a user community by a voluntary standards organisation, as in the case of many Internet standards; two machines connected to the Internet may use any interconnection protocols they themselves agree on, for example, not necessarily the standard protocols, such as TCP and UDP, defined by the Internet Engineering Task Force.[2] Finally, standards may emerge from multiple independent choices of one particular technology over others made by many individual adopters; the common *QWERTY* typewriter layout is one such *bottom-up* standard (Gomes, 1998).

However, if standards are not imposed by some government or regulatory agency, then scope exists for multiple voluntary organisations to recommend competing standards or for competing standards to emerge from user decisions. To some extent, this may be occurring in the agent technologies domain, with several organisations having developed or aiming to develop standards related to the interoperation and interaction of intelligent software entities: the Foundation for Intelligent Physical Agents (FIPA, which has just been

[1] IDC: *Worldwide Mobile Phone 2005-2009 Forecast & Analysis Report* (Report 33290), May 2005.
[2] www.ietf.org

accepted by the IEEE as its eleventh standards committee),[3] the Object Management Group,[4] the Global Grid Forum,[5] and the World Wide Web Consortium.[6] The view has even been expressed that having multiple competing standards may be in the interests of major technology development companies, none of which wishes to see a standards body adopt a standard favourable to a competitor's products. In this view, large development companies may actually act so as to *divide and conquer* the various competing standards bodies, by, for example, participating intensely in one standards organisation at one time and another organisation at another time.

Faced with competing recommendations for standards, what will a potential adopter do? One result may be decision paralysis, with a user or company deciding to postpone adoption of a new technology until the standards position is clearer. Thus, in this case, multiple competing standards may inhibit uptake of a new technology and hence inhibit market growth. On the other hand, the proponents of competing standards have an interest in promoting their particular solution, and so the presence of multiple standards may lead to faster and more effective dissemination of information about the new technology than would be the case if there was only one standard. In this view, therefore, competing standards may actually encourage uptake of a new technology and hence of market growth. Which of these countervailing pressures actually dominates in any one situation will depend on the other factors influencing the decision processes of a potential adopter, for example, the extent to which the proposed technology satisfies an unmet need, the criticality of the need, and the extent of network effects.

Related to the issue of standards and network effects in adoption decisions by potential users of new technologies is the issue of *business ecologies*. Most companies and organisations are enmeshed in a network of business relationships, with customers, suppliers, competitors, and other stakeholders. If a downstream customer or an upstream supplier insists on adoption of a particular technology or standard as a condition of business, then a company may adopt it much sooner than they would otherwise. Thus, for example, the US company GE has insisted that most of its suppliers, including even law firms providing legal advice, bid for its business through online auctions. Of course, such pressure along a supply chain or across a business network may also greatly reduce the risks and costs associated with a new technology; thus, adoption decisions under such circumstances are not necessarily irrational. Recent research has considered the impact of networks of influence in business ecologies on software adoption decisions, e.g., von Westarp 2003.

[3] www.fipa.org
[4] www.omg.org
[5] www.ggf.org
[6] www.w3c.org

Agents versus Objects

In attempting to understand the likely future development of agent-based computing, and its pathway to adoption, one might usefully consider the history of object-oriented technologies. The origins of object orientation lie in early programming languages and AI technologies, starting with the Simula language in 1962 (Dahl 2002, Dahl & Nygaard 1965), predating the coining of the term “object-orientation” in 1970 by Alan Kay. Although several further developments ensued, including Smalltalk at Xerox PARC in 1973 and the introduction of frames by Marvin Minsky in 1975, it wasn’t until 1983 that C++ was formally established. The first textbook was released in 1985, the OOPSLA and OODBS conferences established in 1986, and the *Journal of Object Oriented Programming* only started in 1988.

These events were followed by more rapid developments of a practical nature, with the Object Management Group being formed in 1989, the development of Java in 1991 (though it wasn’t publicly released until 1995), and the establishment of standards that include CORBA (first specification in 1992, CORBA 2.0 in 1994), UML in 1994, and ANSI C++ in 1998. This is an extended period over which the technologies and techniques involved came to maturity and to wide scale adoption. Indeed, the time taken from the first object-oriented language until the ANSI C++ standard was established thus amounts to 32 years.

Agent and object technologies are both essentially disruptive technologies that provide (among other benefits) more effective and flexible techniques for software and its development. To understand how the future of agent-based computing may progress, we need to look to the differences between these two technologies.

First, object technology began in an era in which computing as a discipline and as an industry was relatively immature, and limited in scope. Although potential for applications certainly existed, the reality on the ground was not as pervasive and rooted in techniques, technologies, standards and paradigms as it is now, so that the change required and created then was far less substantial and challenging than is the case now.

Second, while there are still many problems to be tackled in computing, the degree of improvement, in terms of productivity or efficiency, to be realised from specific advances decreases as the general level of maturity in computing increases. Thus, while there was no step change arising through object orientation, the gradual improvement in the state of software is still likely to be far less marked with agent technologies.

Third, the current computing environment is much more heterogeneous, distributed and diverse than at any point previously, and it continues to change further in these directions. The consequence of this is a plethora of standards, techniques, methodologies and, importantly, multiple vested interests and corporate initiatives that must be integrated, overcome or otherwise addressed for broad acceptance of new paradigms. Investment in new technologies at this point of the IT adoption cycle presents a much more challenging problem than ever before. For all these reasons, it is likely that no technology in the near future will have anything like the impact of object orientation.

5.4 Agent Technologies

With this marketing background in mind, it is useful to consider the position of agent-based computer technologies. Adoption of agent technologies has not yet entered the mainstream of commercial organisations, unlike, say, object-oriented technologies. The majority of commercial organisations adopting agent technologies would, in our opinion, be classified as *early adopters*. We believe this because we know of only a small number of deployed commercial and industrial applications of agent technology, and because we believe considerable potential exists for other organisations to apply the technology.

What is the range of applications? To date, deployed applications of agent technologies have been concentrated in a small number of industrial sectors, and for particular, focused, applications. These have included: automated trading in online marketplaces, for example for financial products and commodities; simulation and training applications in defence domains; network management in utilities networks; user-interface and local interaction management in telecommunication networks; schedule planning and optimisation in logistics and supply-chain management; control system management in industrial plants, such as steel works; and, simulation modelling to guide decision-makers in public policy domains, such as transport and medicine.

Why are agent technologies still only in the early-adopter phase of diffusion? There are a number of reasons for this. Firstly, research in the area of agents technology is also still only in its infancy. Here, a reasonable comparison is with object-oriented programming (OOP) approaches, where the initial research commenced in 1962 (see box), some 32 years before the public release of the first version of Java and the widespread commercial adoption of OO technologies.[7] As a consequence of this, knowledge of agent technologies is still not widespread among commercial software developers, although of course projects such as AgentLink have tried to overcome this.

Secondly, as a result of the immaturity of research and development in agent technologies (discussed earlier), the field lacks proven methodologies, tools, and complementary products and services, the availability of which would act to reduce the costs and risks associated with adoption.

Thirdly, the applications for which agent technologies are most suited are those involving interactions between autonomous intelligent entities. While some applications of this sort may be implemented as closed systems inside a single company or organisation (for

The British news magazine, *The Economist*, has recently argued that the IT industry is currently in its third 15-year wave of progress, in which devices of every kind are connecting to the Internet. Unlike the first wave of the 1970s and 1980s, dominated by large proprietary mainframes, and the second wave of PCs hooked up to servers, with its *de facto* standards, this third wave is seeing *de jure* (industry agreed) standards taking over. [Make it Simple, *The Economist*, London, 28 October 2004].

[7]And 39 years before the two original researchers, Ole-Johan Dahl and Kristen Nygaard, received a Turing Award for their work.

example, agent-based simulation for delivery schedule decision-making) most potential applications of agent technologies require the participation of entities from more than one organisation. Automated purchase decisions along a supply-chain, for example, requires the participation of the companies active along that chain, so that implementing a successful agent-based application will require agreement and coordination from multiple companies. In other words, the application domains for which agent technologies are best suited typically exhibit strong network good effects, a factor that complicates technology adoption decisions by the companies or organisations involved.

It is for this reason that the agent community has expended so much effort on developing standards for agent communication and interaction, as undertaken by FIPA, so that agent systems may interoperate without the need for prior coordinated technology adoption decisions. However, as noted above, the agent technology standards landscape is currently one in which multiple organisations have developed or are developing standards for the interoperation and interaction of intelligent software entities. In these circumstances, adoption of agent technologies is not necessarily promoted by the presence of competing, and subtly different, standards.

5.5 Modelling Diffusion of Agent Technologies

AgentLink III developed a simple computer model to study the diffusion of agent technologies (McKean *et al.,* 2005). Our model uses assumptions about adoption decision processes and the relationships between different companies, and has not been calibrated against any real market data. It is intended only to provide a means for exploration of relationships between relevant variables and indicative information about these relationships. We fully recognise that the results of a generic model such as this will be highly dependent on the structure and assumptions used to create the model. Moreover, the features of specific markets, such as those for agent technologies, may result in very different outcomes from those described here. Thus the results described here should not be considered as guidance for specific marketing strategies or industrial policies in the domain of agent-based computing.

5.5.1 Model Design

Organisations potentially adopting agent technologies were represented as individual nodes in a graph. Directed connections (edges) between nodes were used to represent the influence of one organisation over another in a decision to adopt or not adopt agent technologies. Thus, for example, a large company may be able to influence technology decisions of its suppliers. Because different industries have different degrees of concentration and different networks of influence, our model incorporated several different graphical structures – network topologies – which we believe to be representative of the diversity of real-world industrial and commercial networks, as follows.

- A: 50 nodes not connected (i.e., no influence from one node to another). This topology models an industry that is highly disaggregated, with independent technology decision-making.
- B: 50 nodes with a dense set of connections, and influence in one or the other direction. This topology models an industry that disaggregated, but where peer relationships are important in technology decisions.
- C: 5 major nodes (parents), each connected to and influencing 9 subsidiary nodes (children), in a cluster formation. This topology models an industry where supply chains are not deep.
- D: 5 parent nodes, each connected to and influencing 9 subsidiary nodes linked together as in a linear supply chain. This topology models an industry where supply chains are deep, and downstream companies have distinct supply chains.
- E: 5 parent nodes, each connected to and influencing 9 subsidiary nodes linked together as in a linear supply chain, with at least one child node also influenced by a second parent. This topology models an industry where supply chains are deep, and downstream companies have overlapping supply chains.

Nodes were then modelled as independent and autonomous decision-makers, each making decisions to progress (or not) through a technology adoption lifecycle. The five stages in this lifecycle were:

- Agent technology not adopted
- Agent technology under consideration
- Agent technology being trialed
- Agent technology partially adopted
- Agent technology fully adopted.

Time in the model is assumed to be discrete and linear, with nodes making decisions between timepoints, based on the status of variables at the most recent timepoint. Each timepoint may be considered as a generation in the adoption lifecycle.

At each stage in the lifecycle, a node may decide to proceed to the next stage, remain at the current stage, or to return to the previous stage. The mechanism used by each node at each stage to make these decisions depends on a number of relevant factors, which were drawn from a study of the marketing literature (Lilien *et al.*, 1992, Mahajan *et al.,* 1993, Urban and Hauser 1993) and the economics literature (Weitzel 2004, von Westarp 2003):

- The current need of the organisation for the technology. This was assigned randomly to nodes.
- The costs of adoption. These costs fall as the number of nodes progressing through the adoption lifecycle increases. Nodes are assumed randomly to be able to afford the technology at the current level costs.
- The availability of complementary software tools. These are increasingly available as more nodes move through the technology adoption cycle, and thus encourage adoption of the technology.
- The presence of a technology standard. The existence of a single standard is assumed to encourage technology adoption by nodes, while the presence of more than one standard encourages adoption in some nodes and discourages it in others.
- The success of a technology trial. Not all trials are successful. However, an unsuccessful trial does not necessarily lead to non-adoption of the technology, since an organisation may have pressing needs for the technology.
- The extent of influence of other connected nodes over each node. Thus, downstream customers may strongly influence upstream suppliers in their choice of technologies. It is through this factor that the network topology impacts upon the decisions of individual nodes, and therefore how the model demonstrates the effect of the technology being a network good.

For each node and for each decision, these factors were then combined through a factor-weighting mechanism; the outcome of this combination is a decision: to progress forward to the next state; to remain in the current state; or to revert to the earlier state, in the technology adoption lifecycle. The weighting mechanism differs across the states of the technology adoption lifecycle to better represent the real-world decision processes. The weights and weighting mechanism used in the model were developed on what are believed to be reasonable assumptions regarding real-world decision processes, informed by the marketing literature. It is important to recognise that the factor-weights and the decision mechanism has not been calibrated directly against any real-world agent technology adoption decisions in companies or organisations. The AgentLink III model allows the weights to be set by the user, and so it may be possible to calibrate the model in this way in future work. Further details on the design and implementation of the model can be found in McKean *et al.* 2005. The model itself can be downloaded from AgentLink's web-pages.[8]

[8] www.agentlink.org

5.5.2 Simulation Results

One thousand simulation runs with random starting values were undertaken for each network topology and assuming different numbers of technology standards (zero, one and two). In each simulation run, the diffusion model ran until all nodes had adopted the technology, and the number of generations required to reach this end-state was then recorded. These measurements were than averaged across the 1000 simulation runs, and the results are shown in Table 5.1.

As might be expected, the network topology can have a major effect on the numbers of generations needed to reach full adoption. Likewise, for any given topology, the presence of a single standard may reduce the time steps needed for full adoption by more than half. Interestingly, having two competing standards inhibits full adoption, but not as greatly as having no standard at all. Thus, the model provides indicative support for the positive impact of standards on technology adoption decisions. It is also noteworthy that this impact is seen regardless of the network topology, in other words, regardless of the industry structure, at least for those topologies included in the model simulations.

Network Topology	No Standards	Single Standard	Two Standards
A: Disaggregated industry (non-connected nodes)	66.9	26.5	48.4
B: Disaggregated industry with peer relationships	66.7	26.8	48.7
C: Industry with shallow supply chains	25.0	17.6	22.1
D: Industry with deep, independent supply chains	76.5	26.6	49.1
E: Industry with deep, overlapping supply chains	67.6	19.8	48.7

Table 5.1: Numbers of generations to 100% adoption (by topology and numbers of standards).

5.6 Activity in Europe

The European position on research and development in agent systems is healthy. There have been numerous active research groups in universities and research laboratories across Europe since the early days of the emergence of the field of agent-based computing as a distinct discipline, and the quality of work done is competitive at a global level. One reason for this is that since 1998, the European Commission has provided funding (albeit limited) to support the community through coordination projects, providing a focus and coherence to the community that might not otherwise have been possible. The value of these AgentLink projects has not just been in academia; AgentLink counts around 40% of its organisational members from industry. Interestingly, research activity was generally sustained despite the bursting of the Internet bubble, and it can be argued that the efforts of the Commission in supporting the agent community helped to minimise the consequences of this crash.

Yet, there have been consequences. According to one analysis (The Netherlands Ministry of Economic Affairs, 2004), in the period before the bursting of the bubble, the ICT sector was characterised by hypercompetition, in which industries tried to outpace their competitors with speed of innovation. Business innovations were implemented in a quick and dirty fashion so as to minimise time to market and achieve rapid, exponential growth, at the cost of poorly conceived business models, and a high cash burn rate. The collapse led to consolidation in ICT sectors, and the emphasis has since shifted to the *e-enablement* of core business processes, like fully integrated supply chains and supply networks, with a focus on visible and measurable impact. This shift can now also be seen in the positioning of agent technology providers, who now focus more on these latter areas, and less on fundamental process change.

In the USA, ICT is stimulated by the cultivation of a high-tech entrepreneurial culture, providing ready customers for new technologies and close cooperation between industry and universities. In addition, public R&D is oriented towards areas considered important for future applications and identified as national priorities. Among the USA's 16 "Grand Challenges" are the following relevant to agent technologies: knowledge environments for science and engineering; collaborative intelligence: integrating humans with intelligent technologies; and managing knowledge intensive organisations in dynamic environments.

By contrast, European innovation culture and policy are more sluggish, despite the efforts of the European Commission. The grand challenges may be reflected in the strategic objectives of FP6, and in other relevant policy documents, but the ready customers for new technologies and the close cooperation between business and universities are not always apparent. In addition, there is also a recognition at the level of the European presidency, in the report published by The Netherlands Ministry of Economic Affairs (2004), of the need to "accelerate the introduction of disruptive technologies," the most relevant of the 10 breakthroughs identified as being needed to move towards the Lisbon goals (European Commission, 2000). Broad deployment and use of disruptive technologies require understanding and acceptance. Yet the lack of adequate and sophisticated interactions between industry, government and society stakeholders often obstructs the process of achieving understanding and acceptance.

However, due to the support of AgentLink by the Commission, at least some form of drawing together of the research and business communities has taken place in the domain of agent-based computing, and there are ready channels for interaction to facilitate different models of cooperation.

Figure 5.3 illustrates the impact of activity in Europe, with AgentLink and Agentcities. NET providing coordination of the community through a period of intense change and innovation at the research level. Usable FIPA standards, for example, were developed

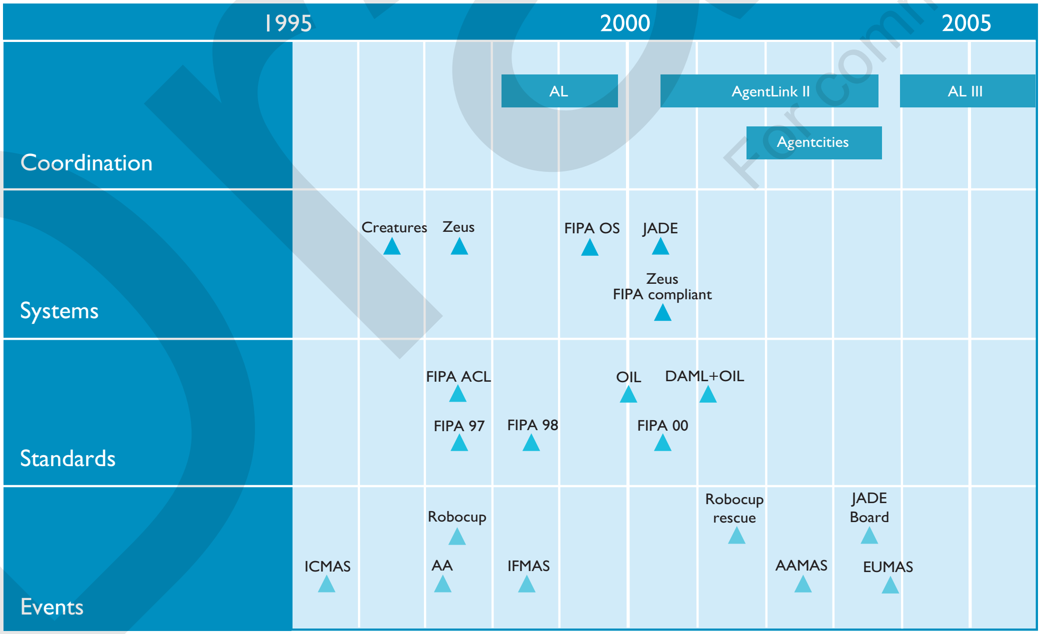


Figure 5.3: European activity in agent-based computing in recent years.

in 1998, but matured in 2000; several FIPA compliant agent platforms (JADE, Zeus and FIPA-OS) were also released by 2000, while other systems have also made an impact, such as the Creatures computer game, for example. Meanwhile developments in the Semantic Web gave rise to OIL and then DAML+OIL. At the bottom of the figure, key events in the development of the research community are indicated: the International Conference on Multi-Agent Systems (ICMAS) first appeared in 1995, the Autonomous Agents Conference (AA) in 1997, and both were combined into the International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS) in 2002. In addition, the International Foundation for Multi-Agent Systems (IFMAS) was established in 1998, and a similar European initiative was lanuched in 2003 with a European workshop, the European Workshop on Multi-Agent Systems (EUMAS).

6.1 Deliberative Delphi Survey

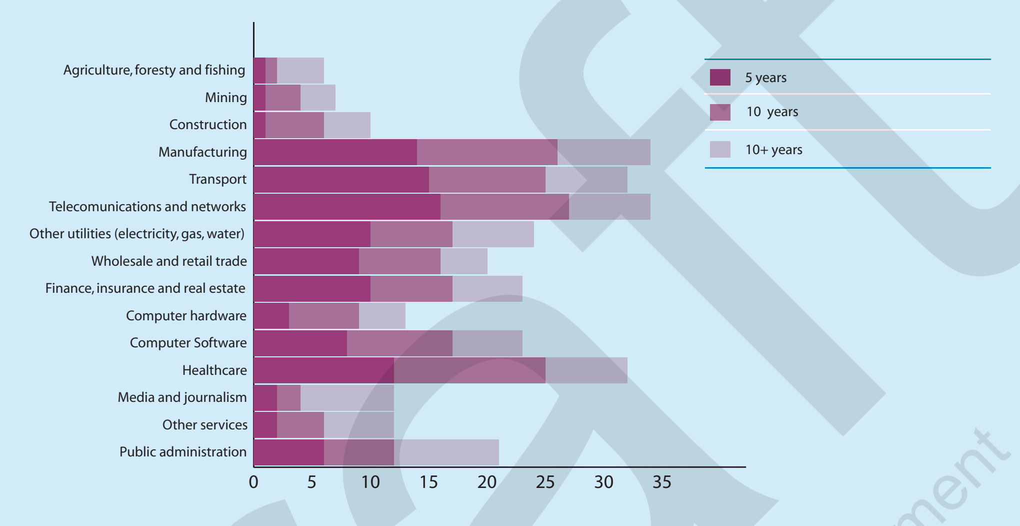
In an effort to elicit an informed assessment of the current state of development of agent technologies and the likely future market penetration for different areas, AgentLink III undertook a Delphi survey of opinion from a selected group of experts in the field. The Delphi method makes use of a limited panel of experts, selected on the basis of their expertise, and calling on their insights and experience. The hypothesis underlying Delphi is that these experts are better equipped to predict the future than theoretical approaches, extrapolation of trends, or more generally survey methods. In standard Delphi studies, participants are asked to give their predictions, which are aggregated and shown again to the participants in subsequent rounds. After seeing their peer-group average, the participants are allowed to revise their predictions, with the intention that the group will converge toward the "best" response through this consensus process. In AgentLink's *Deliberative Delphi* study, we modified this process by asking participants to give their reasons for their predictions and opinions, and circulated these reasons, as well as the aggregated results, in order to provide a more justified and useful exercise. The experts deliberated on their projections, hence the *deliberative* study.

The study involved 23 participants, of whom 5 were senior academic experts, and the remaining 18 coming from industry. Of this latter group, 11 were from major, typically multi-national companies, and 7 from smaller, newer companies specialising in agent technology. The industrial group included one major *traditional* manufacturer, two telecommunications companies, and several IT services companies. Participants were mostly European, but included representatives from the US, Japan and Australia. Full results are available in (Munroe et al., 2005).

6.1.1 Industry Sector Penetration

It is still too early to consider the penetration of different industry sectors, but in a relative analysis of those domains that are likely to encourage the take-up and deployment of agent technologies, the Deliberative Delphi study identified telecommunications and networks, manufacturing, transport and healthcare as the most significant over the next 5 years, 10 years and beyond. Participants were asked to select those in which they considered there would be likely deployment, with the results showing three broad classes. The second tier of domains includes: wholesale and retail trade; finance, insurance and real estate; computer software; public administration; and other utilities. The results are summarised in Figure 6.1, with all industry sectors represented, showing the number of times each was selected by participants over the different time periods. It is interesting to note

Figure 6.1: Manufacturing, transport, telecommunications and healthcare are likely to encourage deployment of agent technologies

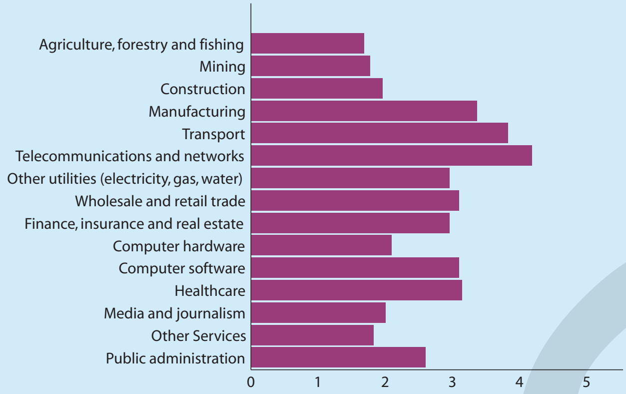


that computer software comes relatively low down the list, in this second tier. This contrasts with much academic research that has been focussed on eCommerce and eBusiness systems in recent years, partly because of its relative currency in the light of the Internet boom, and partly because of its ready availability as a domain to study. One question to consider is whether this focus is out of sync with business application realities, or whether the survey points beyond immediate application domains.

Later, when asked to evaluate in which sectors agents were expected to make the greatest impact, by rating each on a 1 to 5 scale (with 1 indicating no impact at all, and 5 indicating a very large impact), responses were broadly similar, suggesting a considered analysis. The means of these responses are shown in Figure 6.2.

More specifically in relation to computing, however, our experts were extremely confident that today's major software vendors will have developed products with integrated agent technologies for supply chain management by 2010. One reason for this is that we are already seeing the emergence of products in this space, even if just at the start of that development. For some, supply chain management is part of the eBusiness domain, which will see agent-based systems emerging as the most prevalent technology, as a differentiator based on intelligence and autonomy, to address intense competition. Other domains are less clear, with the little confidence in the view of agent technology deployment across all products.

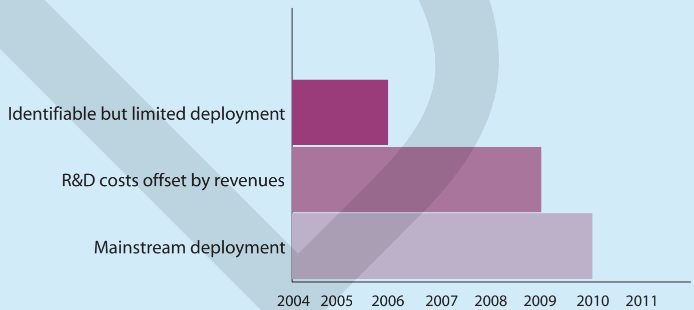
Figure 6.2: Agents will make the greatest impact in telecoms, transport and manufacturing



6.1.2 *Deployment of Agent Technologies*

Turning this around, the expert panel considered identifiable but limited deployment of agent technologies in more general applications (such as negotiation as part of e-commerce applications) to be achievable on average by 2006, research and development costs in agent technologies to be offset by revenues generated by 2009. Although some companies are already in the enviable position of generating revenue that exceeds costs, the mainstream deployment of agent technologies, on average, is not expected to be realised until 2010. The mean response for these issues is shown in Figure 6.3. However,

Figure 6.3: Mainstream deployment of agent technologies will not appear until 2010



given the responses to the earlier questions, this seems optimistic, and is coherent only for limited domains or applications.

Reasons for the expressed opinions varied, but some suggested that the strategic decisions required by companies in order to adopt new technologies have not yet taken place, leading to a delay in the possibilities for deployment. Nevertheless, there have been deployments in several large commercial organisations, electronic assistants in the form of software agents for wireless, pervasive or so called context-aware computing, and applications in which specific agent technologies are used (in manufacturing control, diagnosis, space, and so on). Though these are limited, this number will increase over the next few years, but they may not be labelled as agent-based systems. Indeed, if there is a lack of mainstream success in the short term, at least one expert suggests that agent technology may need to rebadge itself, especially in light of current grid computing standards such as web service agreements.

However, one respondent shows some insight by stating that it will be hard to calculate returns, since successful products will not look as though they have any agents. A general problem with software, especially in research and development, is the tendency to focus on the technologies applied rather on the effective solution to a problem. Yet a focus on the solution, regardless of the technologies used, may obscure the explicit value of agent technologies through their successful use and integration.

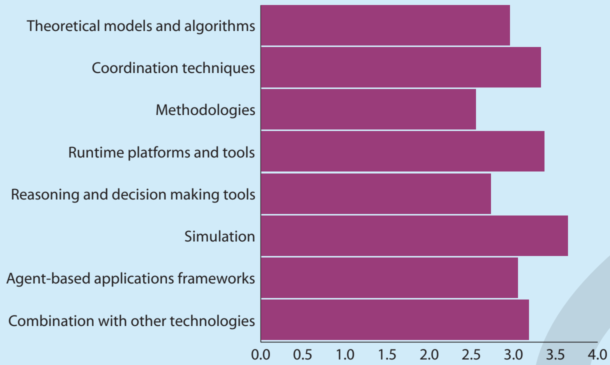
Other difficulties relate to issues such advanced reasoning capabilities being needed only for complex problem types, but until infrastructure is more standardised, the focus can only be on deployment of simple composition of services. Similarly, trust and legal issues appear to be a hindrance to commercial adoption.

6.1.3 Technology Areas and Maturity

In relation to specific technological areas, the experts were asked to assess the current state, and to what extent agent technologies were ready for deployment now. Again, they rated different technology areas on a 1 to 5 scale (with 1 indicating that the area was not ready for deployment, and 5 indicating that the technology was ready now). The means of these responses are shown in Figure 6.4. Those areas that exceeded the average for deployment now include coordination techniques, runtime platforms and tools, simulation, and integration or combination with other technologies. Those below average include theoretical models, algorithms and paradigms, methodologies for development, reasoning and decision-making tools, and agent-based application frameworks.

Participants were also asked which technology areas were seen as strong for the application of agent tools, models and solutions, and which were not. The areas exceeding

Figure 6.4: Simulation, runtime platforms and coordination techniques are ready for deployment now

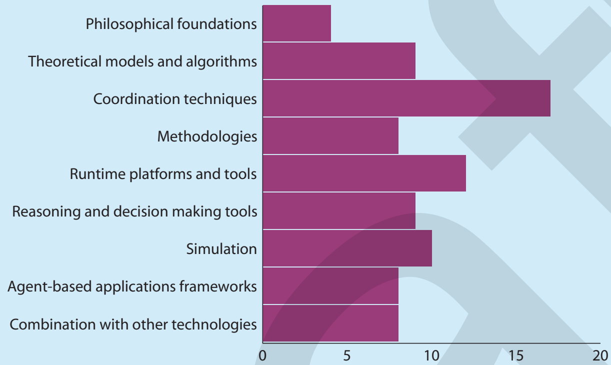


the average in terms of suitability for agent applications corresponded directly to those indicated above as being ready for deployment now, perhaps not surprisingly, while those suitable for application of non-agent solutions included the other areas of theoretical models, algorithms and paradigms, methodologies for development, reasoning and decision-making tools, and agent-based application frameworks. Interestingly, runtime platforms and tools were deemed appropriate for both agent and non-agent solutions.

The results are shown on the graphs in Figures 6.5 and 6.6, which indicate the number of times each area was selected by respondents as suitable for agent and non-agent solutions, respectively. We can see that coordination techniques are seen as being especially strong for agent technologies, which are also relatively ready for deployment. Runtime platforms are also above average in comparison to other areas in all measures, but attract the highest score for the suitability for non-agent tools. Reasoning and decision-making tools score close to the average on all issues, and simulation is similar, except that it is seen as being the most ready for deployment now. By contrast, agent-based application frameworks are below average in comparison to other areas except in readiness for deployment of agent technologies, in which it reaches the average.

At the same time, the participants were asked which problem areas were suitable for application of current agent technologies now, in 5 years, in 10 years, and beyond, by rating the problem areas on a 1 to 5 scale (with 1 indicating that the area was not suitable and indicating that it was very suitable). The results, in Figure 6.7, showed that interfaces, negotiation, coordination, complex systems modelling, and simulation scored highest, with

Figure 6.5: Coordination techniques offer the strongest agent solutions



all problem areas showing suitability in the higher range after 10 years.

6.1.4 Standards

Since the current technological context provides an appropriate base on which to build agent systems, and also suggests the use of agent technologies as never before, we also

Figure 6.6: Non agent solutions are better for methodologies and runtime platforms

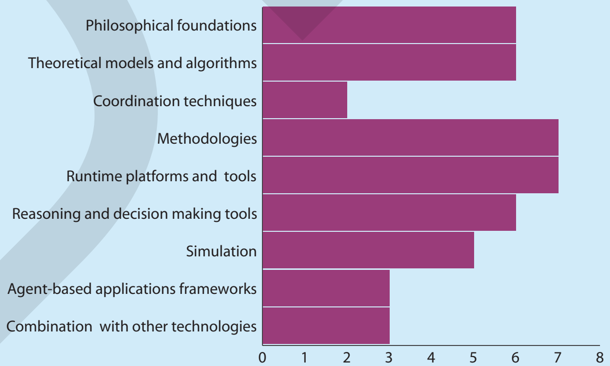
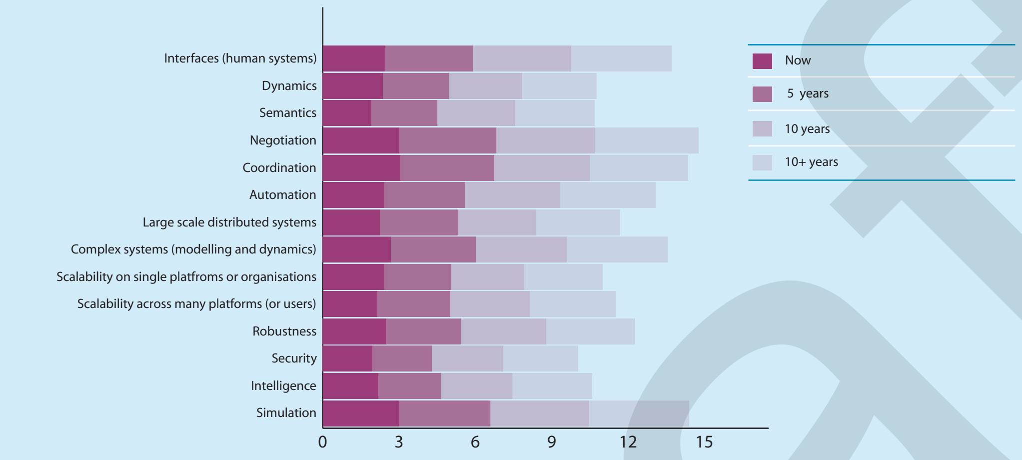


Figure 6.7: Negotiation, coordination, simulation, interfaces and complex systems are suitable for the application of current agent technologies



asked how important different technologies and standards were to the take-up of agents now, in 5 years, in 10 years, and beyond. The results for each are shown in Figures 6.8 and 6.9, which suggest the overriding significance of web services and other web technologies for take-up from now onwards. As time progresses, the impact of the semantic web, Grid technologies, P2P, AI planning systems and other eBusiness technologies are likely to have

Figure 6.8: Web services and technologies are important now, but semantic web, Grid and P2P technologies are catching up

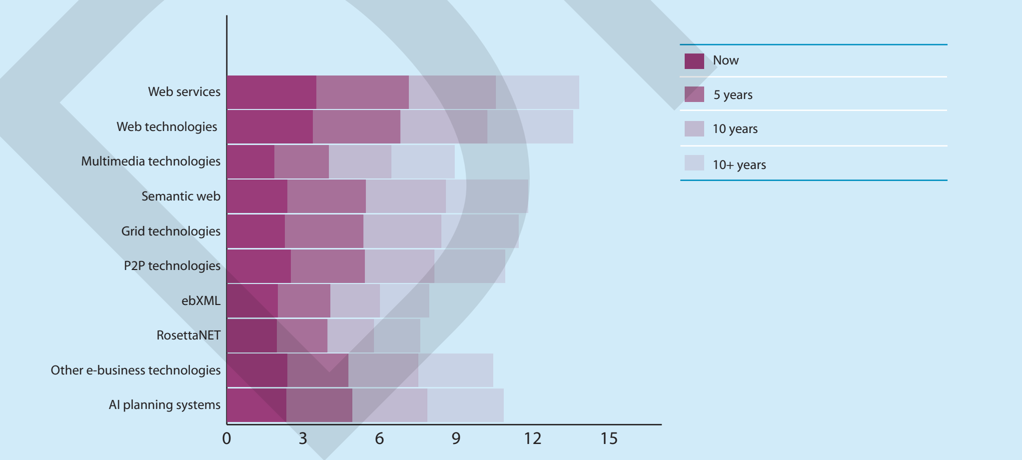
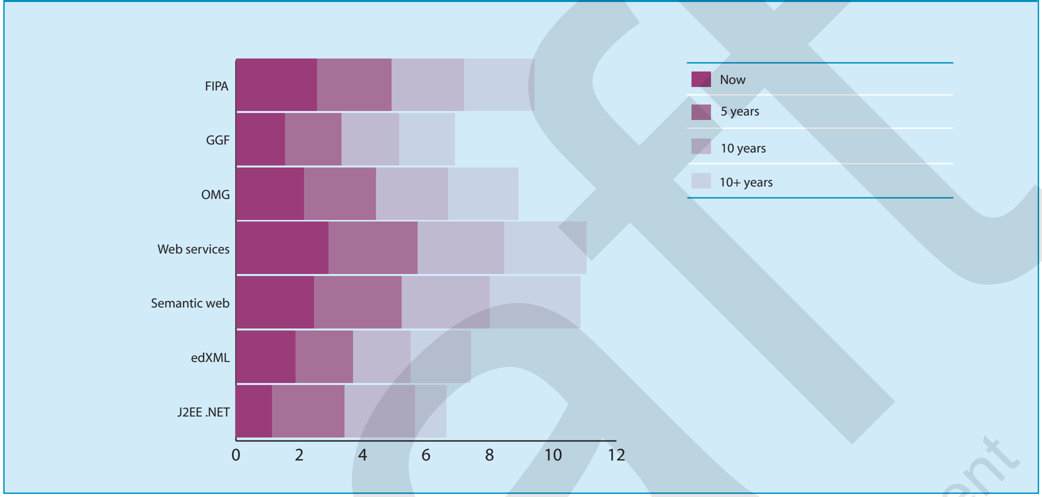


Figure 6.9: Web services and semantic web standards are most important, with FIPA and OMG standards close behind



an increasing impact. In terms of standards, web services and the semantic web are most important, but the efforts of FIPA and the OMG are also regarded as facilitating take-up and deployment.

6.1.5 Prospects

In relation to the issue of whether or when agent technology is likely to replace object-oriented technology, the majority (59%) of respondents do not believe that this will ever happen, with most of these arguing that agent and OO technologies are complementary, and not competitive. The view is consistent with that taken in this document, yet it is

Figure 6.10: Most believe that agent technology will not replace object technology but complement it instead

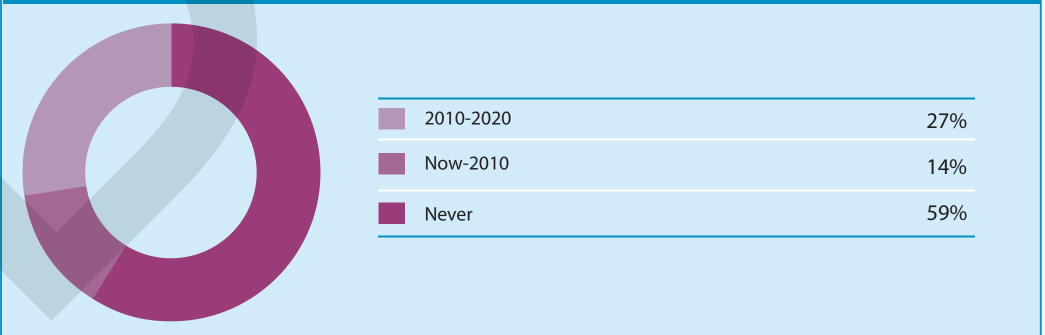
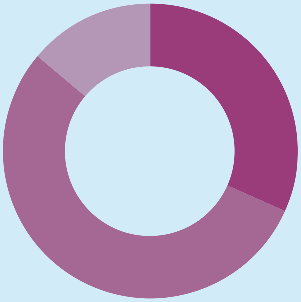


Figure 6.11: The vision and commitment of the academic and research communities should be longer term



■	Short-term	14%
■	Medium-term	54%
■	Long-term	32%

interesting to note that the remaining 41% believe that there will come a point in time at which agents will replace object technologies, though it is recognised that the technologies may converge rather than one supplanting the other.

More generally, the participants were also asked what kind of timeline the vision and commitment of the academic and research communities should take, choosing from short term (1-3 years) medium term (4-6 years) and long term. Perhaps not unreasonably, the results, shown in Figure 6.11, suggest that the short term is still too close, only 14% choosing such an immediate outlook, with the majority of 54% identifying the medium term as the right timescale. The remaining 32% took the longer term view of 7-10 years or more.

6.2 The Agent Technology Hype Cycle

Technology forecasting is a notoriously difficult task. In seeking to understand patterns of technology development in the mid-1990s, Gartner devised a model known as the Hype Cycle (described below), which indicates the maturity of a technology, from initial excitement to disillusionment and then, for some eventual market acceptance.

The Hype Cycle involves the following five stages.

- Technology trigger: introduction of the technology to a wider audience.
- Peak of inflated expectations: the high point, at which the claims of the benefits of the technology are often exaggerated.
- Trough of disillusionment: as the promises fail to be delivered, many observers begin to ignore the technology.
- Slope of enlightenment: more is learned about the technology and, as many of the problems from the trough are resolved, standardisation takes place, and the technology is adopted primarily in the areas that perceive the greatest benefit.

- Plateau of productivity: the new technology is well understood and stable, and becomes mainstream. Benefits and drawbacks for adoption are also widely known.

6.2.1 The Gartner Analysis

Gartner's July 2004 analysis of technologies and applications[9] places various agent technologies, agent-related technologies, application domains and drivers at various different points in the cycle.

In terms of infrastructure, business process execution languages (BPEL) are rising on the technology trigger path, with between 1% and 5% market penetration. Basic web services for service definition and application integration, using SOAP and WSDL, are climbing the slope of enlightenment and are implemented by major software vendors, reaching 20% to 50% market penetration. Advanced web services for higher quality of service, which will enable advanced business-critical functions over standards-based networks, using SOAP, WSDL, UDDI, WS-Security and WS-R, depend on the availability of standards, and implementations are not yet fully delivered by vendors.

Drivers and domains figure primarily through the semantic web, both of which are placed at the peak of expectation; while the expectation is for a transformational impact, at present it has less than 1% market penetration. Similarly, the Trading Grid, an interconnection of networks and marketplaces to support virtual organisations, is also transformational but just at the very start of the cycle. With lower perceived impact, but more mature, are eMarketplaces, now with up to 5% market penetration. Each of these is predicted to take up to 10 years to plateau.

Intelligent agents as a whole are seen as being in the trough, having been overhyped in the past, as synthetic characters and chatterbots were in the past. By contrast, web self-service agents, which act on a customer's or business's behalf to automate transactions are finally "catching on", and have reached up to 5% penetration. In all these cases, however, these are lightweight agents, with the mainstream of agent technologies still to engage. For example, agent-based integration is concerned with enabling distributed applications that demand autonomy and flexibility. In this area, commercial technology is still new, and

[9] Hype Cycle for Application Integration and Platform Middleware, Gartner, 2004.
Hype Cycle for Application Development, Gartner, 2004.
Hype Cycle for Human-Computer Interaction, Gartner, 2004.
Hype Cycle for B2B CRM Technologies, Gartner, 2004.
Hype Cycle for the Knowledge Workplace, Gartner, 2004.
Hype Cycle for Supply Chain Management, Gartner, 2004.

the sector is dominated by small startups and only a small number of users, so agent-based integration is at the start of the cycle. Gartner estimates that market penetration is less than 1% of the target. Given the position of the semantic web, this is perhaps not surprising, but the time to plateau is shorter, at up to 5 years.

At the embryonic stage are: swarm intelligence, or emergent computing, which fits directly with the complex systems discussed above; and affective computing, which seeks to recognise human emotional states for better user interfaces. At present, these are mainly in the domain of research laboratories.

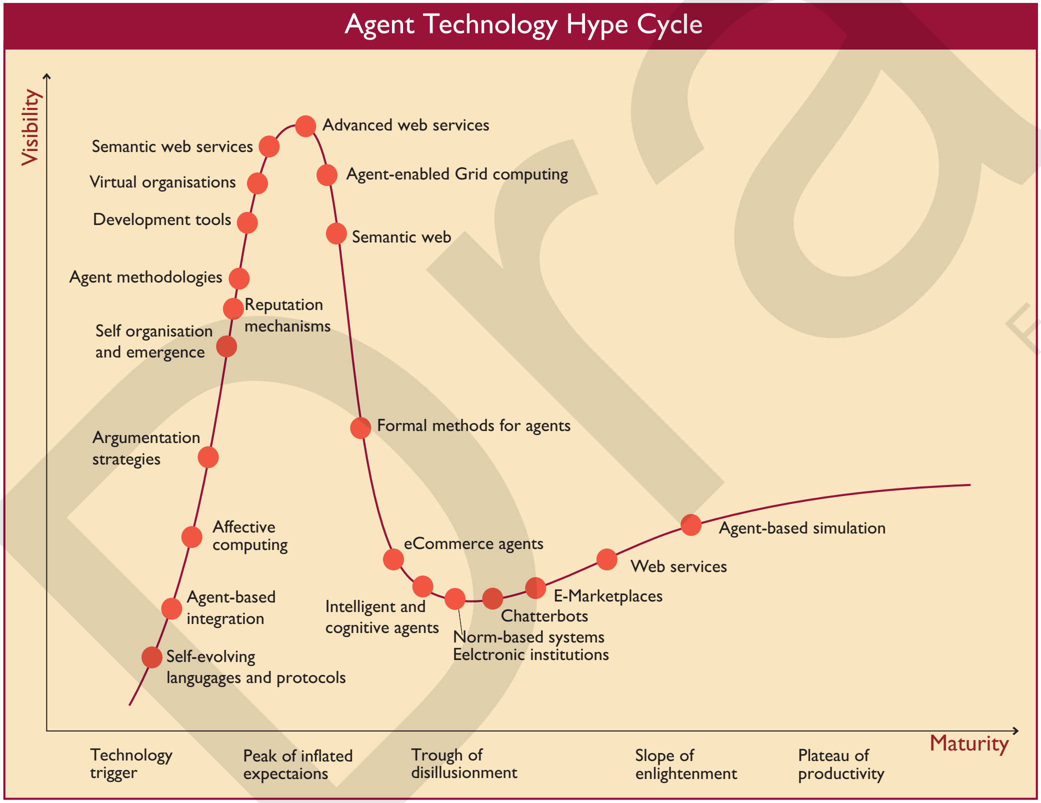


Figure 6.12: The agent technology hype cycle

6.2.2 *The AgentLink Analysis*

Based on Gartner's analysis, and a review from the AgentLink community, taking into account the analyses reported earlier in this document, we have developed a complementary Hype Cycle for agent technologies, illustrated in Figure 6.12. Here, some technologies are seeing real deployed value across a range of applications. Increasingly, for example, agent-based simulation is being applied to logistics and other application domains, achieving clear and distinct results, with suppliers creating a space for themselves in this market niche. Similarly, web services are increasingly being used for the development of systems where there is a genuine understanding of the business benefits, rather than inflated and false expectations.

However, many technologies are still to mature. Intelligent and cognitive agents, with sophisticated architectures, such as BDI, are situated in the trough of disillusionment, as are norm-based systems and electronic institutions, not yet finding roles in most mainstream business applications. Similarly, eCommerce agents have much promise, but as yet have mostly been deployed in prototypes and demonstrators, though the infrastructure for enabling their operation (through electronic marketplaces) is now starting to mature.

More interesting, perhaps, are the early runners: self-evolving communication languages and protocols have promise, but it is far too early to consider them seriously. Climbing upwards to the peak of inflated expectations are formal methods for agent-based computing, self-organisation and emergence (as discussed in detail earlier in this report), methodologies, development tools and virtual organisations (which have gathered much interest from the business communities, but are not yet so developed technologically). The drivers of the semantic web and Grid computing are just past the peak, but it is still early to determine how quickly they will move into and out of the trough.

7 Technology Roadmap

In any high-technology domain, the systems deployed in commercial or industrial applications tend to embody research findings somewhat behind the leading edge of academic research. Multi-agent systems are no exception to this, with currently-deployed systems having features found in published academic research and prototypes of three to five years ago. By looking at current academic research interests and areas of focus, we are able to extrapolate future trends in deployed systems.

Accordingly, we have identified four broad phases of the future development of deployed multi-agent systems. These phases are, of necessity, only indicative, since some companies and organisations will be leading users of agent technologies, pushing applications ahead of these phases, while many others will be laggards. We aim to describe the majority of research challenges at each time period. Note that this view on timescales takes the research view rather than the development view in that typically research is about three to five years ahead of development in this context. This analysis is an updated version of the analysis intially undertaken in (Luck et al., 2003), but in this draft document, **we are seeking further input from the community on technologies and timescales; in particular, we are keen to receive suggestions for additions of technologies to add to this view, and comments on the timescales suggested here. To contribute, please contact the corresponding editor, Michael Luck, at the address given at the start or end of this document.**

7.1 Phase 1: Current

Multi-agent systems are currently typically designed by one design team for one corporate environment, with participating agents sharing common high-level goals in a single domain. These systems may be characterised as closed. (Of course, there is also work on individual competitive agents for automated negotiation, trading agents, and so forth, but typically also constrained by closed environments.) The communication languages and interaction protocols are in-house protocols, defined by the design team prior to any agent interactions. Systems are usually only scalable under controlled, or simulated, conditions. Design approaches tend to be ad hoc, inspired by the agent paradigm rather than using any specific methodologies. Although this is still largely true, there is now an increased focus on, for example, taking methodologies out of the laboratory and into development environments, with commercial work being done on establishing industrial-strength development techniques and notations.

It remains true that, for the foreseeable future, there will be a substantial commercial demand for closed multi-agent systems because of the security concerns that arise from open systems. While progress in this respect will change the nature of agent systems, the importance of closed, well-protected systems must not be underestimated.

7.2 Phase 2: Short-Term Future

In the next phase of development, systems will increasingly be designed to cross corporate boundaries, so that the participating agents have fewer goals in common, although their interactions will still concern a common domain, and the agents will be designed by the same team, and will share common domain knowledge. Increasingly, standard agent communications languages, such as FIPA ACL, will be used, but interaction protocols will remain non-standard. These systems will be able to handle large numbers of agents in pre-determined environments, such as those of Grid applications. Development methodologies will have reached a degree of maturity, and systems will be designed on top of standard infrastructures such as web services or grid services, for example. Example systems developed in this phase include those to enable automated scheduling coordination between different departments of the same company, closed user groups of manufacturing suppliers engaged in electronic procurement activities, or network-centric operations.

7.3 Phase 3: Medium-Term Future

In the third phase, multi-agent systems will permit participation by heterogeneous agents, designed by different designers or teams. Any agent will be able to participate in these systems, provided their (observable) behaviour conforms to publicly-stated requirements and standards. However, these open systems will typically be specific to particular application domains, such as B2B eCommerce or bioinformatics. The languages and protocols used in these systems will be agreed and standardised, perhaps drawn from public libraries of alternative protocols that will, nevertheless, likely differ by domain. In particular, it will be important to master this semantic heterogeneity.

Systems will scale to large numbers of participants, although typically only within the domains concerned, but with particular techniques (such as domain-bridging agents), to translate between separate domains. Thus, for example, a multi-agent system for automated meta-analysis of research results in some area of biology will be able to utilise bridge agents to undertake commercial negotiations when interaction with an eCommerce system is required, say for access to information protected by patent. System development will proceed by standard agent-specific methodologies, including templates and patterns for different types of agents and organisations. Semantic issues related to, for example, coordination between heterogeneous agents and access control, are of particular importance here.

Examples of systems in this phase will be corporate B2B electronic procurement systems permitting participation by any supplier, rather than closed user groups.

7.4 Phase 4: Long-term Future

The fourth phase in this projected future will see the development of open multi-agent systems spanning multiple application domains, and involving heterogeneous participants developed by diverse design teams. Agents seeking to participate in these systems will be able to learn the appropriate behaviour for participation in the course of doing so, rather than having to prove adherence before entry. Although standard communications languages and interaction protocols will have been available for some time, systems in this phase will enable these to emerge by evolutionary means from actual participant interactions, rather than being imposed. Of course, such languages, protocols and behaviours may be mere refinements of previously-developed standards, but will be tailored to their particular contexts of use.

By this phase, systems will be fully scalable in the sense that they will not be restricted to arbitrary limits (on agents, users, complexity, etc). As with the previous phase, systems development will proceed by use of rigorous agent-specific design methodologies.

7.5 Technologies and Timescales

Arising from this picture of the future of agent research, we see a number of broad technological areas of research and development over the next decade. These are summarised in Figure 7.1, with each area being broken down into distinct technological issues that are organised into the short term, medium-term and long-term future, according to the points in time at which they will attract successful attention from the research and development communities. In particular, the tables suggest that long-term issues are worthy of strategic investment and effort while short-term issues are largely already addressed or are being addressed. A much more detailed treatment of many of these issues can be found in (Luck et al., 2003; Luck et al., 2004).

	Short Term	Medium Term	Long Term
Industrial Strength software	Peer to peer Better development tools Agent UML Web Services	Generic Designs for Coordination Libraries for agent-oriented development	Best practice in agent systems design
Agreed Standards	FIPA ACL Peer to peer Better development tools Web Services Semantic description	Flexible business/trading languages Libraries of interaction protocols	Tools for evolutions of communications languages and protocols
Infrastructure for Open Communities	Web mining Semantic interaction Data integration and Semantic Web	Agent-enabled semantic web (services) Electronic institutions	Shared, improved ontologies Dynamic norms, roles, laws
Reasoning in Open Environments	Organisational views of agent systems	Enhanced understanding of agent society dynamics Theory and practice of argumentation strategies Norms and social structure Theory and practice of negotiation strategies	Automated eScience systems and other application domains
Learning Technologies	Adaptation Personalisation Hybrid technologies	Evolving Agents Self organisation Distributed learning	Run-time reconfiguration and re-design
Trust and Reputation	Security and verifiability for agents Reliability testing for agents Self-enforcing protocols	Norms and social structures Reputation mechanisms Formal methods for open agent systems Electronic contracts	Trust techniques for coping with malicious agents

Figure 7.1: Agent technology comprises areas that will be addressed over different timescales

8 Challenges

As IBM has stated in relation to Autonomic Computing, the vision of agent-based computing itself is enough to constitute a Grand Challenge, because of the need to bring together multiple technical and scientific disciplines as well as stakeholders across different sectors. The specific technical challenges continue to change as the field of agent-based computing advances and matures, and as related areas (like those discussed above) emerge and galvanise efforts that contribute to the general area. Inevitably, standards will continue to be critical, but it is not clear whether these should come from within the agent community or should emerge from more general computing infrastructure progress. (Recent relevant standards efforts are depicted in Figure 8.1.) Nevertheless, some key challenges have already been articulated in relevant areas. This section summarises some of the more salient challenges relevant to agent-based computing, drawing on the work of the AgentLink Technical Forum Groups, the UK Computing Research Committee (2004), Bullock and Cliff (2004), Foster et al. (2004), and Kephart and Chess (2003). These challenges are an initial set of broad challenges in different aspects of the field, and are not intended to be exhaustive. **One of the key purposes of this draft document is to focus the thoughts of the community on a more detailled and comprehensive set of challenges for agent-based computing, and we are seeking contribution from the community in this respect. Please email the corresponding editor, Michael Luck, at the address at the start and end of this document, with suggestions for inclusion in this challenge list.**

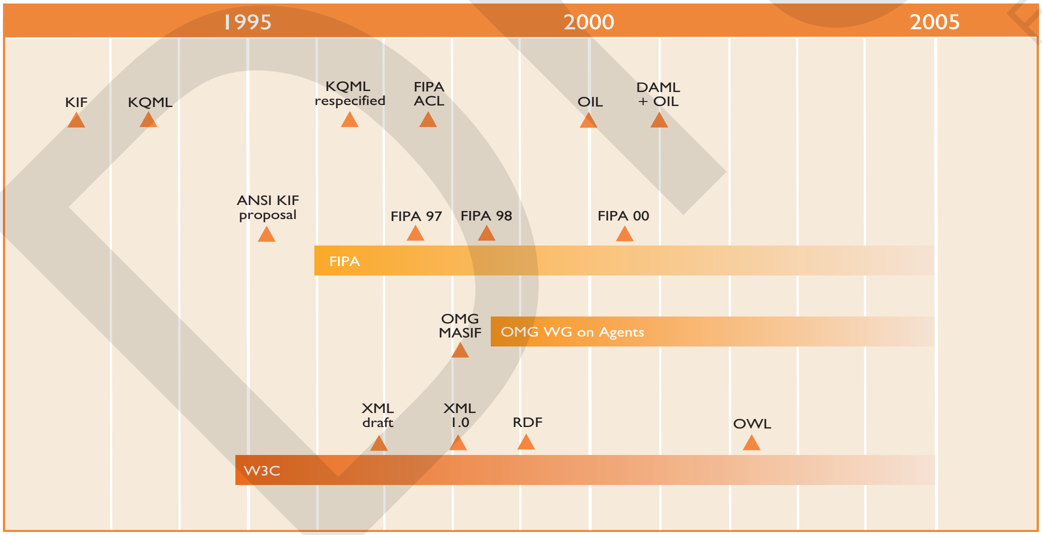


Figure 8.1: Standards activity in the area of agent-based computing

Trust negotiation and management

Sophisticated distributed systems are likely to involve action in the absence of strong existing trust relationships. While middleware addresses secure authentication, and there exist techniques for verification and validation, these do not consider the harder problems of establishing, monitoring, and managing trust in a dynamic, open system. As discussed above, we need new techniques for expressing and reasoning about trust, on both an individual and a social level (with reputation) to enable interaction in dynamic and open environments.

Virtual organisation formation and management

Virtual Organisations (VOs) have been identified as the means to release the power of the Grid, but well-defined procedures for determining when to form new VOs, how to manage them, and ultimately how and when to disband them, are still missing.

Resource allocation and coordination

The coordinated, autonomic management of distributed resources requires new abstractions, mechanisms and standards in the face of multiple, perhaps competing, objectives from different parties. Most R&D effort to date has focused on allocation and coordination mechanisms drawn from human societies (for example, common auction protocols), but the processing power and memory advantages of computational devices mean that completely new mechanisms and protocols may be appropriate for automated interactions.

Negotiation

To date, work on negotiation has provided point solutions. There is a need for a solid theoretical foundation for negotiation that covers algorithms and negotiation protocols, while determining which bidding or negotiation algorithms are most effective. From the system perspective, behaviour arising through the interplay of different negotiation algorithms must be analysed, and determining what kind of negotiation to consider, and when, must be established. Finally, effective negotiation strategies and protocols that establish the rules of negotiation, as well as languages for expressing service agreements, and mechanisms for negotiating, enforcing, and reasoning about agreements are also needed.

Emergence in large-scale agent systems

While still relatively young, research in the area of emergent properties of large-scale agent systems offers insights from natural physical processes in the real world to better understand the dynamics of the increasingly large-scale artificial systems now being built. This approach views large-scale multi-agent systems as examples of complex, adaptive systems, which are the domain of the new discipline of complexity science. As this science matures, its focus on macro-scale properties of interacting entities may impact on the design, implementation and control of large-scale multi-agent systems. Approaches from physics, biology and other related fields provide different methods to model large scale systems, but it is not clear to what extent are they equivalent, and what each approach brings.

Learning and optimisation theory

While learning and adaptation has a long tradition of research, particular contexts raise new issues. In sophisticated autonomic systems, agents continually adapt to the environment of other agents, and to each other, violating the assumptions of single-agent learning theories, and potentially leading to instabilities. Here, optimisation that assumes a stationary environment also fails pathologically, and new methods must be developed.

Methodologies

Many of today's challenges in software design stem from the distributed, multi-actor nature of new software systems. The development of methodologies for the design and management of multi-agent systems seeks to address this problem by extending current software engineering techniques to explicitly address the automated nature of their components. A wide range of methodologies have so far been developed, often addressing different elements of the modelling problem or taking different inspirations as their basis, yet there is no clear means of combining them to reap the benefits of different approaches. Similarly, agent-oriented methodologies still need to be successfully integrated with prevailing methodologies from mainstream software engineering, while at the same time taking on board new developments in other challenge areas.

Provenance

Today's distributed environments (including Grid, web services and agent-based systems) suffer from a lack of mechanisms to trace results and a lack of infrastructures to build up trusted networks. Provenance enables users to trace how a particular result has been arrived at by identifying the individual and aggregated services that produced a particular output. From both an academic and an industrial perspective, the research question is to design, formalise and implement an open provenance architecture. Such a provenance architecture should be scalable and secure; it must be open and promote interoperability.

Service architecture

There is a need for an integrated service architecture providing a robust foundation for autonomous behaviour, in order to support dynamic services, and important negotiation, monitoring, and management patterns. This will aid the convergence of agent and Grid technologies.

Service composition

While web service technologies define conventions for describing service interfaces and workflows, we need more powerful techniques for dynamically describing, discovering, composing, monitoring, managing, and adapting multiple services in support of virtual organisations, for example.

Semantic integration

In open systems, different entities will have distinct information models, demanding that techniques are developed for bridging the semantic gap. Advances are required in such areas as ontology definition, schema mediation, and semantic mediation.

9 Conclusions

A fundamental obstacle to the take-up of agent technology is the lack of mature software development methodologies for agent-based systems. Clearly, basic principles of software and knowledge engineering need to be applied to the development and deployment of multi-agent systems, as with any software. This applies equally to issues of scalability, security, transaction management, etc, for which there are already available solutions. The challenge with agent-based computing is to augment these existing solutions to suit the differing demands of the new paradigm, while taking as much as possible from proven methods. For example, agent software engineering methodologies need to draw on insights gained from the design of economic systems, social systems, and complex engineering control systems. In addition, existing middleware solutions need to be leveraged as much as possible, and this message has been understood: several companies have been working on platforms based on existing and standard middleware that is known and understood in the commercial domain.

In application terms, we are already seeing the deployment of de facto agent applications (in areas of pervasive computing, the semantic web, P2P networks, and so on). In the longer term, we expect to see the industrial development of infrastructures for building highly scalable applications made up of pre-existing agents that must be organised or orchestrated. Making the transition from research laboratory to deployed industrial applications is indeed a challenge, however, and it will be important to make scientifically sound business cases for implementations and descriptions that work both as stimulators for industry adoption and further research.

Many companies (including, Agentis, Magenta, Whitestein and others) have identified logistics as the first sector that will adopt agent technology, because it bridges the gap between the *functional* and the *technical*. At the same time, there is an increasing demand for distributed autonomous software, for example in transportation domains seeking to distribute operations, and supply chains without dominant parties representing the needs of individual companies and improving efficiency. One success here leads to another: companies within a supply chain using agent technologies consider possibilities for adoption internally. Factories, for example, want autonomous control systems that monitor and measure production line and factory-wide output.

For such commercial and industrial systems, agent technologies must emerge from the laboratory with a focus on business issues, quality and convergence with existing and emerging industrial technologies rather than innovation. Here, safety, reliability and traditional software quality measures are equally important, and must be addressed in order to encourage wider adoption. In particular, we need agent solutions for distributed enterprise-ready environments with exacting requirements. This might be achieved through transition mechanisms by which existing systems can be upgraded with a successively increased agent presence in a seamless fashion.

More generally, the adoption of agent technologies in business environments depends on how fast and how well agents can be linked to existing and proven software (methods). Agent technologies should be targetted at those aspect to which they are best suited, augmenting traditional techniques that should be used when agents are not applicable or appropriate. Ultimately, achieving this aim requires a commitment on the part of both business and research communities to collaborate effectively in support of more effective solutions for all. Such a dialogue is already underway.

References

K. J. Arrow, A. K. Sen, and K. Suzumura, editors. *Handbook of Social Choice and Welfare*. North-Holland, 2002.

T. Berners-Lee, J. Hendler and O. Lassila, The Semantic Web, *Scientific American*, 35-43, May 2001

D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris and D. Orchard, Web Services Architecture, W3C Working Group Note 11 February 2004, http://www.w3.org/TR/ws-arch/

S. J. Brams and A. D. Taylor. *Fair Division: From Cake-cutting to Dispute Resolution*. Cambridge University Press, 1996.

S. Bullock and D. Cliff, *Complexity and Emergent Behaviour in ICT Systems*, Foresight Report, DTI, UK, 2004

Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. Negotiating over small bundles of resources. In *Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems*. ACM Press, 2005.

P. Cramton, Y. Shoham, and R. Steinberg, editors. *Combinatorial Auctions*. MIT Press, 2005. To appear.

O.-J. Dahl, The roots of object orientation: the Simula language, in M. Broy and E. Denert (Editors): *Software Pioneers: Contributions to Software Engineering*, Heidelberg, Germany, 79—90, Springer, 2002.

O.-J. Dahl and K. Nygaard, *SIMULA: A language for programming and description of discrete event systems. Introduction and user's manual.* Oslo, Norway: Technical Report 11, Norwegian Computing Centre, 1965

U. Endriss and N. Maudet. Welfare engineering in multiagent systems. In *Engineering Societies in the Agents World IV*, LNAI 3071, Springer, 2004.

European Commission, *The Lisbon European Council: An agenda of economic and social renewal for Europe*. Contribution of the European commission to the Special European Council in Lisbon, 23-24th March, DOC/00/7, 2004.

I. Foster, N. R. Jennings and C. Kesselman, Brain meets brawn: Why Grid and agents need each other, in *Proceedings of the Third International Conference on Autonomous Agents and Multi-Agent Systems*, New York, USA, 8–15, 2004.

I. Foster and C. Kesselman (eds.), *The Grid 2: Blueprint for a New Computing Infrastructure*, Morgan Kaufmann, 2004.

Y. Fujishima, K. Leyton-Brown, and Y. Shoham. Taming the computational complexity of combinatorial auctions: Optimal and approximate approaches. In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*. Morgan Kaufmann Publishers, 1999.

L. Gomes, Ventures Column. *Wall Street Journal*, 25 February 1998. New York City, NY, USA, 1998.

A. Gould, S. Barker, E. Carver, D. Golby, M. Turner, BAEgrid: From e-Science to e-Engineering, in *Proceedings of the UK e-Science All Hands Meeting*, 2003.

Information Age Partnership Grid Taskforce, *Unlocking the Grid*, 2004. www.iapuk.org

IST Advisory Group, *Software technologies, embedded systems and distributed systems: A European strategy towards an Ambient Intelligence environment*, European Commission, 2002.

J. O. Kephart and D. M. Chess, The Vision of Autonomic Computing, *IEEE Computer*, 36(1), 41–50, 2003.

M. Lemaître, G. Verfaillie, H. Fargier, J. Lang, N. Bataille, and J.-M. Lachiver. Equitable allocation of earth observing satellites resources. In *Proceedings of the 5th ONERA-DLR Aerospace Symposium*, 2003.

T. Levitt, Exploit the Product Life Cycle. *Harvard Business Review*, **43** (6): 81—94, 1965.

G. L. Lilien, P. Kotler and K. S. Moorthy, *Marketing Models*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1992.

M. Luck, P. McBurney and C. Preist, *Agent Technology: Enabling Next Generation Computing (A Roadmap for Agent Based Computing)*, AgentLink, 2003.

M. Luck, P. McBurney and C. Preist, A Manifesto for Agent Technology: Towards Next Generation Computing, *Journal of Autonomous Agents and Multi-Agent Systems*, 9(3), 203–252, 2004.

V. Mahajan, E. Muller and F. M. Bass, New-product diffusion models. pp. 349—408, in: J. Eliashberg and G. L. Lilien (Editors): *Handbooks in Operations Research and Management Science, Volume 5: Marketing.* Amsterdam, The Netherlands: North-Holland, 1993

J. McKean, H. Shorter, P. McBurney and M. Luck, *The AgentLink III Technology Diffusion Model*. Technical Report ULCS-05-008, Department of Computer Science, University of Liverpool, UK, 2005.

D. F. Midgley, *Innovation and New Product Marketing*. London, UK, Croom Helm,1977.

D. S. Milojicic, V. Kalogeraki, R. Lukose, Rajan, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins and Z. Xu, *Peer-to-Peer Computing*, HP Technical report HPL-2002-57, 2002

G. A. Moore, *Crossing the Chasm: Marketing and Selling High-Tech Products to Mainstream Consumers*. New York City, NY, USA: HarperCollins, 1991.

S. Munroe, M. Luck and P. McBurney, *The AgentLink III Deliberative Delphi Survey*. Technical Report. Department of Electronics and Computer Science, University of Southampton, UK, 2005.

The Netherlands Ministry for Economic Affairs, Directorate-General for Telecommunications and Post, *Rethinking the European ICT Agenda: Ten ICT-breakthroughs for reaching Lisbon goals*, 2004.

R. Phaal, C. Farrukh and D. Probert, Technology roadmapping—a planning framework for evolution and revolution. *Technological Forecasting and Social Change*, 71, 5–26, 2004.

E. M. Rogers, *Diffusion of Innovations*. New York City, NY, USA: The Free Press,1962.

T. W. Sandholm. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence*, 135:1–54, 2002.

UK Computing Research Committee, Grand Challenges in Computing Research, The British Computer Society, 2004. http://www.ukcrc.org/grand_challenges/

G. L. Urban and J. R. Hauser, *Design and Marketing of New Products*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993

T. Weitzel, *Economics of Standards in Information Networks*. Information Age Economy Series. Heidelberg, Germany: Physica, 2004.

F. von Westarp, *Modeling Software Markets: Empirical Analysis, Network Simulations, and Marketing Implications.* Information Age Economy Series. Heidelberg, Germany: Physica, 2003.

F. Zambonelli and H. V. Parunak, Signs of a revolution in computer science and software engineering, *Engineering Societies for the Agents World*, Lecture Notes in Artificial Intelligence 2577, 13–28, Springer, Berlin, 2003.

Appendix: Methodology

In January 2004, a core roadmapping group was set up within AgentLink III, including Michael Luck, Peter McBurney and Onn Shehory, to oversee the development of this roadmap. Subsequently Steve Willmott joined the core team, which would lead a programme of review, discussion, consultation and debate across the first 18 months of AgentLink III.

The programme established was determined by three key timepoints at which documents would be produced: at 12 months with the initial Consultation Report that would be used for placing a marker in the community as a means of eliciting contributions and comment; at 18 months with the Roadmap Draft, which would essentially be the complete document that would be available for detailed analysis and discussion, both by targeted reviewers, and by the general community; and at 21 months, when the final document would be printed and widely distributed for maximum impact. These three key points delimit the three stages of roadmap development.

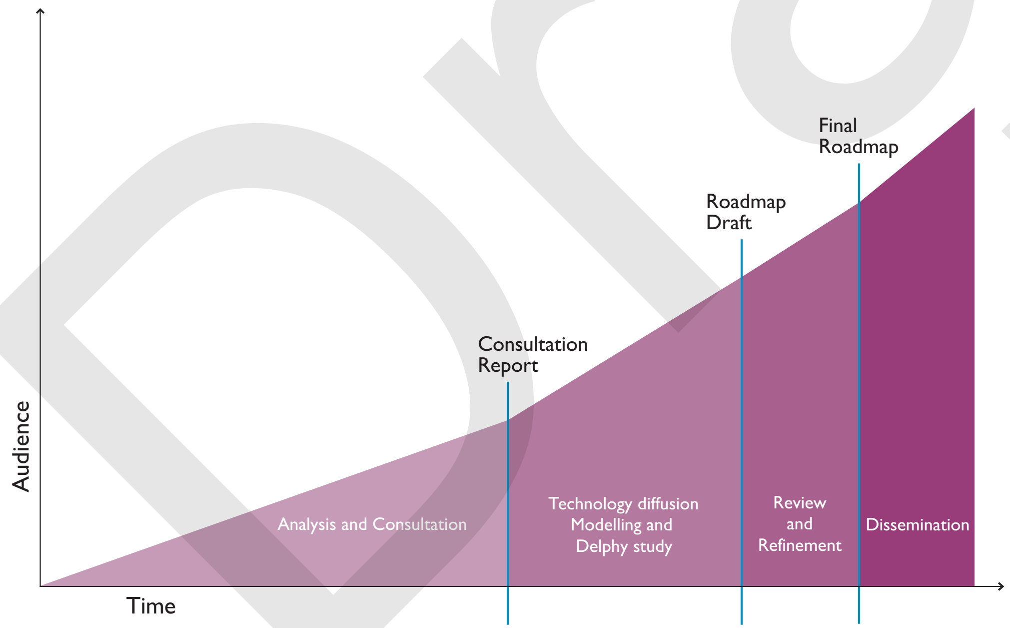


Figure A.1: Stages of roadmap development

Stage 1: The initial effort on roadmapping was primarily devoted to analysing the field of agent-based computing, as well as related fields, to determine the prevalent trends and drivers, and providing a broad assessment of the state-of-the-art in the research and development spheres. This involved both desk research on reports and papers, and discussion with leading thinkers at a range of important and relevant conferences, and culminated in the production of the consultation report, which was distributed with calls for contributions and participation. In addition, initial planning for two novel exercises was undertaken, on the Deliberative Delphi study, and on developing the technologies diffusion model.

Stage 2: After the Consultation Report was published, inputs from the AgentLink Technical Forum Groups and the wider community were solicited, and several presentations given, outlining the roadmapping process and the need for further efforts. The Deliberative Delphi study and the technology diffusion model were completed, and compiled into the Roadmap Draft, which is currently being distributed.

Stage 3: During the summer months, and until the end of August 2005, further specific comments and additions will be considered, focussed by this document. By October, the final revised document will be published, and will be widely distributed, both in print and electronic form. Results and conclusions will be presented to the broader community. This stage is intended to refine specific content in relation to details of the challenges and timelines presented, and represents the final opportunity for the community to contribute.

Full Members

Salzburg Research Forschungsgesellschaft mbH	Austria
Austrian Research Institute for Artifical Intelligence	Austria
CETIC	Belgium
K.U.Leuven	Belgium
Vrijie Universiteit Brussel	Belgium
Facultés Universitaires Notre-Dam de la Paix	Belgium
Self-Star Technologies	Belgium
Technical University of Crete	Crete
Czech Technical University	Czech Republic
CertiCon A.S.	Czech Republic
NeuroAgent Ltd	Finland
UTBM	France
University Paris Dauphine	France
Institut de Recherche en Infomatique de Toulouse	France
Team MAIA	France
France Telecom S.A	France
LCIS Research Laborbatory	France
Institut National Polytechnique de Grenoble	France
University Toulouse 1	France
LIRIS-CNRS, Université Claude Bernard-Lyon 1	France
LIP6 University Paris 6	France
LIFL - University of Lille 1	France
LIPN - CNRS UMR 7030	France
EADS Centre Commun de Recherche	France
MASA-SCI	France
Université de Pau et de A'dour	France
Ecole Nationale Superieure des Mines de Saint-Etiene	France
LIRMM - Universite de Montpellier II	France
Siemens AG Corporate Technology	Germany
Freie Universität Berlin	Germany
The Agent Factory GmbH	Germany
Friedrich-Schiller-Universität Jena	Germany
Technical University of Clausthal	Germany
Universität Hohenheim	Germany
Deutsches Forschungszentrum für Künstliche Intellengenz	Germany
University of Hamburg	Germany

Technische Universität Dresden	Germany
Technische Universität Muenchen	Germany
University of Karlsruhe	Germany
University of Bremen,	Germany
Technical University of Aachen	Germany
University of Augsburg	Germany
Cadence Design Systems GmbH	Germany
Frauhofer Institut fur Informations - und Datenverarbeitung	Germany
University of Rostock	Germany
Technische Universität Berlin	Germany
University of Bayreuth	Germany
Humboldt University at Berlin	Germany
University of Duisburg-Essen	Germany
DAI-Labor, Technische Universitaet Berlin	Germany
CITY College, Affiliated Institution of the University of Sheffield	Greece
University of Thessaly	Greece
The Centre for Research and Technology Hellas	Greece
University of Aegean	Greece
Hungarian Academy of Sciences	Hungary
AITIA Inc.	Hungary
IBM Israel	Israel
Hebrew University of Jerusalem	Israel
Bar-Ilan University	Israel
Ben-Gurion University of the Negev	Israel
Università di Bologna	Italy
University of Trento	Italy
University of Brescia	Italy
Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR-CNR)	Italy
University of Modena and Reggio Emilia	Italy
DIMET, Università Mediterranea di Reggio Calabria	Italy
Università di Torino	Italy
Università della Calabria	Italy
Università degli Studi Di Genova	Italy
Università degli Studi di Parma	Italy
University of Ferrara	Italy
University of Udine	Italy
ITC-irst (Istituto per la Ricerca Scientifica e Technologica)	Italy
Politecnico di Milano	Italy
Universita' degli Studi di L'Aquila	Italy
Università Politecnica delle Marche	Italy

University of Bari	Italy
University of Cagliali	Italy
Università di Padova	Italy
Fiat Research Center	Italy
Telecom Italia	Italy
Institute of Cognitive Sciences and Technology, CNR	Italy
University of Milan-Bicocca	Italy
Universita di Camerino	Italy
Institute of Computer Science, Polish Academy of Sciences	Poland
Instituto de Desenvolvimento e Inovaçäo Technológica	Portugal
Instituto Superior de Engenharia do Porto	Portugal
Universidade Do Porto	Portugal
Universidade de Lisboa	Portugal
Instituto Politécnico de Bragança	Portugal
Universidade Nova de Lisboa	Portugal
University Petroleum-Gas from Ploiesti	Romania
West University of Timisoara	Romania
Lucian Blaga University, Sibiu, Romania	Romania
University " Politehnica" of Burcharest	Romania
Wittmann & Partner Computer Systems S.R.L	Romania
Technical University of Cluj-Napoca	Romania
St Petersburg Institute For Infomatics and Automation	Russia
MIPT Computer Science Chair	Russia
University of Maribor	Slovenia
Institute Jozef Stefan	Slovenia
Universitat Politècnica de Catalunya	Spain
University of Girona	Spain
Universidad Rey Juan Carlos	Spain
Universidad Complutense Madrid	Spain
IIIA-CSIC	Spain
Universidad de Murcia,	Spain
Technical University of Madrid	Spain
Universidad Politécnica de Valencia	Spain
Universitat Rovira I Virgili	Spain
Agents Inspired Technologies S.A.	Spain
University of Vigo	Spain
Universitat Autonoma de Barcelona	Spain
University of Zaragoza	Spain
MICROART	Spain
University of Barcelona	Spain

Semantic Systems, S.A.	Spain
Stockholm University	Sweden
Swedish Institute of Computer Science	Sweden
Blekinge Institute of Technology	Sweden
Orebro University	Sweden
Whitestein Technologies AG	Switzerland
University of Geneva	Switzerland
Savannah Simulations	Switzerland
Acklin B.V.	The Netherlands
Tryllian Solutions BV	The Netherlands
Nederlands Organisation for Applied Scientific Research TNO	The Netherlands
Centrum voor Wiskunde en Informatica	The Netherlands
Rijksunversiteit Groningen	The Netherlands
Almende b.v.	The Netherlands
Vrije Universiteit Amsterdam	The Netherlands
University of Amsterdam	The Netherlands
Morpheus Software	The Netherlands
DECIS Lab	The Netherlands
University of Twente	The Netherlands
MP Objects	The Netherlands
Delft University of Technology	The Netherlands
Erasmus University Rotterdam	The Netherlands
Utrecht University	The Netherland
INITI8	The Netherlands
Universiteit Maastricht	The Netherlands
Bogazici University	Turkey
Ege University	Turkey
University of Liverpool	United Kingdom
University of Southampton	United Kingdom
British Telecommunications plc	United Kingdom
University of Nottingham	United Kingdom
City University, London	United Kingdom
University of Warwick	United Kingdom
Agent Oriented Software Limited	United Kingdom
Magenta Technology	United Kingdom
University of Bath	United Kingdom
Advanced Computation Lab, Cancer Research UK	United Kingdom
University of Surrey	United Kingdom
Manchester Metropolitan University	United Kingdom
De Montfort University	United Kingdom

Eurobios	United Kingdom
University of East London	United Kingdom
Calico Jack Ltd	United Kingdom
King's college London	United Kingdom
University of Dundee	United Kingdom
Sheffield Hallam University	United Kingdom
Cardiff University	United Kingdom
Oxford Brookes University	United Kingdom
Queen Mary & Westfield College, University of London	United Kingdom
UMIST	United Kingdom
General Dynamics (UK) Ltd	United Kingdom
School of Law, Edinburgh University	United Kingdom
University of Bradford	United Kingdom
Lost Wax Media Ltd	United Kingdom
GlaxoSmithKline	United Kingdom
The University of Edinburgh	United Kingdom
Agentis Software	United Kingdom
EDS Defense Ltd	United Kingdom
University College London	United Kingdom
University of Aberdeen	United Kingdom
University of Durham	United Kingdom
ISTRAT	United Kingdom
University of York	United Kingdom
The Macaulay Institute	United Kingdom
Cambridge Consultants Ltd	United Kingdom

Acknowledgements and information sources

AgentLink would like to thank all the organisations and individuals who have contributed, directly or indirectly in providing content and opinion in the development of this document and of the activities that will take place in the future: University of Southampton; University of Liverpool; European Commission; Institut d'Investigació en Intel.ligència Artificial, CSIC; Universitat Politècnica de Catalunya; International Conference on Autonomous Agents and Multi-Agent Systems, 2004; European Conference on Artificial Intelligence, 2004; European Workshop on Multi-Agent Systems, 2004; IEEE Systems, Man and Cybernetics (SMC) Conference, 2004.

AgentLink Technical Forum Groups

Agent-Oriented Software Engineering; Agents in Bioinformatics; Agents Applied in Healthcare; Environments for MAS; Intelligent Information Agents for Web Economies; Law and Electronic Agents; Multi-Agent Resource Allocation; Networked Agents; Programming Mult-Agent Systems; Self-Organisation in Multi-Agent Systems; Towards Semantic Web Agents; Trust for Open Collaborative Agent Business Environments.

AgentLink Roadmap Development

Core Team

Michael Luck	University of Southampton, UK
Peter McBurney	University of Liverpool, UK
Onn Shehory	IBM, Israel
Steve Willmott	Universitat Politècnica de Catalunya, Spain

Management Committee

Monique Calisti	Whitestein Technologies, Switzerland
Wiebe van der Hoek	University of Liverpool, UK
Michael Luck	University of Southampton, UK
Peter McBurney	University of Liverpool, UK
Jörg Müller	Siemens AG Corporate Technology, Germany
Andrea Omicini	University di Bologna, Italy
Terry Payne	University of Southampton, UK
Michal Pěchouček	Czech Technical University, Czech Republic
Onn Shehory	IBM, Israel
Simon Thompson	British Telecom, UK
Steve Willmott	Universitat Politècnica de Catalunya, Spain
Mike Wooldridge	University of Liverpool, UK

Other Contributors

Ronald Ashri	University of Southampton, UK
Petr Becvar	CertiCon, Czech Republic
Roxana Belecheanu	University of Southampton, UK
Fabio Bellifemine	Telecom Italia, Italy
Michael Berger	Siemens AG Corporate Technology, Germany
Carole Bernon	Université Paul Sabatier, France
Rafael Bordini	University of Durham, UK
Francesco Buccafurri	Università degli Studi Mediterranea di Reggio Calabria, Italy
Stefan Bussmann	Daimler Chrysler AG Research and Technology, Germany
Valérie Camps	Université Paul Sabatier, France
Carlos Carrascosa	Universidad Politécnica de Valencia, Spain
Cristiano Castelfranchi	University of Siena, Italy
Yann Chevaleyre	Université Paris Dauphine, France
Helder Coelho	Universitade Nova De Lisboa, Portugal
Ulises Cortes	Universitat Politècnica de Catalunya, Spain
Vince Darley	Eurobios, UK
Joris Deguet	Laboratoire Leibniz, France
Mark d'Inverno	University of Westminster, UK
Erik van Eekelen	MP Objects, The Netherlands
Ulle Endriss	Imperial College London, UK
Sylvia Estivie	Université Paris Dauphine, France
Michel Fabien	Université Montpellier II, France
Martyn Fletcher	Agent Oriented Software Ltd, UK
Jean-Pierre George	Université Paul Sabatier, France
Marie-Pierre Gleizes	Université Paul Sabatier, France
Pierre Glize	Université Paul Sabatier, France
Nathan Griffiths	University of Warwick, UK
Christian Herneth	Capgemini, Austria
Jon Himoff	Magenta Technology, UK
Gabriel Hopmans	Universiteit Maastricht, The Netherlands
Menno Jonkers	Tryllian, The Netherlands
Anthony Karageorgos	University of Thessaly, Greece
David Kinny	Agentis Software, USA
Stefan Kirn	University of Hohenheim, Germany
Elfriede Krauth	Erasmus University, The Netherlands
Habin Lee	British Telecom, UK
Michel Lemaitre	ONERA/DCSD/CD Centre de Toulouse, France
Victor Lesser	University of Massachusetts, USA
Beatriz Lopez	Universitat de Girona, Spain

Vincent Louis	France Telecom, France
Vladimir Mařík	Rockwell Automation, Czech Republic
Thierry Martinez	France Telecom, France
Giovanna Di Marzo Serugendo	University of Geneva, Switzerland
Viviana Mascardi	University of Genova, Italy
Nicolas Maudet	Université Paris Dauphine, France
Jez McKean	Jazzle, UK
Andre Meyer	Delft Cooperation on Intelligent Systems, The Netherlands
Ambra Molesini	Università degli Studi di Bologna, Italy
Luc Moreau	University of Southampton, UK
Steve Munroe	University of Southampton, UK
Pablo Noriega	Universitat Autònoma de Barcelona, Spain
Peter Novak	Technical University of Clausthal, Germany
Ann Nowe	Vrije Universiteit, Belgium
James Odell	Agentis Software, USA
Eugénio Oliviera	Universidade do Porto, Portugal
Steve Osborn	Lost Wax, UK
Sascha Ossowski	Universidad Rey Juan Carlos, Spain
Lin Padgham	RMIT, Australia
Juan Pavon	Universidad Complutense, Spain
Gauthier Picard	Université Paul Sabatier, France
Eric Platon	University of Tokyo, Japan
Agostino Poggi	Università degli Studi di Parma, Italy
Chris Preist	HP Labs, UK
Chris Reed	Calico Jack, UK
Juan A. Rodriguez	Universitat Autònoma de Barcelona, Spain
Josep Lluis de la Rosa	Universitat de Girona, Spain
Jeff Rosenschein	Hebrew University of Jerusalem, Israel
Nicolas Sabouret	Université Pierre et Marie Curie, France
Calin Sandru	West University of Timisoara, Romania
Jorge Gomez Sanz	Universidad Complutense de Madrid
Hayden Shorter	AePONA, UK
Carles Sierra	Universitat Autònoma de Barcelona, Spain
Liz Sonenberg	University of Melbourne, Australia
Paulo Sousa	Instituto Superior de Engenharia do Porto, Portugal
James Spillings	General Dynamics, UK
Susan Marie Thomas	SAP, Germany
Filip Verhaeghe	Self-Star Corporation, Belgium
George Vouros	University of the Aegean, Greece
George Weichhart	Profactor Produktionsforschungs GmbH, Austria

Danny Weyns	Katholieke Universiteit Leuven, Belgium
Nadezhda Yakounina	Magenta Technology, UK
Makoto Yokoo	University of Kyushu, Japan

AgentLink Support

Catherine Atherton	University of Liverpool, UK
Becky Earl	University of Southampton, UK
Adele Maggs	University of Liverpool, UK
Serena Raffin	University of Southampton, UK

Contact Information

Published by the University of Southampton on behalf of AgentLink III.
Contact: Prof Michael Luck
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
United Kingdom
mml@ecs.soton.ac.uk
+44 23 8059 6657

If your organisation would like to develop particular technology roadmaps that relates to the technologies and knowledge in this document, contact Michael Luck at the address above.

AgentLink III is a Coordination Action funded by the European Commission in Semantic-Based Knowledge Systems (Project Reference: IST 002006)

Designed and typeset by Serena Raffin.