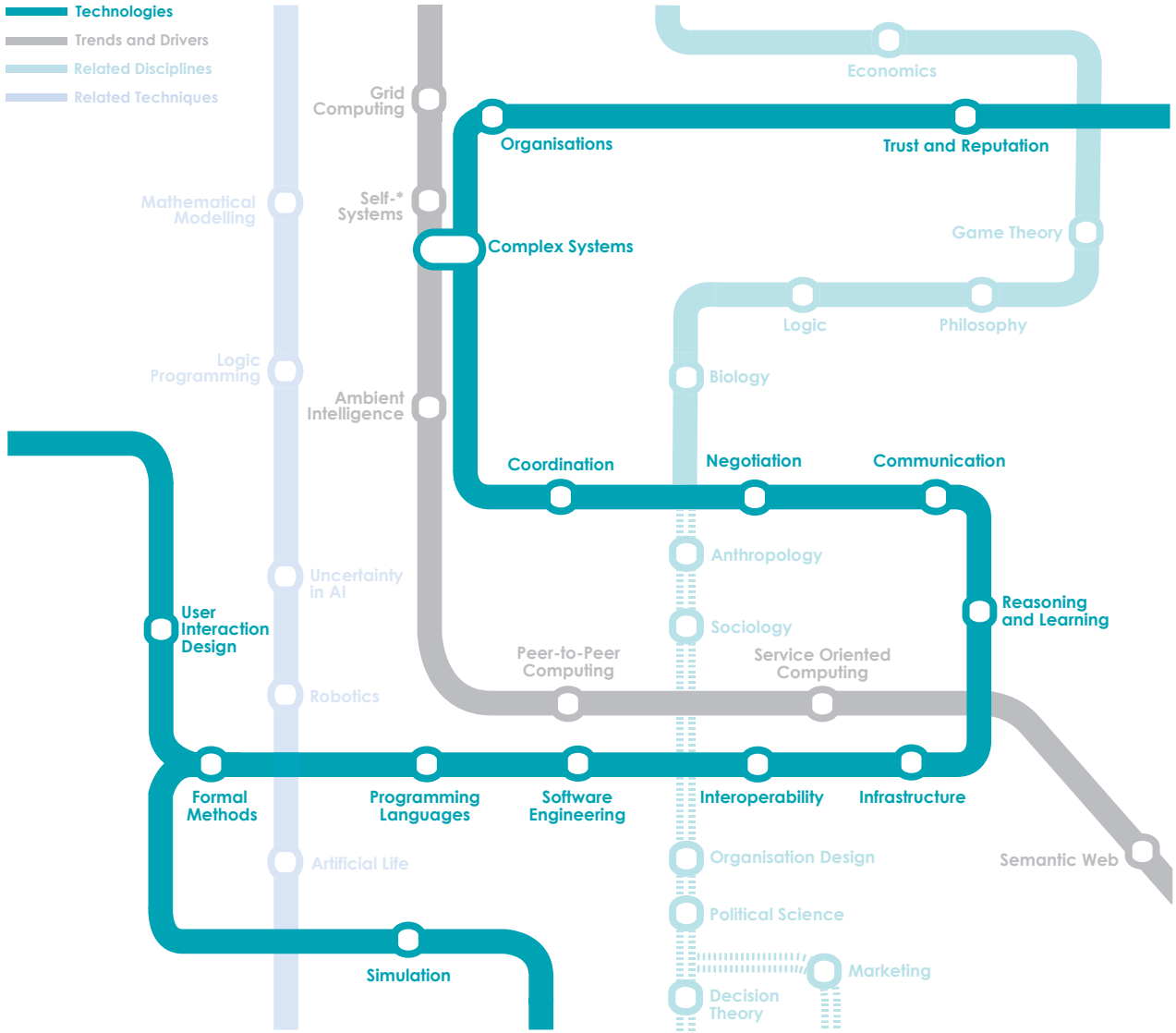
Agent Technology: Computing as Interaction

A Roadmap for Agent Based Computing



Compiled, written and edited by

Michael Luck, Peter McBurney, Onn Shehory, Steve Willmott and the AgentLink Community

Supported by

Michael Luck, Peter McBurney, Onn Shehory and Steven Willmott

ISBN 085432 845 9

This roadmap has been prepared as part of the activities of AgentLink III, the European Coordination Action for Agent-Based Computing (IST-FP6-002006CA). It is a collaborative effort, involving numerous contributors listed at the end of the report. We are grateful to all who contributed, including those not named.

Neither the editors, authors, contributors, nor reviewers accept any responsibility for loss or damage arising from the use of information contained in this report.

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

The cover was produced by Serena Raffin, based on an original design by Magdalena Koralewska.

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

The corresponding editor of this document is Michael Luck.

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, and its activities, is available from the AgentLink website at www.agentlink.org

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.

Contents

Executive Summary

In its brief history, computing has enjoyed several different metaphors for the notion of computation. From the time of Charles Babbage in the nineteenth century until the mid-1960s, most people thought of computation as *calculation*, or operations undertaken on numbers. With widespread digital storage and manipulation of non-numerical information from the 1960s onwards, computation was re-conceptualised more generally as *information processing*, or operations on text, audio or video data. With the growth of the Internet and the World Wide Web over the last fifteen years, we have reached a position where a new metaphor for computation is required: computation as *interaction*. In this metaphor, computing is something that happens by and through communication between computational entities. In the current radical reconceptualisation of computing, the network is the computer, to coin a phrase.

In this new metaphor, computing is an activity that is inherently social, rather than solitary, leading to new ways of conceiving, designing, developing and managing computational systems. One example of the influence of this viewpoint is the emerging model of software as a service, for example in service-oriented architectures. In this model, applications are no longer monolithic, functioning on one machine (for single user applications), or distributed applications managed by a single organisation (such as today's Intranet applications), but instead are societies of components.

- These components are viewed as providing services to one another. They may not all have been designed together or even by the same software development team; they may be created, operate and be decommissioned according to different timescales; they may enter and leave different societies at different times and for different reasons; and they may form coalitions or virtual organisations with one another to achieve particular temporary objectives. Examples are automated procurement systems comprising all the companies connected along a supply chain, or service creation and service delivery platforms for dynamic provision of value-added telecommunications services.

- The components and their services may be owned and managed by different organisations, and thus have access to different information sources, have different objectives, and have conflicting preferences. Health care management systems spanning multiple hospitals or automated resource allocation systems, such as Grid systems, are examples here.

- The components are not necessarily activated by human users but may also carry out actions in an automated and coordinated manner when certain conditions hold. These preconditions may themselves be distributed across components, so that action by one component requires prior co-ordination and agreement with other components. Simple multi-party database commit protocols are examples, but significantly more complex coordination and negotiation protocols have been studied and deployed, for example in utility computing systems and ad hoc wireless networks.

- Intelligent, automated components may even undertake self-assembly of software and systems, to enable adaptation or response to changing external or internal circumstances. An example of this is the creation of on-the-fly coalitions in automated supply-chain systems in order to exploit dynamic commercial opportunities. Such systems resemble those of the natural world and human societies much more than they do the example arithmetic programs taught in Fortran classes, so ideas from biology, statistical physics, sociology and economics play an increasingly important role in computing systems.

How should we exploit this new metaphor of computing as social activity, as interaction between independent and sometimes intelligent entities, adapting and co-evolving with one another? The answer, many people believe, lies with agent technologies. An agent is a computer program capable of flexible and autonomous action in a dynamic environment, usually an environment containing other agents. In this abstraction, we have encapsulated autonomous and intelligent software entities, called agents, and we have demarcated the society in which they operate, a multi-agent system. Agent-based computing concerns the theoretical and practical working through of the details of this simple two-level abstraction.

In the sense that it is a new paradigm, agent-based computing is *disruptive*. As outlined above, it causes a re-evaluation of the very nature of computing, computation and computational systems, through concepts such as *autonomy*, *coalitions* and *ecosystems*, which make no sense to earlier paradigms. Economic historians have witnessed such disruption with new technologies repeatedly, as new technologies are created, are adopted, and then mature. A model of the life-cycle of such technologies, developed by Perez (2002), and reproduced in Figure 0.1, suggests two major parts: an *installation* period of exploration and development; and a *deployment* period concentrating on the use of the technology. As will be argued later in this document, agent technologies are still in the early stages of adoption, the stage called *irruption* in this life-cycle. In the chapters that follow, we examine the current status of agent technologies and compare their market diffusion to related innovations, such as object technologies. We also consider the challenges facing continued growth and adoption of agent technologies.

This document is a strategic roadmap for agent-based computing over the next decade. It has been prepared by AgentLink III, a European Commission-funded coordination action, intended to support and facilitate European research and development in agent technologies. The contents of the roadmap are the result of an extensive, eighteen-month effort of consultation and dialogue with experts in agent technology from the 192 member organisations of AgentLink III, in addition to experts in the Americas, Japan and Australasia. The roadmap presents our views of how the technology will likely develop over the decade to 2015, the key research and development issues involved in this development, and the challenges that currently confront research, development and further adoption of agent technologies.

This strategic technology roadmap is not intended as a prediction of the future. Instead, it is a reasoned analysis: given an analysis of the recent past and current state of agent technologies, and of computing more generally, we present one possible future development path for the technology. By doing this, we aim to identify the challenges and obstacles that will need to be overcome for progress to be made in research and

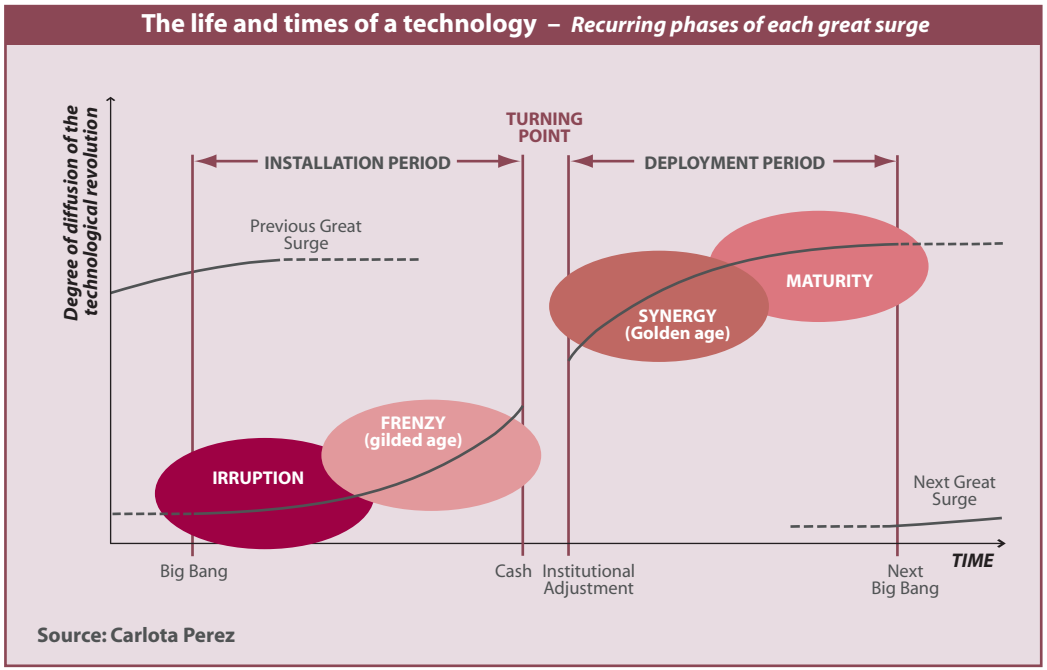


Figure 0.1: The phases of technology life-cycles. Source: Carlota Perez

development, and for greater commercial adoption of the technology to occur. Moreover, by articulating a possible future path and identifying the challenges to be found along that path, we hope to galvanise the attention and efforts both of the agent-based computing community and of the IT community more generally: these challenges and obstacles will only be overcome with concerted efforts by many people. We hope the ideas presented here are provocative, because a strategic roadmap should not be the end of a dialogue, but the beginning.

1 What is Agent Technology?

Agent-based systems are one of the most vibrant and important areas of research and development to have emerged in information technology in the 1990s. Put at its simplest, an agent is a computer system that is capable of flexible autonomous action in dynamic, unpredictable, typically multi-agent domains. In particular, the characteristics of dynamic and open environments in which, for example, heterogeneous systems must interact, span organisational boundaries, and operate effectively within rapidly changing circumstances and with dramatically increasing quantities of available information, suggest that improvements on traditional computing models and paradigms are required. Thus, the need for some degree of autonomy, to enable components to respond dynamically to changing circumstances while trying to achieve over-arching objectives, is seen by many as fundamental. Many observers therefore believe that agents represent the most important new paradigm for software development since object orientation.

The concept of an agent has found currency in a diverse range of sub-disciplines of information technology, including computer networks, software engineering, artificial intelligence, human-computer interaction, distributed and concurrent systems, mobile systems, telematics, computer-supported cooperative work, control systems, decision support, information retrieval and management, and electronic commerce. In practical developments, web services, for example, now offer fundamentally new ways of doing business through a set of standardised tools, and support a service-oriented view of distinct and independent software components interacting to provide valuable functionality. In the context of such developments, agent technologies have increasingly come to the foreground. Because of its horizontal nature, it is likely that the successful adoption of agent technology will have a profound, long-term impact both on the competitiveness and viability of IT industries, and on the way in which future computer systems will be conceptualised and implemented. Agent technologies can be considered from three perspectives, each outlined below, as illustrated in Figure 1.1.

1.1 Agents as Design Metaphor

Agents provide software 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 computational 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 software components, tools and abstractions are required to enable developers to deal with the complexity of implemented systems, and

supporting infrastructure (embracing other relevant, widely used 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 and cooperate with other agents, and developing appropriate means of forming and managing coalitions (and other organisations). Moreover, the adoption of agent-based approaches is increasingly influential in other domains. For example, multi-agent systems are already providing new and more effective methods of resource allocation in complex environments than previous approaches.

1.3 Agents as Simulation

Multi-agent systems offer strong models for representing complex and dynamic real-world environments. For example, simulation of economies, societies and biological environments are typical application areas.

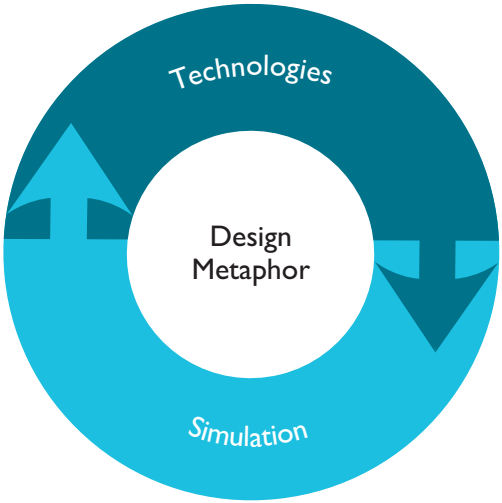
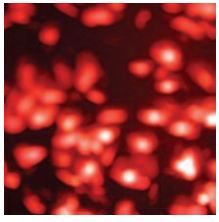


Figure 1.1: Agent-based computing spans technologies, design and simulation

The use of agent systems to simulate real-world domains may provide answers to complex physical or social problems that would otherwise be unobtainable due to the complexity involved, 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 management systems.



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.

We do not claim that agent systems are simply panaceas for these large problems; rather 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.

Acklin and International Vehicle Insurance Claims

Netherlands-based Acklin BV was asked by a group of three insurance companies, from Belgium, the Netherlands and Germany, to help automate their international vehicle claims processing system. At present, European rules require settlement of cross-border insurance claims for international motor accidents within 3 months of the accident. However, the back-office systems used by insurance companies are diverse, with data stored and used in different ways. Because of this and because of confidentiality concerns, information between insurance companies is usually transferred manually, with contacts between claim handlers only by phone, fax and email. Acklin developed a multi-agent system, the KIR system, with business rules and logic encoded into discrete agents representing the data sources of the different companies involved. This approach means the system can ensure confidentiality, with agent access to data sources mediated through other agents representing the data owners. Access to data sources is only granted to a requesting agent when the relevant permissions are present and for specified data items. Because some data sources are only accessible during business hours, agents can also be programmed to operate only within agreed time windows. Moreover, structuring the system as a collection of intelligent components in this way also enables greater system robustness, so that business processes can survive system shutdowns and failures. The deployment of the KIR system immediately reduced the human workload at one of the participating companies by three people, and reduced the total time of identification of client and claim from 6 months to 2 minutes! For reasons of security, the KIR system used email for inter-agent communication, and the 2 minutes maximum time is mainly comprised of delays in the email servers and mail communication involved.

2 Technological Context

The growth of the World Wide Web and the rapid rise of eCommerce have led to significant efforts to develop standardised software models and technologies to support and enable the engineering of systems involving distributed computation. These efforts are creating a rich and sophisticated context for the development of agent technologies. For example, so-called service-oriented architectures (SOAs) for distributed applications involve the creation of systems based on components, each of which provides pre-defined computational services, and which can then be aggregated dynamically at runtime to create new applications. Other relevant efforts range from low-level wireless communications protocols such as Bluetooth to higher-level web services abstractions and middleware.

The development of standard technologies and infrastructure for distributed and eCommerce systems has impacted on the development of agent systems in two major ways.

- Many of these technologies provide implementation methods and middleware, enabling the easy creation of infrastructures for agent-based systems, such as standardised methods for discovery and communication between heterogeneous services.
- Applications now enabled by these technologies are becoming increasingly agent-like, and address difficult technical challenges similar to those that have been the focus of multi-agent systems. These include issues such as trust, reputation, obligations, contract management, team formation, and management of large-scale open systems.

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

- Base Technologies:
  - The Extensible Markup Language (XML) is a language for defining mark-up languages and syntactic structures for data formats. Though lacking in machine-readable semantics, XML has been used to define higher-level knowledge representations that facilitate semantic annotation of structured documents on the Web.
  - The Resource Description Format (RDF) is a representation formalism 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 and discovering businesses 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/WS-CDL: 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. WS-CDL allows the definition of abstract interfaces of web services, that is, the business-level conversations or public processes supported by a web service.

Conversely, agent-related activities are already beginning to inform development in a number of these technology areas, including the Semantic Web standardisation efforts of the World Wide Web Consortium (W3C), and the Common Object Request Broker Architecture (CORBA) of the Object Management Group (OMG). Contributions have also come through

the Foundation for Intelligent Physical Agents (FIPA; accepted in 2005 by the IEEE as its eleventh standards committee), which defines a range of architectural elements similar to those now adopted in the W3C Web Services Architecture specifications and elsewhere.

These developments with regard to the technological context for agent systems are illustrated in Figure 2.1, which presents the main contextual technologies supporting agent systems development. While research in agent technologies has now been active for over a decade, the figure shows that it is only from 1999, with the appearance of effective service-oriented technologies and pervasive computing technologies, that truly dynamic (ad hoc) networked systems could be built without large investments in establishing the underlying infrastructure. In particular, only with the emergence of Grid computing from 2002, and calls for adaptive wide-scale web service based solutions, is there now a widespread need to provide attractive solutions to the higher-level issues of communication, coordination and security.

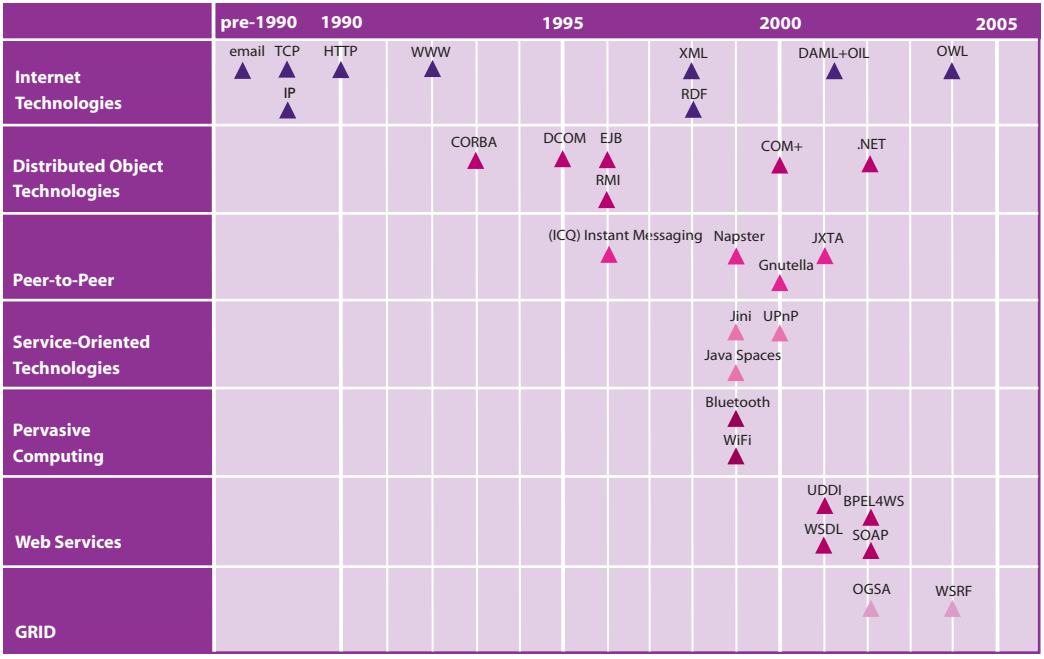


Figure 2.1: Agent-related technologies for infrastructure support

Eurobios and SCA Packaging

Many companies find themselves under strong pressures to deliver just-in-time high quality products and services, while operating in a highly competitive market. In one of SCA Packaging's corrugated box plants, customer orders often arrive simultaneously for a range of different boxes, each order with its own colour scheme and specific printing, and often to be delivered at very short notice. Because of the complexity of factory processes and the difficulty of predicting customer behaviour and machine failure, large inventories of finished goods must therefore be managed. SCA Packaging turned to Eurobios to provide an agent-based modelling solution in order to explore different strategies for reducing stock levels without compromising delivery times, as well as evaluating consequences of changes in the customer base. The agent-based simulation developed by Eurobios allowed the company to reduce warehouse levels by over 35% while maintaining delivery commitments.

In general, it is clear that broad technological developments in distributed computation are increasingly addressing problems long explored within the agent research community. There are two inter-related developments here. First, supporting technologies are emerging very quickly. As a consequence, the primary research focus for agent technologies has moved from infrastructure to the higher-level issues concerned with effective coordination and cooperation between disparate services. Second, large numbers of systems are being built and designed using these emerging infrastructures, and are becoming ever more like multi-agent systems; their developers therefore face the same conceptual and technical challenges encountered in the field of agent-based computing.

3 Emerging Trends and Critical Drivers

The development of agent technologies has taken place within a context of wider visions for information technology. In addition to the specific technologies mentioned in the previous section, there are also several key trends and drivers that suggest that 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.

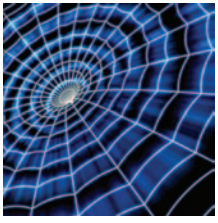
The Semantic Web is based on the idea that the data on the Web can be defined and linked in such a way that it can be used by machines for the automatic processing and integration of data across different applications (Berners-Lee et al., 2001). This is motivated by the fundamental recognition that, in order for web-based applications 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 in such a way that it is possible for machines to reason automatically 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 the two fields are intimately connected. Indeed, the Semantic Web 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.

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 value-added 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, etc), Grid architectures (which are now increasingly based on web services infrastructures) and others, towards a more general notion of service-oriented architectures (SOA). Here, distributed systems are increasingly viewed as collections of service provider and service consumer components, interlinked by dynamically defined workflows. Web services can therefore be realised by 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, this infrastructure 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, since provider and consumer web services environments are naturally seen as a form of agent-based system (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 (or 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, scalability and maintenance (Milojicic et al., 2002).

The best known P2P applications include hugely popular file sharing applications such as Gnutella and Bit Torrent, Akamai 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 (GGF), which now includes a 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 encourage correct behaviour of clients. (For example, many commercial e-marketplace systems, such as eBay, include simple credit-reputation systems to reward socially beneficial behaviour). As P2P systems become more complex, an increasing number of agent technologies may also become relevant. These include, for example: auction mechanism design to provide a rigorous basis 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 approaches to 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 agents.

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). The Grid 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 systems are 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 computer processing, and often require secure resource sharing across organisational boundaries; they are thus not easily handled by today's Internet and Web infrastructures.

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,

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-effective basis, and achieves a closer 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.

in principle enabling more flexible IT deployment 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 computing 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 the creation of products, and efforts in this direction are already underway from a range of vendors, including Sun, IBM and HP.

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 components) interacting to support user-centred goals and activity, and suggests a component-oriented view of the world in which the components are independent and distributed. The consensus is that autonomy, distribution, adaptation, responsiveness, and so on, are key characteristics of these 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 collaboration 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 virtual organisations, in which groups of agents come together to form coherent groups able to achieve overarching objectives.

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 and connectivity of sensors, actuators and networks, for example. On the other hand, 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, as well as heterogeneity of agents and services, is facilitated by the provision of appropriate ontologies. 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 Autonomic Computing

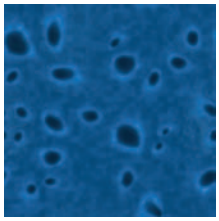
Computational systems that are able to manage themselves have been part of the vision for computer science since the work of Charles Babbage. With the increasing complexity of advanced information technology systems, and the increasing reliance of modern society on these systems, attention in recent years has returned to this. Such systems have come to be called self-* systems and networks (pronounced "self-star"), 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, self correction, 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 explosion of information, the integration of technology into everyday life, and the associated problems of complexity in managing and operating computer systems, autonomic computing takes inspiration from the autonomic function of the human central nervous system, which controls key functions without conscious awareness or involvement. First proposed by IBM (Kephart and Chess, 2003), 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 gives 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 other systems, by negotiating for resource use; 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 aim 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 the complexity of self-* and autonomic systems.

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 nature of these interactions being poorly understood. The science of complex adaptive systems is still in its infancy, and as yet provides little in the way of guidance for designers and controllers of specific systems.

Whether such complex, adaptive systems are explicitly 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 a valuable way of coping with the increasing complexity of modern software systems (Zambonelli and Parunak, 2002), particularly the characteristics of 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, 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

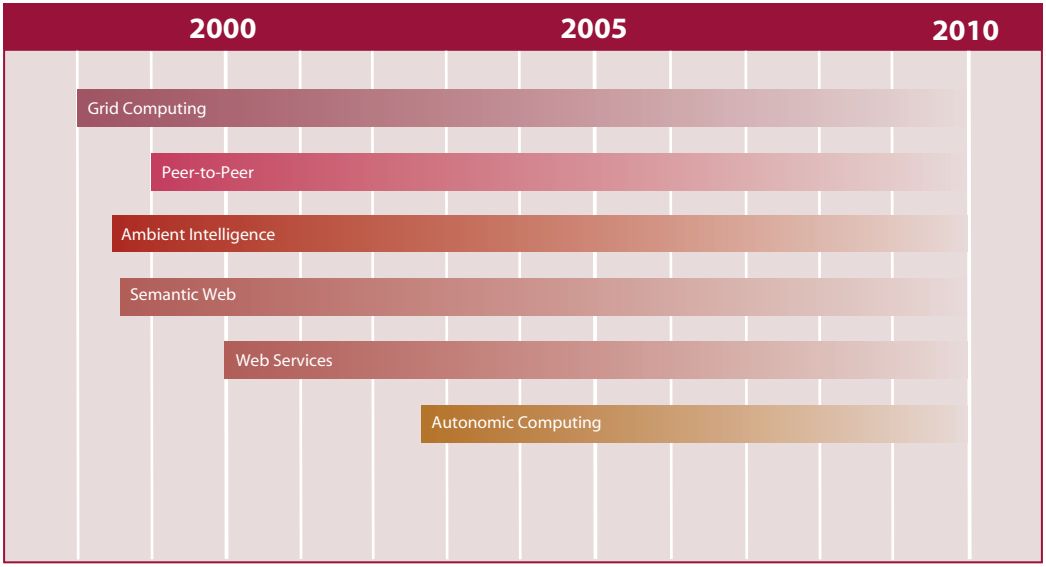


Figure 3.1: The emergence of agent-related domains over time.

NuTech and Air Liquide

Air Liquide America LP, a Houston-based producer of liquefied industrial gases with more than 8000 customers worldwide, turned to agent technology to reduce production and distribution costs. The system was developed by NuTech Solutions, using a multi-agent ant system optimisation approach combined with a genetic algorithm and a suite of expert heuristics. The ant system optimiser discovered efficient product distribution routes from the plant to the customer, while the genetic algorithm was implemented to search for highly optimal production level schedules for individual plants. As a result of using the system, Air Liquide America managed to reduce inefficiencies in the manufacturing process, adapt production schedules to changing conditions and deliver products cost-effectively, where and when the customer demands, and in a manner that is responsive to unexpected events. Together, these benefits offered Air Liquide an optimal cost product with the potential of new market opportunities and operational savings.

to engage in interactions, to negotiate, to make pro-active run-time decisions while responding to changing circumstances, and to allocate and schedule resources across the diverse competing demands placed on infrastructures and systems. In particular, agents with different capabilities will need to collaborate and to form coalitions in support of new virtual organisations.

Of course, these drivers do not cover all 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, supporting or 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

It should be clear that there are several distinct high-level trends and drivers leading to interest in agent technologies, and low-level computing infrastructures making them practically feasible. In this context, we now consider the key technologies and techniques required to design and implement agent systems that are the focus of current research and development. Because agent technologies are mission-critical for engineering and for managing certain types of information systems, such as Grid systems and systems for ambient intelligence, the technologies and techniques discussed below will be important for many applications, even those not labelled as agent systems.

These technologies can now be grouped into three categories, according to the scale at which they apply:

- Organisation-level: At the top level are technologies and techniques related to agent societies as a whole. Here, issues of organisational structure, trust, norms and obligations, and self-organisation in open agent societies are paramount. Once again, many of these questions have been studied in other disciplines — for example, in sociology, anthropology and biology. Drawing on this related work, research and development is currently focused on technologies for designing, evolving and managing complex agent societies.
- Interaction-level: These are technologies and techniques that concern the communications between agents — for example, technologies related to communication languages, interaction protocols and resource allocation mechanisms. Many of the problems solved by these technologies have been studied in other disciplines, including economics, political science, philosophy and linguistics. Accordingly, research and development is drawing on this prior work to develop computational theories and technologies for agent interaction, communication and decision-making.
- Agent-level: These are technologies and techniques concerned only with individual agents — for example, procedures for agent reasoning and learning. Problems at this level have been the primary focus of artificial intelligence since its inception, aiming to build machines that can reason and operate autonomously in the world. Agent research and development has drawn extensively on this prior work, and most attention in the field of agent-based computing now focuses at the previous two higher levels.

In addition to technologies at these three levels, we must also consider technologies providing infrastructure and supporting tools for agent systems, such as agent programming languages and software engineering methodologies. These supporting technologies and techniques provide the basis for both the theoretical understanding and the practical implementation of agent systems.

4.1 Organisation Level

4.1.1 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. They will arise in dynamic (or emergent) agent societies, such as those suggested by the Grid, ambient intelligence and other domains in which agents come together to deliver composite services, all of which 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, which is fundamental to realising the power of the agent paradigm; 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, methods for 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 decade as we seek ways to structure interactions in an open and dynamic online 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. These social factors can provide the basis on which to develop methods for access control, for example, and to ensure that behaviour is regulated and structured when faced with dynamic environments in which traditional techniques are not viable. In addition to appropriate methods and technologies for agent team formation, management, assessment, coordination and dissolution, technologies will also be required for these processes to be undertaken automatically at runtime in dynamic environments.

4.1.2 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. This can often result 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 through contextual local interactions, without central control. 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 ants foraging for food, molecule formation, and antibody detection. Similarly, current software applications involve social interactions (such as negotiations and transactions) with autonomous entities or agents, in highly dynamic environments. Engineering applications to achieve robustness and adaptability, based on the principles of self-organisation, is thus gaining increasing interest in the software community. This interest 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 (or the means by which the individual parts of a system communicate with one another by modifying their local environment, much like ants), 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 by 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 high-dimensional 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 system behaviours that 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 can increase 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. In consequence, therefore, new theories of emergence need to be developed based on inspiration from natural or social 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.

4.1.3 Trust and Reputation

Many applications involving multiple individuals or organisations must take into account the relationships (explicit or implicit) between participants. Furthermore, individual agents 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 agents using financial disincentives 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), other social sciences (social laws, social power) or mathematics (game theory and mechanism design).

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, when considering agents autonomously acting on behalf of their owner several additional factors need to be addressed. In particular, 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 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.

4.2 Interaction Level

4.2.1 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. In turn, cooperation 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) and 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).

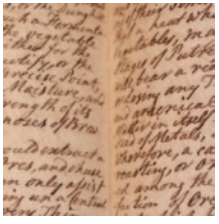
4.2.2 Negotiation

Goal-driven agents in a multi-agent society typically have conflicting goals; in other words, not all agents may be able to satisfy their respective goals simultaneously. This may occur, for example, with regard to contested resources or with multiple demands on an agent's time and attention. In such circumstances, agents will need to enter into negotiations with each other to resolve conflicts. Accordingly, considerable effort has been devoted to negotiation protocols, resource-allocation methods, and optimal division procedures. This work has drawn on ideas from computer science and artificial intelligence on the one hand, and the socio-economic sciences on the other.

For example, 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. Many concepts studied in social choice theory can 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 (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 (Lemaître et al., 2003).

While much recent work on resource allocation has concentrated on centralised approaches, in particular combinatorial auctions (Cramton et al., 2006), many applications are more naturally modelled as truly distributed or P2P 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). A new challenge in the field of multi-agent 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 its trustworthiness), but which also provide 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, as well as a new perspective on what "optimal" means in a distributed setting.

Other negotiation techniques are also likely to become increasingly prevalent. For example, one-to-one negotiation, or bargaining, over multiple parameters or attributes to establish service-level agreements between service providers and service consumers will be key in future service-oriented computing environments. In addition to approaches drawn from economics and social choice theory in political science, recent efforts in argumentation-based negotiation have drawn on ideas from the philosophy of argument and the psychology of persuasion. These efforts potentially provide a means to enable niches of deeper interactions between agents than do the relatively simpler protocols of economic auction and negotiation mechanisms. Considerable research and development efforts will be needed to create computational mechanisms and strategies for such interactions, and this is likely to be an important focus of agent systems research in the next decade.



4.2.3 Communication

Agent communication is the study of how two or more software entities may communicate with each other. The research issues 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 depends 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 is not normally 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.

Key to this area is the need 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 rigorously compare one language or protocol with another. In addition, progress towards understanding the applicability of different agent communication languages, content langauges and protocols in different application domains is necessary for wider adoption of research findings.

4.3 Agent Level

Reasoning is a critical faculty of agents, but the extent to which it is needed is determined by 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, which can underpin the development and deployment of dynamic electronic institutions or other open multi-agent system. 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 can in complex settings, and is required, for example, for Grid applications.

One enabler for this is negotiation, yet while there have already been significant advances and real-world applications, 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 argumentation techniques 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 also relatively new, and considerably more work is needed here. Both these topics were considered in the discussion on negotiation above.

Even though learning technology is clearly important 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 the fundamental difficulty of learning, but also 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.

4.4 Infrastructure and Supporting Technologies

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, as well as standards, will be crucial for the medium-term development of agent systems.

4.4.1 Interoperability

At present, the majority of agent applications exist in academic and commercial laboratories, but are not widely available in the real world. 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 plays 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 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 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 agent components across the layers of a single system, among multiple computing systems, as well as between different self-organisation frameworks.

4.4.2 Agent Oriented Software Engineering

Despite a number of languages, frameworks, development environments, and platforms that have appeared in the literature (Luck et al., 2004b), 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 life-cycle, 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 previous unavoidable tight coupling of agent IDEs and agent platforms, which results from the variety of agent models, platforms and programming languages. This is now changing, however, with an increased trend towards modelling rather than programming.

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, but 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, analogous to the remote debugging tools in current object-oriented IDEs, are needed, and some are now starting to appear (Botía et al, 2004). 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.

The inherent complexity of agent applications also demands a new generation of CASE tools to assist application designers in harnessing the large amount of information involved. This requires 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.

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.

Metrics for agent-oriented software are also needed: 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 multi-agent systems applications (be they self-organising or not) are needed.

4.4.3 Agent Programming Languages

Most research in agent-oriented programming languages is based on declarative approaches, mostly logic based. Imperative languages are in essence inappropriate for expressing the high-level abstractions associated with agent systems design; however, agent-oriented programming languages should (and indeed tend to) allow for easy integration with (legacy) code written in imperative languages. From the technological 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 niche applications, they remain unused in practice. However, recent years have seen a significant increase in the maturity of such languages, and major improvements in the development platforms and tools that support them (Bordini et al., 2005).

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 (as well as more fundamental concerns relating to the lack of clear semantics for agents, coordination, etc), 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-oriented 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, and provide interfaces to existing mainstream languages for easy integration with code and legacy packages. While existing agent languages already address some of these issues, further progress is expected in the short terrm, but thorough practical experimentation in real-world settings (particularly large-scale systems) will be required before such languages can be adopted by industry, in the medium to long term.

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 is a longer term goal.

4.4.4 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 and analysed 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 logics of belief or other modalities, along with temporal modalities, and such logics require efficient theorem-proving or model-checking algorithms when applied to problems of significant scale. Recent efforts have used logical formalisms to represent social properties, such as coalitions of agents, preferences 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.

4.4.5 Simulation

As mentioned earlier, agent-based computing provides 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 agent-based 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? How can we avoid over-interpretation of results with abstract or vague terms? 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 arising from the Kyoto Protocol to the UN Framework Convention 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.

4.4.6 User Interaction Design

In future complex system environments, human involvement 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, such as those caused by the 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 (hybrid systems). 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. In addition, questions of decision-making authority, responsibility, delegation and control arise with systems of agents acting on behalf of, or in collaboration with, human decision-makers in mixed initiative systems. If agents or multi-agent systems are themselves responsible for decisions, these issues become more problematic (see Kuflik, 1999).

5 Adoption of Agent Technologies

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, or because the needs of potential customers change with time. Most high-technology products are

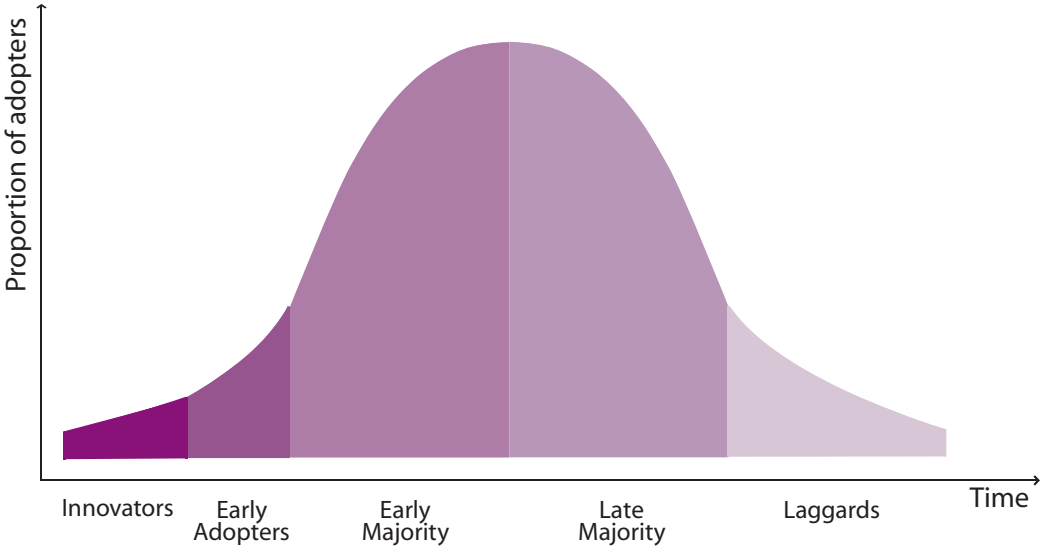


Figure 5.1: The technology adoption life-cycle

adopted initially only by people or companies with a keen interest in that type of new technology and the disposable income to indulge their interest. Thus, early adopters are often technologically sophisticated, well-informed, and 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 several reasons for this, as follows.

- Potential adopters must 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 may need to be paid for from sales of the product, thus delaying uptake of the product or service.
- Once they learn about a new technology, not all eventual adopters 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. 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 adopters 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 adopters 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 life-cycle 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 did 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 adopters creates particular challenges for marketers (Moore, 1991). This has led to the notion of a "chasm" between one adopter segment and the next as shown in Figure 5.2, in which the gaps between segments indicate that users in adjacent segments are distinct.

How quickly do new products and technologies reach saturation? If one considers an innovation such as written communication, which began several thousand years ago,

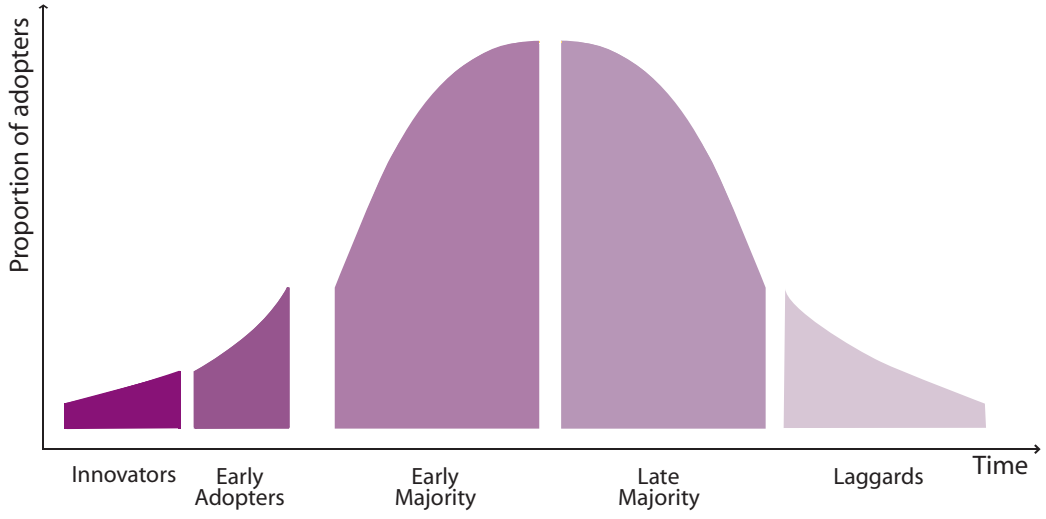


Figure 5.2: The revised technology adoption life-cycle

diffusion has been very slow. It is unfortunate but true that perhaps as many as half the world's population are still unable to read and write. In contrast, cellular mobile telephones are now used by almost 1.7 billion people, a position reached in just over two decades from the launch of the first public cellular networks (IDC, 2005).

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. 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

accepted by the IEEE as its eleventh standards committee), the Object Management Group, the Global Grid Forum, and the World Wide Web Consortium. 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 seek to divide and conquer the various competing standards bodies by, for example, participating intensely in one standards organisation at one time and another competing 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, 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 depends 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).

5.4 Agent Technologies

With this marketing background , it is useful to consider the position of agent-based computer technologies. Adoption of agent technologies has not yet entered the mainstream of

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 not 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 is the case now. Consequently, the changes required for the adoption of objects was far less substantial and challenging than it is now for agent technologies.

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 likely to be even 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.

commercial organisations, unlike, for example, object-oriented technologies. Indeed, the majority of commercial organisations adopting agent technologies might be classified as early adopters, since only a relatively small number of deployed commercial and industrial applications of agent technology are visible, and because 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, such as 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 agent technology is also still only in its infancy. Here, a reasonable comparison is with object-oriented (OO) programming approaches, where the initial research commenced in 1962 (see box), more than 20 years before the advent of C++, and some 32 years before the public release of the first version of Java, both key points for the widespread commercial adoption of OO technologies (and 39 years before the two original researchers, Ole-Johan Dahl and Kristen Nygaard, received a Turing Award for their work). 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 to 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 example, agent-based simulation for delivery schedule decision-making) many potential applications of agent technologies require the participation of entities from more than one group or organisation. Automated purchase decisions along a supply-chain, for example, require the participation of the companies active along that chain, so that implementing a successful agent-based application requires 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, such as those 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 to give indicative insight into 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 in the model 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. These different topologies are presented in detail in (McKean et al., 2005).

Nodes were then modelled as independent and autonomous decision-makers, each making decisions to move (or not) through a technology adoption life-cycle. The life-cycle

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].

began with non-adoption, and progressed through consideration, trial, partial adoption and full adoption. At each stage in the life-cycle, 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 depended 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 factors included elements such as: organisational needs for the technology; the costs of adoption; the presence of complementary software tools; and the presence of a technology standard or multiple standards.

For each node and for each decision, these factors were then combined through a factor-weighting mechanism, the outcome of which 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 life-cycle. The weighting mechanism differs across the states of the technology adoption life-cycle 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 have 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, so it may be possible to calibrate the model in this way in future work. Further information about the design and implementation of the model can be found in (McKean et al., 2005).

5.5.2 Simulation Results

One thousand simulation runs with random starting values were undertaken for each network topology, 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 then averaged across the 1000 simulation runs, with results 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 simulations.

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 or research institutes. 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

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: Average numbers of generations to 100% adoption (by topology and numbers of standards).

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 (Interagency Working Group, 2003).

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

Magenta Technology and Tankers International

Tankers International, which operates one of the largest oil tanker pools in the world, has applied agent technology to dynamically schedule the most profitable deployment of ships-to-cargo for its Very Large Crude Carrier fleet. An agent-based optimiser, Ocean i-Scheduler, was developed by Magenta Technology for use in real-time planning of cargo assignment to vessels in the fleet. The system can dynamically adapt plans in response to unexpected changes, such as transportation cost fluctuations or changes to vessels, ports or cargo. Agent-based optimisation techniques not only provided improved responsiveness, but also reduced the human effort necessary to deal with the vast amounts of information required, thus reducing costly mistakes, and preserving the knowledge developed in the process of scheduling.

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, through Coordination Actions like AgentLink, 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 activity in Europe, with AgentLink and Agentcities.NET providing coordination of the community through a period of intense change and innovation at

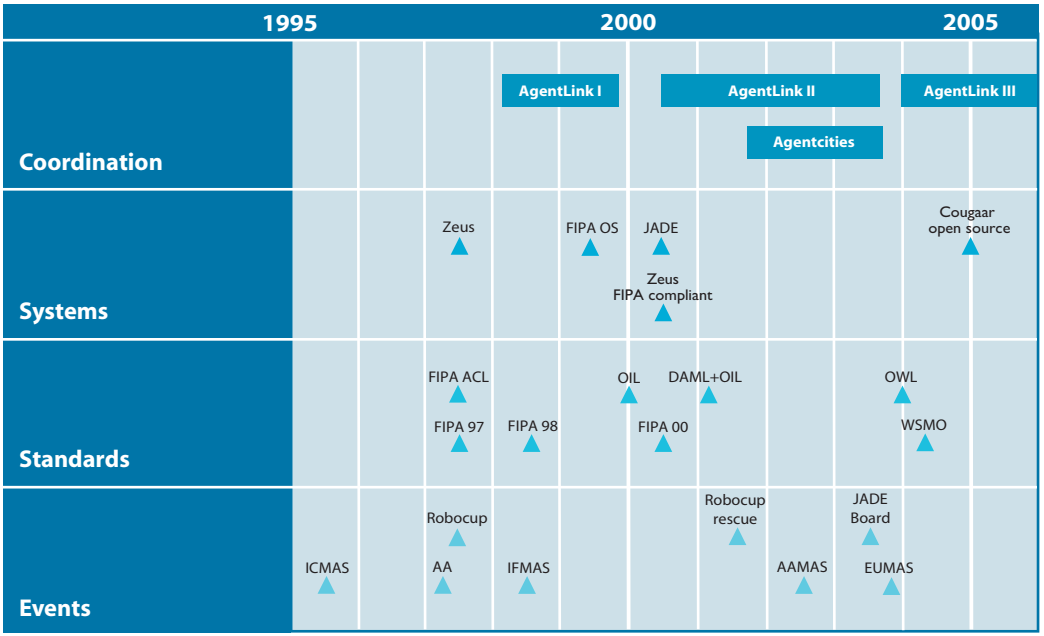


Figure 5.3: European activity in agent-based computing in recent years.

the research level. Usable FIPA standards, for example, were developed in 1998, but matured in 2000; several FIPA compliant agent platforms (JADE, Zeus and FIPA-OS) were also released by 2000. 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 European initiative was launched in 2003 with a European workshop, the European Workshop on Multi-Agent Systems (EUMAS).

6.1 Deliberative Delphi Survey

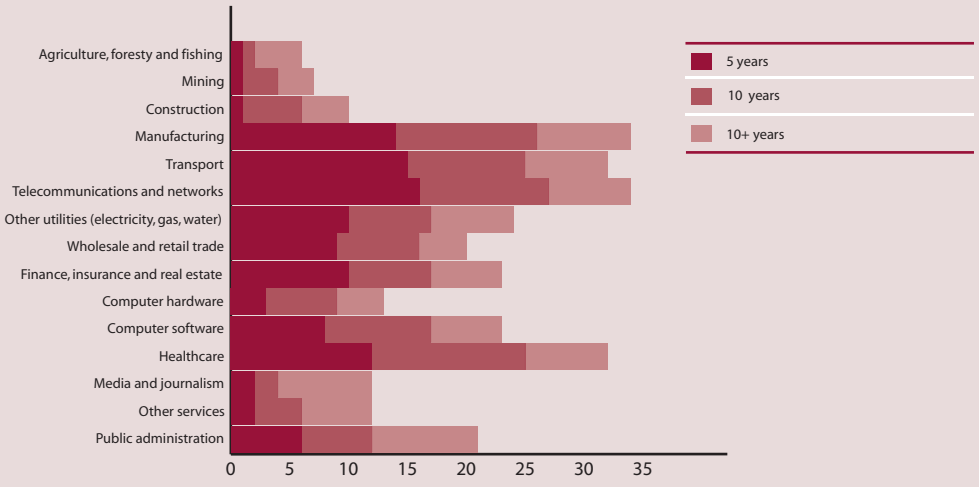
In an effort to elicit an informed assessment of the current state of development of agent technologies and the likely future market penetration for different areas, AgentLink III undertook a Delphi survey of opinion from a selected group of experts in the field. The Delphi method makes use of a limited 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 are theoretical approaches, extrapolation of trends, or more general 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, with 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

Figure 6.1: Manufacturing, transport, telecoms and healthcare will encourage agent deployment

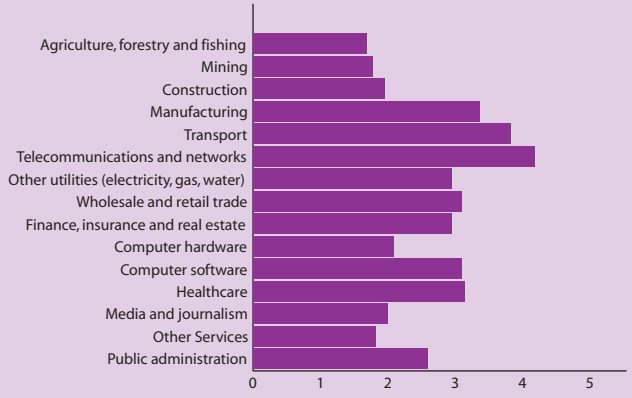


to note that computer software comes relatively low down the list, in this second tier. This contrasts with much work that has focussed on eCommerce and eBusiness systems in recent years, partly because of its relative currency in the light of the Internet boom, and partly because of its ready availability as a domain to study. One question to consider, therefore, is 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. 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 there are already emerging 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 -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

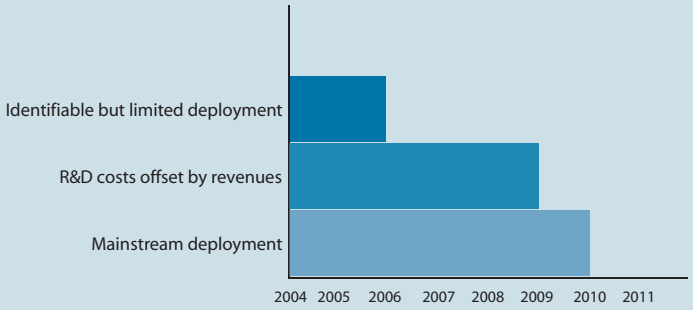


6.1.2 *Deployment of Agent Technologies*

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, with 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, 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.

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



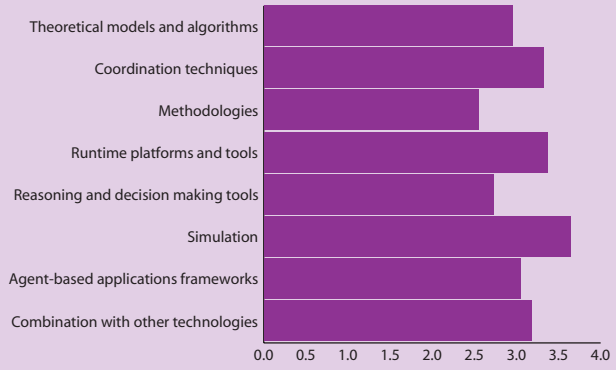
However, one respondent shows some insight by stating that it will be hard to calculate returns, since successful products will not look as though they have any agents. A general problem with software, especially in research and development, is the tendency to focus on the technologies applied rather than 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 the development of advanced reasoning capabilities that are needed not for the majority of systems, but only for complex problem types; until infrastructure is more standardised, however, 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.

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

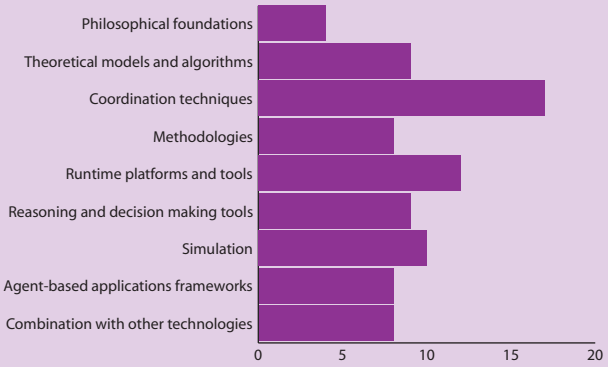


Participants were also asked which technology areas were seen as strong for the application of agent tools, models and solutions, and which were not. The areas exceeding the average in terms of suitability for agent applications corresponded directly to those indicated above as being ready for deployment 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 they reach 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

Figure 6.5: Coordination techniques offer the strongest agent solutions



rating the problem areas on a 1 to 5 scale (with 1 indicating that the area was not suitable, and 5 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 all problem areas showing suitability in the higher range after 10 years.

Figure 6.6: Non agent solutions rate highly for methodologies and runtime platforms

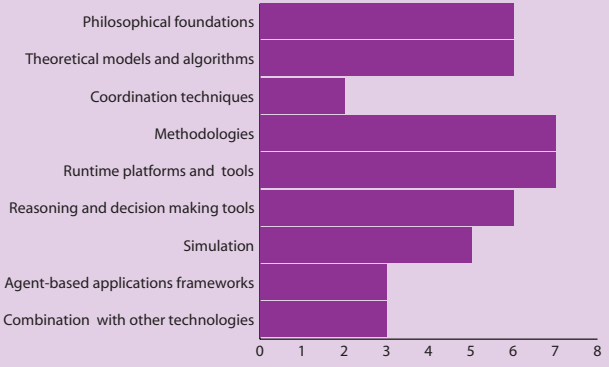
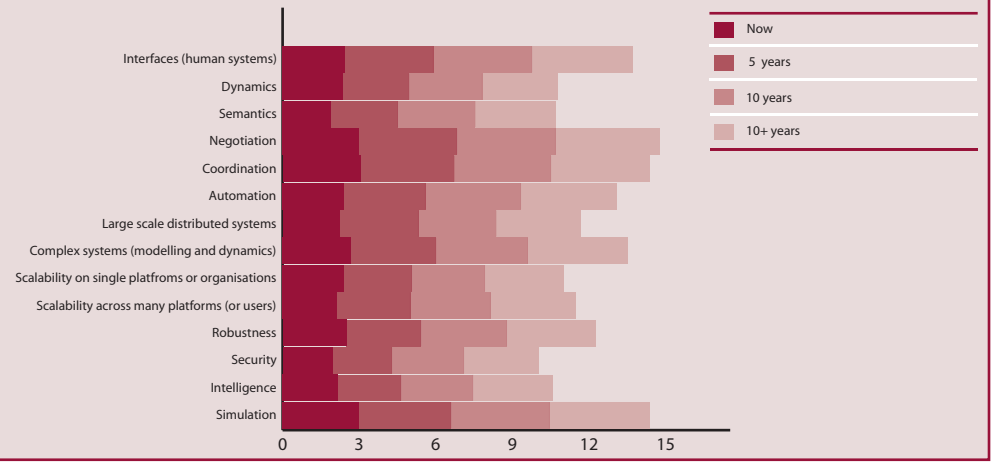


Figure 6.7: Negotiation, coordination, simulation, interfaces and complex systems are suitable for the application of current agent technologies



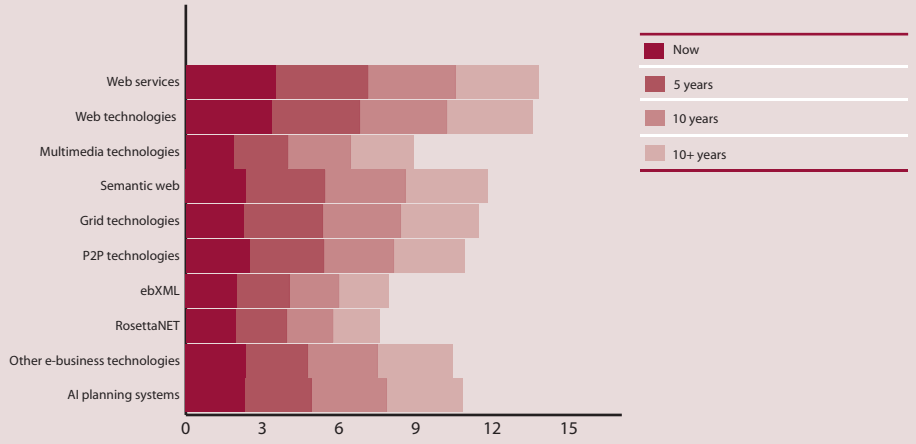
6.1.4 Standards

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 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 question 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 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, as shown in Figure 6.10. The view is consistent with that taken in this document, yet it is 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.

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



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 (7–10 years). 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.

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

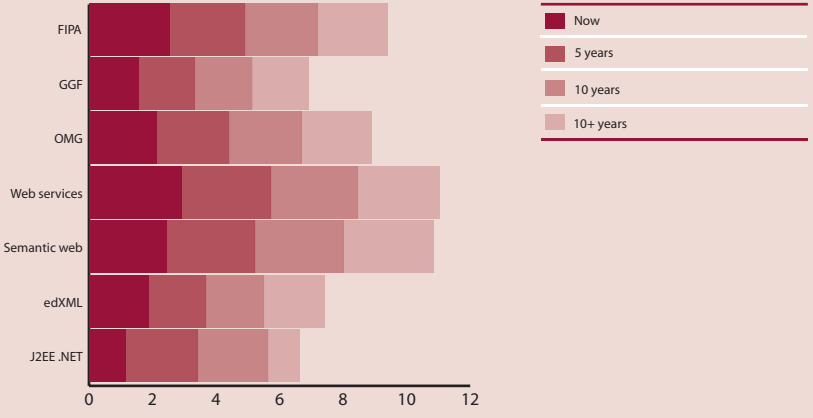
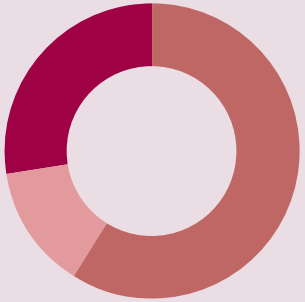


Figure 6.10: Most believe that agent technology will not replace object technology but complement it instead



When will agent technology replace object technology?

2010-2020	27%
Now-2010	14%
Never	59%

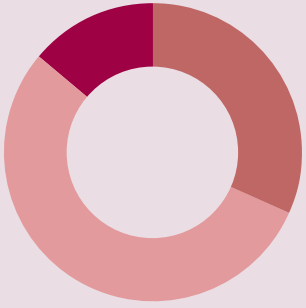
6.2 The Agent Technology Hype Cycle

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

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



What timescale should be adopted for research?

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

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 (Gartner 2004a–2004f) places various agent technologies, agent-related technologies, application domains and drivers at various different points in the hype cycle, as shown in Figure 6.12.

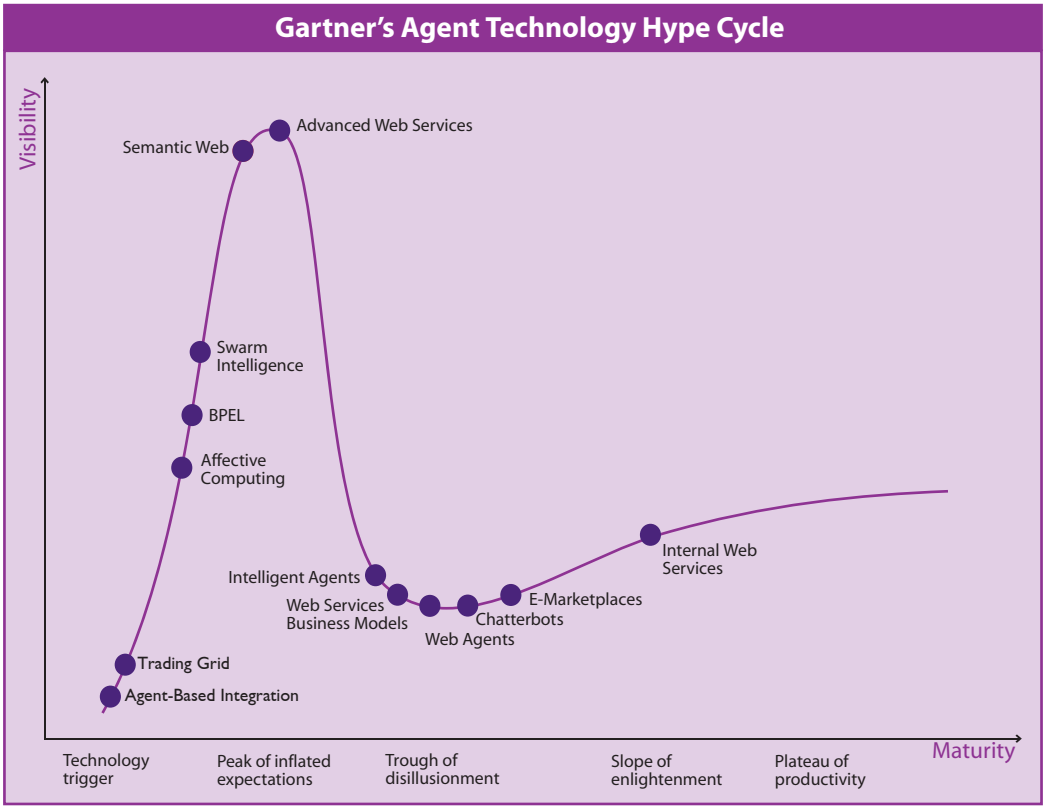


Figure 6.12: The Gartner aggregated agent technology hype cycle

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 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.

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 6.13. 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

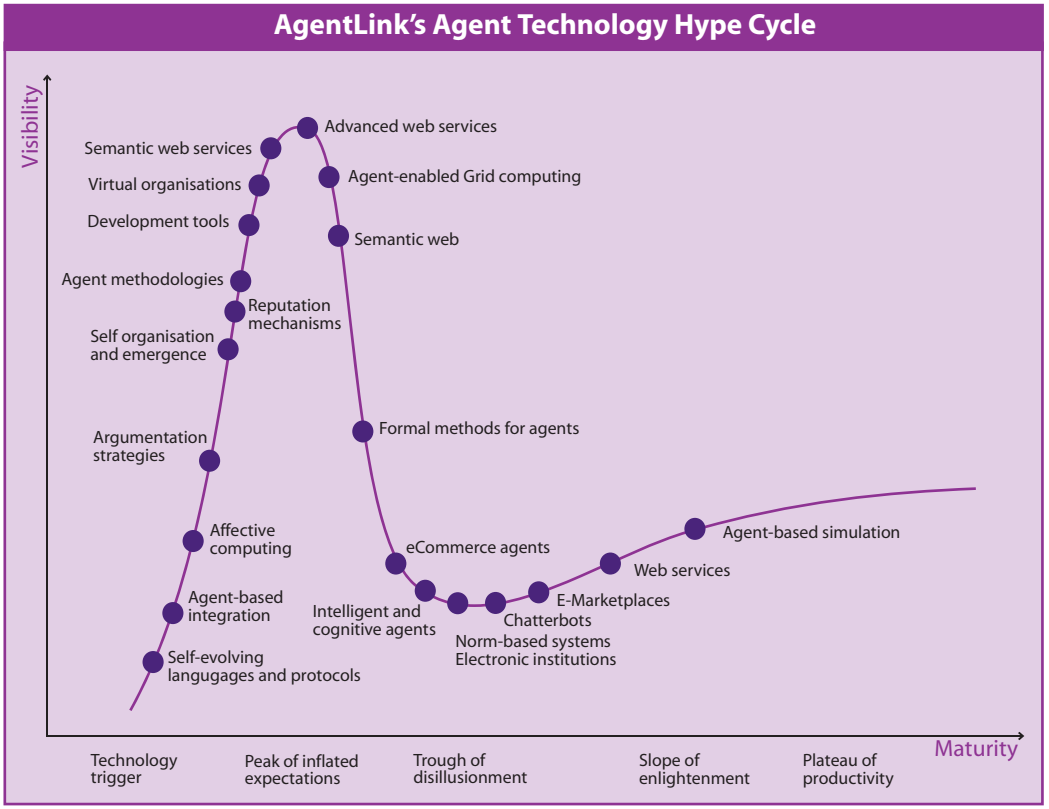


Figure 6.13: The AgentLink agent technology hype cycle

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 self-organisation and emergence (as discussed in detail earlier in this report), methodologies, development tools and virtual

Calico Jack and Healthcare

Calico Jack has been working with the Chief Scientist Office, part of the Scottish Executive Health Department, to develop prototype solutions tackling several key issues in primary care. The company has delivered an agent-based system that integrates with existing email services and in-practice processes, adding new functionality. In particular, and in collaboration with mobile telecoms company, Orange, new services are being offered to patients by SMS and WAP. By modelling the stakeholders in the primary care system as agents, the system has been easily introduced into an already complex mix of IT processes, interpersonal processes, regulatory processes and the relationships between them. In working with patients, GPs and administrators to tailor the service to their needs, agent-based representation has been key in supporting flexibility in design, implementation and deployment. Among the new services currently offered by the system are the ability to coordinate repeat prescriptions using SMS (reducing load on the practice administrator, and simplifying the process for the patient), and to book appointments and handle reminders through a combination of SMS and email (with the aim of reducing the expensive wasteful missed appointments and smoothing the booking process for patients). The system is currently being trialled in a GP practice in Tayside, UK, with a view to subsequent wider rollout.

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 and industrial research. Multi-agent systems are no exception to this, with currently-deployed systems having features found in published research and prototypes of three to five years ago. By looking at current research interests and areas of focus, we are therefore able to extrapolate future trends in deployed systems.

Accordingly, we have identified four broad phases of the future development of 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 not be as advanced as this. 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 prognosis initially undertaken in (Luck et al., 2003).

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 typically 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, as well as development platforms, tend to be ad hoc, inspired by the agent paradigm rather than using principled methodologies, tools or languages. 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. As part of this effort, some platforms now come with their own protocol libraries and force the use of standardised messages, taking one step towards the short-term agenda.

It remains true that, for the foreseeable future, there will be a substantial commercial demand for closed multi-agent systems, for two reasons. First, there are very many problems that can be solved by multi-agent systems without needing to deal with open systems, and this is where many companies are now realising business benefit. Second, in problems involving multiple organisations, agreement among stakeholders on the objectives of the open system may not always be readily achieved, and there may also be security

concerns that arise from consideration of open systems. While progress on technologies for open systems 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 communication languages, such as FIPA ACL, will be used, but interaction protocols will be mixed between standard and non-standard ones. These systems will be able to handle large numbers of agents in pre-determined environments, such as those of Grid applications. Development methodologies, languages and tools 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 suppliers engaged in electronic procurement along a supply-chain, and industry-wide transportation scheduling systems. Even when agents representing multiple organisations participate in these systems, the systems and the associated templates for agent participants will still normally be developed by a dominant company or a consortium on behalf of the entire business network.

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 for agents and systems to master this semantic heterogeneity. Supporting this will be the increased use of new, commonly agreed modelling languages (such as Agent-UML, an extension of UML 2.0), which will promote the use of IDEs and, hopefully, start a harmonisation process as was the case for objects with UML.

Systems will scale to large numbers of participants, although typically only within the domains concerned, and with particular techniques (such as domain-bridging agents),

to translate between separate domains. System development will proceed by standard agent-specific methodologies, including templates and patterns for different types of agents and organisations. Agent-specific programming languages and tools will be increasingly used, making the use of formal verification techniques possible to some extent. Semantic issues related to, for example, coordination between heterogeneous agents, access control and trust, are of particular importance here. Also, because these systems will typically be open, issues such as robustness against malicious or faulty agents, and finding an appropriate trade-off between system adaptability and system predictability, will become increasingly important.

Examples of systems in this phase will be corporate B2B electronic procurement systems permitting participation by any supplier (rather than closed user groups), using agents not conforming to a template.

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 interacting, rather than having to prove adherence before entry. Selection of communications protocols and mechanisms, and of participant strategies, will be undertaken automatically, without human intervention. Similarly, ad hoc coalitions of agents will be formed, managed and dissolved automatically. Although standard communication languages and interaction protocols will have been available for some time, systems in this phase will enable these mechanisms to emerge by evolutionary means from actual participant interactions, rather than being imposed at design time. Of course, such languages, protocols and behaviours may be mere refinements of previously-developed standards, but they will be tailored to their particular contexts of use. In addition, agents will be able to form and re-form dynamic coalitions and virtual organisations on-the-fly and pursue ever-changing goals through appropriate interaction mechanisms for distributed cognition and joint action. In these environments, emergent phenomena will likely appear, with systems having properties (both good and bad) not imagined by the initial design team. Multi-agent systems will be able, adaptable and adept in the face of such dynamic, indeed turbulent, environments, and they will exhibit many of the self-aware characteristics described in the autonomic computing vision. Agents and organisations will be considered as high level system components, easy to customise and train, and which can be combined to provide new components and services, such as in automated or self-assembling software.

By this phase, systems will be fully scalable in the sense that they will not be restricted to arbitrary limits (on agents, users, interaction mechanisms, agent relationships, complexity,

etc). As previously, systems development will proceed by use of rigorous agent-specific design methodologies, in conjunction with programming and verification techniques.

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, which shows the main research and development topics of each area, classified according to the timepoint at which they will attract most attention. Thus, for example, in the area of Industrial Strength Software, peer-to-peer aspects are a short-term focus of attention, while best practice in agent systems design, implementation and verification will likely only be a focus in the long term. In particular, the table suggests 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., 2004a).

By considering the marketing theory of the diffusion of new technologies, together with the features particular to agent technologies, such as standards, and by comparing the historical growth of object technologies and the future growth of agent technologies, we can estimate an adoption curve for object technologies. Such a curve, shown in Figure 7.2, indicates the total proportion of adopters in a population at each moment of time, and is the cumulative version of a product life-cycle presented earlier. Marketers commonly use an exponential function to model new product diffusion, as we have done, based on (McBurney et al., 2002).

In the case of object and agent technologies, the relevant population comprises all organisations and companies engaged in software development, either internally or via commissioned projects. To calibrate the adoption curve, we have assumed that, in the long-run, 75% of all such organisations will adopt object-oriented programming (OOP) techniques. Using qualitative information about the growth in interest in OOP (from the "Agents versus Objects" box on page 48) we have estimated the rate of growth of the curve, where the market grows increasingly rapidly until late 1997, after which the rate of growth in adoption slows down.

To calibrate the model for agent technologies, we have assumed the same curve but starting later (1985, rather than 1962), and with a smaller long-run potential. Because agent technologies are appropriate for fewer application domains than are object technologies, it is assumed that only 35% of the population of organisations or companies engaged in software development will ever adopt agent technologies.

Will agent technologies be adopted faster than were object technologies? On the one hand, competitive pressures and the faster pace of technology change now experienced suggest that agent technologies will be adopted sooner than object technologies. On the other hand, the greater complexity of typical agent applications, and the fact that many applications require inter-organisational collaboration, suggest a slower rate of adoption than for object technologies. Putting together these countervailing forces, we are led to propose the same growth rate as for object technologies. The resulting adoption curve is also shown in Figure 7.2, and, as can be seen there, the rate of growth of adoption increases until mid 2014, after which it slows down.

This adoption curve for agent technologies is consistent with the findings of the previously described Deliberative Delphi study. For instance, Figure 6.3 indicates that, on average, Delphi respondents expect mainstream deployment of agent technologies only from

	Short Term	Medium Term	Long Term
Industrial Strength software	Peer to peer Better development tools Agent UML Service oriented computing	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 Service oriented computing Semantic description	Flexible business/trading languages Libraries of interaction protocols	Tools for evolutions of communications languages and protocols
Infrastructure for Open Communities	Web mining Data integration and Semantic Web Metadata	Semantic interaction Agent-enabled semantic web (services) Electronic institutions Dynamic norms, roles, laws, organisations	Shared, improved ontologies
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

2010. The curve in Figure 7.2 indicates a penetration level for agent technologies of 12% of organisations engaged in software development by 2010, or about one-third of the long-run adoption level of 35%. At this level of penetration, it is reasonable to assume that applications of agent technologies have become mainstream. However, not all applications of agent technologies may be labelled as such, as for example, with trust and reputation systems, automated auction bidding systems, or Grid systems. All of these applications may use agent technologies without being called agent systems.

A similar rate of growth to that for object-oriented technologies can only be acheived if the obstacles currently in the way of adoption of agent technologies are overcome, as indeed they were for object technologies. Thus, for example, issues of standards and the provision of software development methodologies and tools are important to be resolved if we are to move beyond the current early adopter stage of market diffusion.

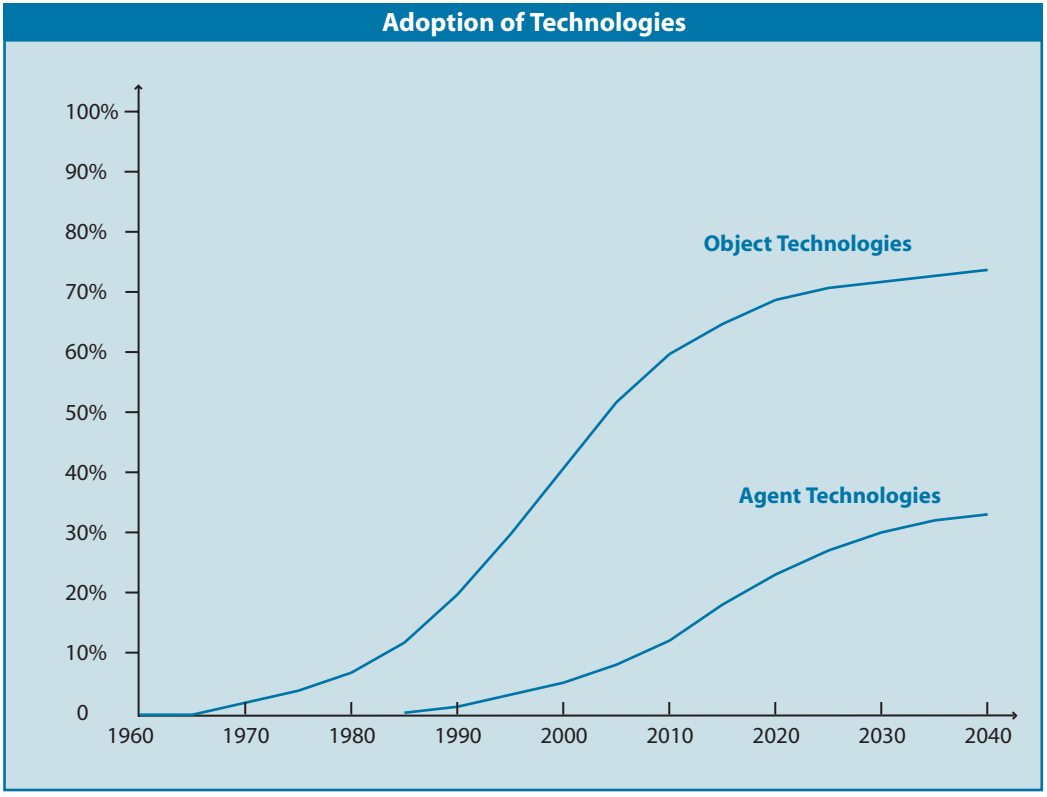


Figure 7.2: Projected penetration levels for object technologies and agent technologies

8 Challenges

Hardware and software have improved significantly in performance and availability over the six decades of modern computing. As these changes have occurred, the objectives of programmers have also changed. Initially, most programmers sought to minimise memory usage and to maximise throughput or processing speeds in their applications. With increasing availability and lower costs of memory, and increasing micro-processor speeds, these objectives became far less important. Instead, by the 1970s and 1980s, the object-oriented paradigm sought to maximise the modularity and re-usability of code, and to minimise post-deployment system maintenance. However, these objectives too have become dated. Partly, this is because the development of proven OOP methods and support tools have enabled the objectives to be readily achieved, and indeed, taken for granted, over the last two decades. More importantly, however, the rise to prominence of the Internet has led to a new understanding of the nature of computation, an understanding which puts interaction at its centre. In this context, the agent-oriented paradigm has sought to maximise adaptability and robustness of systems in open environments.

It is here that one can see how a new technology may be a disruptive force. By tackling a different set of objectives, agent technologies address different problems and different applications than do object technologies. It is not simply that the rules of the game have changed, but rather that a different game is being played. In a world of millions of independent processors interconnected via the Internet and, through it, engaged in distributed cognition, a software design team can no longer assume that software components will share the same goals or motivations, or that the system objectives will remain static over time. Systems therefore need to be able to adapt to dynamic environments, to be able to configure, manage and maintain themselves, and to cope with malicious, whimsical or just plain buggy components. The power of the agent paradigm is that it provides the means, at the appropriate level of abstraction, to conceive, design and manage such systems.

8.1 Broad Challenges

Each of the compelling visions discussed in the context of trends and drivers above — the Semantic Web, ambient intelligence, the Grid, autonomic systems — will require agent technologies, or something very like them, before being realised: agent technologies are upstream of these visions and mission-critical to them. For agent-based computing to support these visions, considerable challenges remain, both broad, over-arching challenges across the entire domain of agent technologies, and challenges specific to particular aspects. The broad challenges are as follows.

- Creating tools, techniques and methodologies to support agent systems developers. Compared to more mature technologies such as object-oriented programming, agent developers lack sophisticated software tools, techniques and methodologies to support the specification, development and management of agent systems.
- Automating the specification, development and management of agent systems. Agent systems and many of their features are still mostly hand-crafted. For example, the design of auction mechanisms awaits automation, as does the creation and management of agent coalitions and virtual organisations. These challenges are probably several decades from achievement, and will draw on domain-specific expertise (for example, economics, social psychology and artificial intelligence).
- Integrating components and features. As is evident from Sections 2 and 4 above, many different theories, technologies and infrastructures are required to specify, design, implement and manage agent systems. Integrating these pieces coherently and cost-effectively is usually a major undertaking in any system development activity, a task made more challenging by the absence of mature integration tools and methodologies.
- Establishing appropriate trade-offs between adaptability and predictability. Creating systems able to adapt themselves to changing environments, and to cope with autonomous components, may well lead to systems exhibiting properties that were not predicted or desired. Striking a balance, appropriate to the specific application domain, between adaptability and predictability is a major challenge, as yet unresolved either theoretically or practically. Associated with predictability is the requirement for practical methods and tools for verification of system properties, particularly in multi-agent systems that are likely to exhibit emergent behaviour.
- Establishing appropriate linkage with other branches of computer science and with other disciplines, such as economics, sociology and biology. One task here is to draw appropriately on prior research from these other areas and disciplines. Another task is to avoid reinvention of existing techniques and methods, whether by agent researchers or by others. Awareness-building between areas and disciplines, and coordination of research and development activities, are essential if the appropriate linkages are to established and maintained.

8.2 Specific Challenges

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 6.) Nevertheless, in addition to the broad challenges,

there are challenges specific to different aspects and features of agent systems (Bullock and Cliff, 2004; Foster et al., 2004).

Trust and reputation

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 earlier, we need new techniques for expressing and reasoning about trust and reputation, on both an individual and a social level to enable interaction in dynamic and open environments.

Virtual organisation formation and management

Virtual organisations (VOs) have been identified as one of the key contributions of Grid computing, but principled and well-defined procedures for determining when to form new VOs, how to manage VOs and portfolios of VOs, how to manage competing and

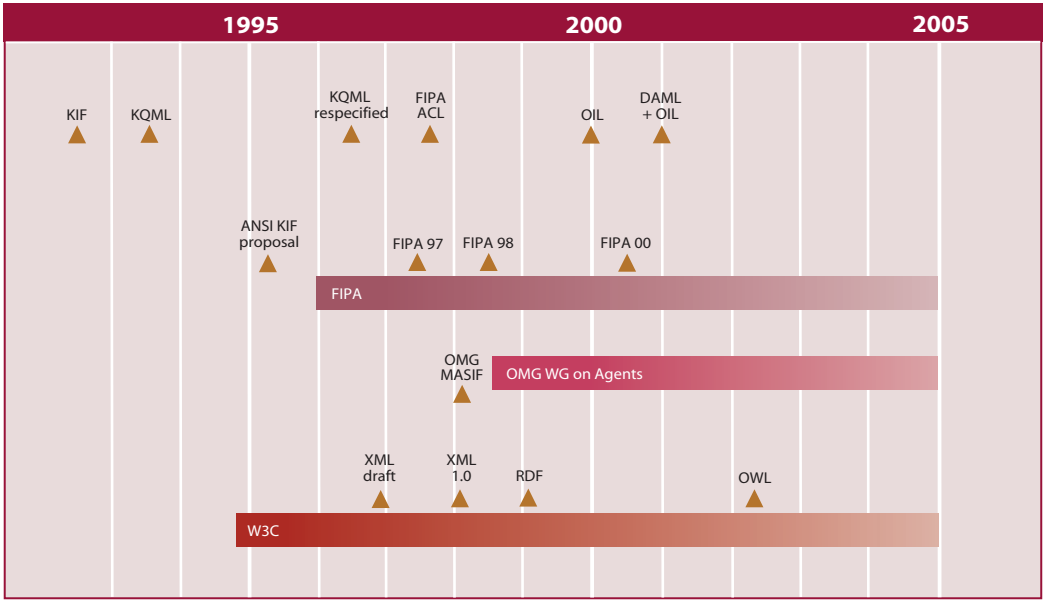


Figure 8.1: Standards activity in the area of agent-based computing

complementary VOs, and ultimately how and when to disband them, are still missing. Moreover, the development of procedures and methods for the automation of VO creation, management and dissolution also provide major research and development challenges. In addition, once such procedures have been defined, creating formal representations of them to support their automated deployment by agents themselves at runtime will be a major research challenge.

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 stakeholders, and different definitions of individual and social welfare. 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, in particular for multi-objective coordination and negotiation. In addition, as with VOs, the automation of the design, implementation and management of mechanisms is a major challenge.

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 under what circumstances. 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. Incorporating capabilities for disagreement and justifications (i.e. arguments) in negotiations is also a major research challenge.

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 they are equivalent, and what each approach provides to software engineering or system control.

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.

Whitestein Technologies and Adaptive Transportation

The Living Systems® Adaptive Transportation Networks (LS/ATN) application is a cost-based optimisation system for transport logistics. Developed by Whitestein Technologies, originally for DHL, LS/ATN is designed to provide automatic optimisation for large-scale transport companies, taking into account the many constraints on their vehicle fleet, cargo, and drivers. Although the agent solution accounts for only 20% of the entire system, agent technology plays a central role in the optimisation. Vehicle drivers send information specifying their location and proposed route, and the system determines if that vehicle can collect an additional load, or swap future loads with another vehicle in order to reduce cost. A negotiation is performed automatically by agents, with each agent representing one vehicle, using an auction-like protocol. The vehicle that can provide the cheapest delivery, wins the auction, reducing the overall cost of cargo delivery and in most cases, the combined distance travelled for all vehicles. The aim is to find a local optimum (that is, not European-wide), so that only vehicles travelling in close proximity to each other will be involved in negotiations.

Moreover, issues such as what is meant by learning in a multi-agent context and what constitutes "good" learning are also important.

Methodologies

Many of today's challenges in software design stem from the distributed, multi-actor nature of new software systems and the resulting change in objectives implied for software engineering. The development of methodologies for the design and management of multi-agent systems seeks to address these problems by extending current software engineering techniques to explicitly address the autonomous nature of their components and the need for system adaptability and robustness. 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 achieved 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 and composition

There is a need for integrated service architectures providing robust foundations for autonomous behaviour, in order to support dynamic services, and important negotiation, monitoring, and management patterns. This will aid application and deployment of agent technologies to the Grid and other domains. 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. This is likely to take the form of agent-oriented architectures based on peer-to-peer or other novel structures.

Semantic integration

In open systems, different entities will have distinct information models, demanding that techniques are developed for bridging the semantic gaps between them. Advances are required in such areas as ontology definition, schema mediation, and semantic mediation. The challenge here is to develop flexible models for semantic capture and integration.

8.3 Recommendations

The different challenges outlined above give rise to several distinct recommendations that can be made in relation to the development of the field of agent-based computing. These recommendations are intended to highlight the needs of the field from a technological standpoint, in order to support the realisation of the vision of future computing systems as described throughout this roadmap. They build on the recommendations provided previously in (Luck et al., 2003), which provide a complementary view of the important challenges facing the field.

1. Create tools, techniques and methodologies to support agent systems developers.
2. Automate the specification, development and management of agent systems and of key components, such as protocols and virtual organisations (VOs).
3. Integrate agent components and features to enable the different theories, technologies and infrastructures to come together coherently.
4. Establish appropriate trade-offs between adaptability and predictability so that agents can exhibit behaviour, emergent or otherwise, that can be supported by tools and property verification.
5. Establish and enhance appropriate linkages with other branches of computer science and with other disciplines, such as economics, sociology and biology, to draw on prior research and avoid reinvention of existing techniques and methods.
6. Develop techniques for expressing and reasoning about trust and reputation, on both an individual and a social level to enable interaction in dynamic and open environments
7. Develop procedures and methods for the automation of virtual organisation creation, management and dissolution, together with appropriate formal representations to support their automated deployment.
8. Develop mechanisms and protocols for automated interactions, in particular for multi-objective coordination and negotiation, as well as techniques for their automated design, implementation and management.

9. Provide negotiation algorithms and protocols, including capabilities for disagreement and reasoned justification, and determine which are most effective under different circumstances.

10. Establish the relevance of, and techniques for, the use of complex, adaptive systems in the design, implementation and control of large-scale multi-agent systems, drawing on approaches from physics, biology and other related fields.

11. Develop a range of new techniques for learning and optimisation in dynamic and unstable multi-agent environments, together with evaluation methods.

12. Integrate techniques from the range of existing software development methodologies, for use with autonomous agents in open environments, while addressing new developments in the field.

13. Develop provenance mechanisms and infrastructure to trace results and build up trusted networks by identifying individual agents and aggregated services in a scalable, secure, open and interoperable fashion.

14. Develop integrated service and agent architectures for dynamic services, negotiation, monitoring, and management of autonomous adaptable organisations.

15. Develop flexible models for semantic information capture and integration in support of interoperability.

9 Conclusions

As just seen, agent technologies can be distinguished from other programming technologies on the basis of their differing objectives. For agent technologies, the objectives are to create systems situated in dynamic and open environments, able to adapt to these environments and capable of incorporating autonomous and self-interested components. How quickly agent technology is adopted by software developers, therefore, will depend at least partly on how many application domains require systems with these characteristics. Considering the domains receiving attention from agent software development companies such as Agentis, Magenta, Lost Wax or Whitestein (among others), the main areas are currently: logistics, transportation, utility management and defence. Common to many of these domains are multiple stakeholders or organisations linked in a network, such as a supply-chain, and with mission-critical, real-time processing requirements. In other words, there are both functional and technical requirements for these applications, a divide that agent technologies are able to bridge.

Most new software technologies require supporting tools and methodologies. A fundamental obstacle to the take-up of agent technology is the current 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. A key 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 development needs 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 agent-like systems (in the 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 comprising pre-existing agents that must be organised or orchestrated. However, making the transition from research laboratory to deployed industrial applications is indeed a challenge, and it will be important to make scientifically sound business cases for implementations and descriptions that work as stimulators both for industry adoption and for further research.

For commercial and industrial systems, agent technologies must emerge from the laboratory with a focus on business issues, on quality and on convergence with existing and emerging industrial technologies rather than innovation. Here, safety, reliability and traditional software quality measures are equally important, and must all be addressed to achieve wider adoption. In particular, we need agent solutions for distributed, enterprise-wide environments with exacting development requirements. This might be achieved through transition approaches by which existing systems can be upgraded with a successively increased agent presence in a seamless fashion. Wrapping legacy systems within autonomous agents situated in a larger multi-agent system is one approach that is being tried, for example, in connecting new and old telecommunications switches together seamlessly, allowing legacy switches to be gradually replaced without major disruption to the overall system.

More generally, the adoption of agent technologies in business environments depends on how fast and how well agent technologies can be linked to existing and proven software and software methods. Agent technologies should be targeted at those application domains 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

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

R. H. Bordini, M. Dastani, J. Dix, A. El Fallah Seghrouchni (Eds.), *Multi-Agent Programming: Languages, Platforms and Applications*, Springer, 2005.

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/

J. Botía, A. López-Acosta and A. Gómez-Skarmeta, ACLAnalyser: A Tool for Debugging Multi-Agent Systems. In Proceedings of the Sixteenth European Conference on Artificial Intelligence, 967–968, 2004.

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. http://www.foresight.gov.uk/Intelligent_Infrastructure_Systems/Emergent_Behaviour.pdf

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, 2006.

O.-J. Dahl, The roots of object orientation: the Simula language, in M. Broy and E. Denert (Eds.), *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 A. Omicini, P. Petta, and J. Pitt, editors, Engineering Societies in the Agents World IV, Lecture Notes in Artificial Intelligence 3071, 93–106, 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, 8–15, ACM Press, 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*, 548–553, Morgan Kaufmann Publishers, 1999.

Gartner, Hype Cycle for Application Integration and Platform Middleware, 2004a.

Gartner, Hype Cycle for Application Development, 2004b.

Gartner, Hype Cycle for Human-Computer Interaction, 2004c.

Gartner, Hype Cycle for B2B CRM Technologies, 2004d.

Gartner, Hype Cycle for the Knowledge Workplace, 2004e.

Gartner, Hype Cycle for Supply Chain Management, 2004f.

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.

IDC: *Worldwide Mobile Phone 2005-2009 Forecast & Analysis Report* (Report 33290), May 2005.

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

Interagency Working Group on Information Technology Research and Development, *Grand Challenges: Science, Engineering, and Societal Advances Requiring Networking and Information Technology Research and Development*, National Coordination Office for Information Technology Research and Development, USA, 2003. http://www.nitrd.gov/pubs/200311_grand_challenges.pdf

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.

A. Kuflik, Computers in control: rational transfer of authority or irresponsible abdication of autonomy, *Ethics and Information Technology*, 1(3): 173–184, 1999.

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, 2004a.

M. Luck, R. Ashri and M. d'Inverno, *Agent-Based Software Development*, Artech House, 2004b.

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

P. McBurney, S. Parsons and J. Green, Forecasting market demand for new telecommunications services: an introduction, *Telematics and Informatics*, 19(3): 225–249, 2002.

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*, 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.

C. Perez, *Technological Revolutions and Financial Capital: The Dynamics of Bubbles and Golden Ages*, Edward Elgar, 2002.

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.

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*, 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, Physica, 2003.

F. Zambonelli and H. V. Parunak, Signs of a revolution in computer science and software engineering, in P. Petta, R. Tolksdorf and F. Zambonelli (Eds.), *Engineering Societies for the Agents World*, Lecture Notes in Artificial Intelligence 2577, 13–28, Springer, 2002.

Glossary

ANSI	American National Standards Institute
B2B	Business to business
BDI	Belief-Desire-Intention (typically of agent architectures)
Bluetooth	Short range wireless connectivity standard
CASE	Computer Aided Software Engineering
CERN	European Organisation for Nuclear Research
CORBA	Common Object Request Broker Architecture
ebXML	Electronic Business using eXtensible Markup Language
FIPA	Foundation for Intelligent Physical Agents
GGF	Global Grid Forum
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
JADE	Java Agent DEvelopment Framework
Jini	Open architecture enabling adaptive network-centric services
JXTA	Open protocols allowing devices to communicate in a P2P manner
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
OOPSLA	Object-Oriented Programming, Systems, Langauges and Applications
OODBS	Object-Oriented Database Systems
P2P	Peer-to-Peer
RDF	Resource Description Format
RosettaNet	Industry consortium developing standards for collaborative commerce
SOA	Service-oriented architecture
SOAP	Simple Object Access Protocol
TCP	Transmission Control Protocol
UDDI	Universal Description, Discovery and Integration
UDP	User Datagram Protocol
UML	Unified Modelling Language
UPnP	Universal Plug and Play
WSDL	Web Service Description Language
WS-CDL	Web Services Choreography Description Language
WS-R	Web Services — Reliability
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

Web Resources and URLs

AgentLink	www.agentlink.org
Autonomic Computing	www.ibm.com/autonomic
Bluetooth	www.bluetooth.com
CORBA	www.corba.org
ebXML	www.ebxml.org
European Commission	www.cordis.lu
Foundation for Intellient Physical Agents	www.fipa.org
Global Computing	www.cordis.lu/ist/fet/gc.htm
Global Grid Forum	www.ggf.org
Information Society Technologies	www.cordis.lu/ist
Internet Engineering Task Force	www.ietf.org
JADE	jade.tilab.com
Jini	www.jini.org
JXTA	www.jxta.org
N1	www.sun.com/n1
OASIS	www.oasis-open.org
Object Management Group	www.omg.org
RosettaNet	www.rosettanet.org
UDDI	www.uddi.org
UML	www.uml.org
UPnP	www.upnp.org
World Wide Web Consortium	www.w3c.org
XML	www.xml.org

Companies Mentioned

Acklin	www.acklin.nl
Agentis Software Inc	www.agentissoftware.com
Calico Jack	www.calicojack.
Magenta Technology	www.magenta-technology.com
Eurobios	www.eurobios.com
Lost Wax	www.lostwax.com
Nutech Solutions	www.nutechsolutions.com
Whitestein Technologies	www.whitestein.com

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 Steven Willmott joined the core team, which aimed to 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 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.

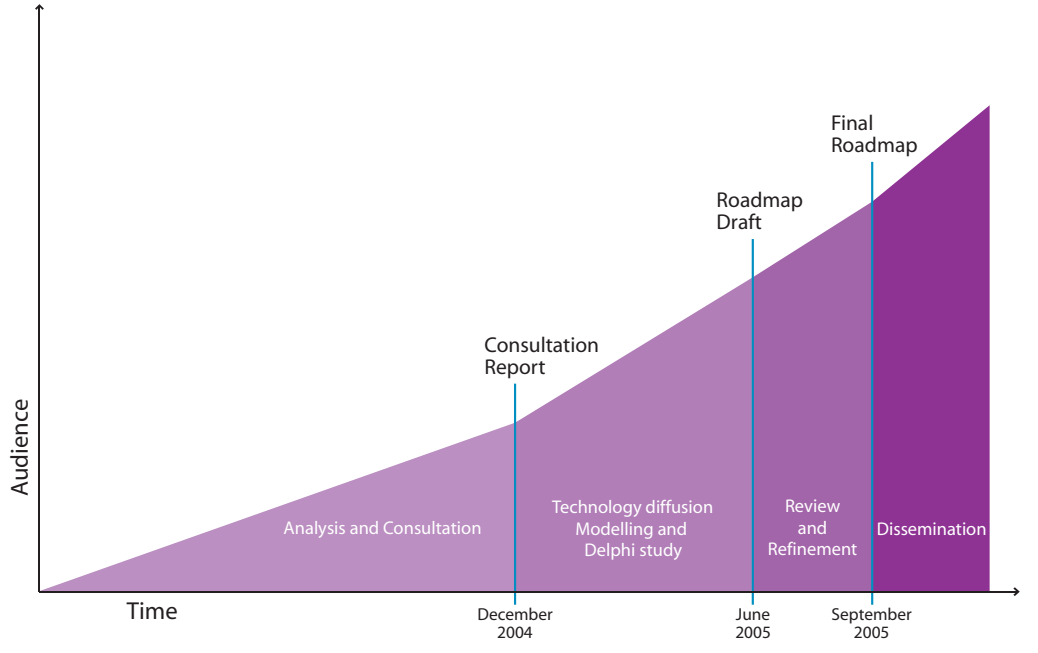


Figure A.1: Stages of roadmap development

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 were 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

[As of September 2005]

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
Czech Technical University	Czech Republic
CertiCon AS	Czech Republic
NeuroAgent Ltd	Finland
UTBM	France
University Paris Dauphine	France
Institut de Recherche en Infomatique de Toulouse	France
Team MAIA	France
France Telecom SA	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	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 Intelligenz	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
University of Applied Sciences	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
Technical University of Crete	Greece
Hungarian Academy of Sciences	Hungary
AITIA Inc.	Hungary
University College Dublin	Ireland
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
Università degli Studi di L'Aquila	Italy
Università Politecnica delle Marche	Italy
University of Bari	Italy
University of Cagliari	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
University of Catania	Italy
Institute of Computer Science, Polish Academy of Sciences	Poland
Institute of Comuter Science, Jagiellonian University	Poland
University of Warsaw	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
Babes-Bolyai University	Romania
St Petersburg Institute For Infomatics and Automation	Russia
Moscow Institute of Physics and Technology (MIPT)	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
Institut d'Investigació en Intel.ligència Artificial	Spain
Universidad de Murcia,	Spain
Technical University of Madrid	Spain
Universidad Politécnica de Valencia	Spain
Universitat Rovira I Virgili	Spain
Agents Inspired Technologies SA	Spain
University of Vigo	Spain
Universitat Autonoma de Barcelona	Spain
University of Zaragoza	Spain
MicroArt	Spain
University of Barcelona	Spain
Semantic Systems, SA	Spain
Stockholm University	Sweden
Swedish Institute of Computer Science	Sweden
Blekinge Institute of Technology	Sweden
Orebro University	Sweden
Royal Institute of Technology	Sweden
Whitestein Technologies AG	Switzerland
University of Geneva	Switzerland
Savannah Simulations	Switzerland
Acklin BV	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
Y'All	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
Vodafone Group R&D	United Kingdom
Aumega Networks	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 Joint Conference on Autonomous Agents and Multi-Agent Systems, 2004 and 2005; European Conference on Artificial Intelligence, 2004; European Workshop on Multi-Agent Systems, 2004; IEEE Systems, Man and Cybernetics Conference, 2004; IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology, 2005.

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 Multi-Agent Systems
Self-Organisation in Multi-Agent Systems
Towards Semantic Web Agents
Trust for Open Collaborative Agent Business Environments
Coordinating Agent Standardisation Activities
Towards a Standard Agent-to-Agent Argumentation Interchange Format.

AgentLink Roadmap Development

Core Team

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

AgentLink Support

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

AgentLink 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
Steven Willmott	Universitat Politècnica de Catalunya, Spain
Mike Wooldridge	University of Liverpool, UK

Other Contributors

Chris van Aart	Y'All, The Netherlands
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
Ed Durfee	University of Michigan, USA
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
Nick Jennings	University of Southampton, UK
Menno Jonkers	Tryllian, The Netherlands
Anthony Karageorgos	University of Thessaly, Greece
David Kinny	Agentis Software, USA
Stefan Kirn	University of Hohenheim, Germany
Magdalena Koralewska	Jagiellonian University of Krakow, Poland
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
Paul Marrow	BT Pervasive ICT Research Centre, UK
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	TNO & DECIS, The Netherlands
Ambra Molesini	Università degli Studi di Bologna, Italy
Luc Moreau	University of Southampton, UK
Steve Munroe	University of Southampton, UK
Pablo Noriega	Institut d'Investigació en Intel.ligència Artificial, Spain
Tim Norman	University of Aberdeen, UK
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
Simon Parsons	City University of New York, USA

Juan Pavon	Universidad Complutense, Spain
Carlota Perez	University of Cambridge and University of Sussex, UK
Gauthier Picard	Université Paul Sabatier, France
Eric Platon	University of Tokyo, Japan
Agostino Poggi	Università degli Studi di Parma, Italy
Chris Preist	HP Laboratories, UK
Chris Reed	Calico Jack, UK
Juan A. Rodriguez	Institut d'Investigació en Intel.ligència Artificial, 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	Institut d'Investigació en Intel.ligència Artificial, Spain
Munindar Singh	North Carolina State University, USA
Liz Sonenberg	University of Melbourne, Australia
Paulo Sousa	Instituto Superior de Engenharia do Porto, Portugal
James Spillings	General Dynamics, UK
Rebecca Steliaros	Engineering and Physical Sciences Research Council, 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
Michael Wooldridge	University of Liverpool, UK
Nadezhda Yakounina	Magenta Technology, UK
Makoto Yokoo	University of Kyushu, Japan

Contact Information

Published by the University of Southampton on behalf of AgentLink III.

Contact: Professor Michael Luck
School of Electronics and Computer Science
University of Southampton
Southampton SO17 1BJ
United Kingdom

AgentLink III is a Coordination Action funded by the European Commission in Semantic-Based Knowledge Systems (Project Reference: IST-FP6-002006CA)

Designed and typeset by Serena Raffin.

ISBN 085432 845 9