

Chapter 5

The X-Internet Platform

X-Internet enabled systems can be built today using the technologies that are already in place. This chapter first examines the constructs of the Internet's architecture so that we can ascertain its strengths and weaknesses. It then discusses the architectural concerns — and possibilities — of building X systems.

The Internet — A Network of Networks

The Internet is not a monolithic, uniform network; rather, it is a network of networks, owned and operated by different companies, including Internet backbone providers. Internet backbones deliver data traffic to and from their customers; often, this traffic comes from, or travels to, customers of another backbone (Kende, 2000).

End users communicate with each other using the Internet, and also access information or purchase products or services from content providers, such as the *Wall Street Journal Interactive Edition*, or E-commerce vendors, such as Amazon.com. End users access the Internet via Internet service providers (ISPs) such as America Online (AOL) or MindSpring Enterprises. Small business and residential end users generally use modems, cable-modems, and DSL to connect to their ISPs, while larger businesses and content providers generally have dedicated access to their ISPs over leased lines.

Content providers use a dedicated connection to the Internet that offers end users 24-hour access to their content. ISPs are generally connected to other ISPs through Internet backbone providers such as UUNET and

PSINet. Backbones own or lease national or international high-speed fiber-optic networks that are connected by routers, which the backbones use to deliver traffic to and from their customers. Many backbones also are vertically integrated, functioning as ISPs by selling Internet access directly to end users, as well as having ISPs as customers.

Each backbone provider essentially forms its own network that enables all connected end users and content providers to communicate with one another. End users, however, are generally not interested in communicating just with end users and content providers connected to the same backbone provider; rather, they want the ability to communicate with a wide variety of end users and content providers, regardless of backbone provider.

To provide end users with such universal connectivity, backbones must interconnect with one another to exchange traffic destined for each other's end users. It is this interconnection that makes the Internet the "network of networks" that it is today. As a result of widespread interconnection, end users currently have an implicit expectation of universal connectivity whenever they log on to the Internet, regardless of which ISP they choose. ISPs are therefore in the business of selling access to the entire Internet to their end-user customers. ISPs purchase this universal access from Internet backbones. The driving force behind the need for these firms to deliver access to the whole Internet to customers is what is known in the economics literature as *network externalities*.

Network externalities arise when the value, or utility, that a consumer derives from a product or service increases as a function of the number of other consumers of the same or compatible products or services. They are called network externalities because they generally arise for networks whose purpose it is to enable each user to communicate with other users; as a result, by definition, the more users there are, the more valuable the network.

These benefits are externalities because a user, when deciding whether to join a network (or which network to join), only takes into account the private benefits that the network will bring her, and will not consider the fact that her joining this network increases the benefit of the network for other users. This latter effect is an *externality*.

Network externalities can be direct or indirect. They are direct for networks that consumers use to communicate with one another; the more consumers who use the network, the more valuable the network is for each consumer. The phone system is a classic example of a system providing direct network externalities. The only benefit of such a system comes from access to the network of users.

Network externalities are indirect for systems that require both hardware and software to provide benefits. As more consumers buy hardware,

this will lead to the production of more software compatible with this hardware, making the hardware more valuable to users. A classic example of this is the compact disc system; as more consumers purchased compact disc players, music companies increased the variety of compact discs available, making the players more valuable to their owners. These network externalities are indirect because consumers do not purchase the systems to communicate directly with others, yet they benefit indirectly from the adoption decision of other consumers.

One unique characteristic of the Internet is that it offers *both* direct and indirect network externalities. Users of applications such as e-mail and Internet telephony derive direct network externalities from the system: the more Internet users there are, the more valuable the Internet is for such communications. Users of applications such as the World Wide Web derive indirect network externalities from the system: the more Internet users there are, the more Web content will be developed, which makes the Internet even more valuable for its users. The ability to provide direct and indirect network externalities to customers provides an almost overpowering incentive for Internet backbones to cooperate with one another by interconnecting their networks.

Peering and Transit

During the early development of the Internet, there was only one backbone, and therefore interconnection between backbones was not an issue. In 1986, the National Science Foundation (NSF) funded the NSFNET, a 56-kilobit-per-second (Kbps) network created to enable long-distance access to five supercomputer centers across the country. In 1987, a partnership of Merit Network, Inc., IBM, and MCI began to manage the NSFNET, which became a T-1 network connecting 13 sites in 1988.

The issue of interconnection arose only when a number of commercial backbones came into being, and eventually supplanted the NSFNET. At the time that commercial networks began appearing, general commercial activity on the NSFNET was prohibited by an Acceptable Use Policy, thereby preventing these commercial networks from exchanging traffic with one another using the NSFNET as the backbone. This roadblock was circumvented in 1991, when a number of commercial backbone operators, including PSINet, UUNET, and CerfNET, established the Commercial Internet Exchange (CIX). CIX consisted of a router, housed in Santa Clara, California, that was set up for the purpose of interconnecting these commercial backbones and enabling them to exchange their end users' traffic. In 1993, the NSF decided to leave the management of the backbone entirely to competing, commercial backbones. To facilitate the growth of

overlapping competing backbones, the NSF designed a system of geographically dispersed Network Access Points (NAPs) similar to CIX, each consisting of a shared switch or local area network (LAN) used to exchange traffic. The four original NAPs were in San Francisco (operated by PacBell), Chicago (BellCore and Ameritech), New York (SprintLink), and Washington, D.C. (MFS).

Backbones could choose to interconnect with one another at any or all of these NAPs. In 1995, this network of commercial backbones and NAPs permanently replaced the NSFNET. The interconnection of commercial backbones is not subject to any industry-specific regulations. The NSF did not establish any interconnection rules at the NAPs, and interconnection between Internet backbone providers is not currently regulated by the Federal Communications Commission or any other government agency.

Instead, interconnection arrangements evolved from the informal interactions that characterized the Internet at the time the NSF was running the backbone. The commercial backbones developed a system of interconnection known as *peering*. Peering has a number of distinctive characteristics. First, peering partners only exchange traffic that originates with the customer of one backbone and terminates with the customer of the other peered backbone.

The original system of peering has evolved over time. Initially, most exchange of traffic under peering arrangements took place at the NAPs, because it was efficient for each backbone to interconnect with as many backbones as possible at the same location.

The dynamic nature of the Internet means that today's market structure and relationships likely will change. New services are continually being made available over the Internet. Many of these services, including Internet telephony and videoconferencing, are real-time applications that are sensitive to any delays in transmission. As a result, quality of service (QoS) is becoming a critical issue for backbones and ISPs.

Backbones face a number of private economic considerations in making such interconnection decisions. Any private decision by one or more backbones not to interconnect to guarantee QoS levels for new services may also have public consequences, however, because consumers of one backbone may not be able to use these new services to communicate with consumers of another backbone. This is a particularly important consideration for X-systems.

The decision to interconnect for the provision of QoS services would appear relatively similar to the one that backbones currently make when deciding whether to peer with one another. The backbones each calculate whether the benefits of interconnecting with one or more other backbones would outweigh the costs. The benefits of interconnecting to exchange

traffic flow from increasing the network of customers with whom one can communicate. This helps attract new users and encourages usage from existing users. The cost comes from a competitive network externality, as defined above; one backbone's decision to interconnect with another backbone makes the other backbone more attractive to customers. The widespread interconnection available today, in the form of either peering or transit agreements, indicates that the benefits of interconnection currently outweigh any costs.

There is, however, a difference between current interconnection arrangements and new interconnection arrangements for the exchange of QoS traffic. The Internet services that interconnection enables today, such as e-mail and Web access, already are universally available, and no one backbone or ISP could differentiate itself based on its unique provision of these services. Universal connectivity, however, is a legacy of the cooperative spirit that characterized the Internet in its early days. In the commercial spirit that pervades the Internet today, backbones and ISPs can view the new services that rely on QoS as a means to differentiate themselves from their competitors. A firm that introduces these new services may be less willing to share the ability to provide these services with competitors, because such sharing may reduce the ability of the firm to charge a premium to its own customers. For example, a while back, UUNET announced a service level agreement (SLA) that guarantees, among other things, the delivery speed (latency) of customers' traffic on its network. This guarantee does not extend to traffic that leaves UUNET's network, however, which encourages customers to keep traffic on-net.

Even if backbones agree in principle to interconnect to be able to offer new services that require QoS guarantees, they may face practical difficulties in reaching a final interconnection agreement. Aside from disagreements over the terms of interconnection, it is possible that the backbones, or their ISP customers, must support compatible versions of a particular new service to be able to exchange traffic that originates with one backbone's end user and terminates with another backbone's end user. Before committing to a particular standard for this service, backbones may wish to wait for an industry-wide standard to emerge.

This presents a coordination problem that may be difficult to resolve — in particular, if any firms have developed their own proprietary standards that they wish to see adopted as the industry-wide standard. In this situation, despite the fact that backbones would be willing to interconnect to exchange QoS traffic, the end result may be the same as if they were not willing to interconnect — end users would not be able to communicate across backbones using certain services.

Another potential issue relating to interconnection for QoS services is that it may exacerbate current congestion, and therefore it may be difficult

to guarantee QoS across backbones. Assuming that interconnection for QoS traffic is implemented under the current settlement-free peering system, backbones will not be paid to terminate QoS traffic. As a result, receiving backbones will have little or no economic incentive to increase capacity to terminate this traffic. QoS traffic that traverses networks may thus face congestion and would be unlikely to provide satisfactory quality. Of course, similar problems exist today with the current peering system, as described above, leading to the current congestion; but given the high data volume characterizing such services, the problem may be worsened. To provide the proper economic incentives to guarantee to customers that they can deliver QoS traffic across networks, backbones may have to implement a traffic-sensitive settlement system for such traffic.

If backbones are unable to overcome the economic, administrative, and technical hurdles to interconnect to exchange traffic flowing from new services requiring QoS, then the Internet faces the risk of balkanization. Backbones will only provide certain new services for use among their own customers. The result would be that network externalities, once taken for granted, would suddenly play a major role for consumers of Internet services. In the current environment of universal connectivity, consumers who simply want to send and receive e-mail and surf on the Web can choose any retail provider without worrying about the choices of other consumers or content providers. If the Internet balkanizes over the offering of new services, consumers would need to become aware of the choices of those with whom they wish to communicate when making their own choice of Internet provider. For example, a consumer who wishes to view real-time streaming video may need to be certain that the provider is connected to the same backbone to ensure high-quality viewing. Likewise, a business that wishes to use the Internet for videoconferencing must make sure that all relevant branches, customers, and suppliers are connected to the same backbone. Thus, any balkanization of the Internet would result in a classic example of network externalities; the specific backbone choice of each consumer would influence the choices of other consumers.

As a result of any balkanization of the Internet with respect to the provision of new services, customers wishing to communicate with a wide variety of others may end up subscribing to competing backbones, unless customers can coordinate on the choice of one backbone. This would raise the specter of the early days of telephony, when competing telephone companies refused to interconnect, resulting in many businesses and even some homes owning more than one telephone, corresponding to multiple local telephone company subscriptions.

There is some danger that a variation of this problem might be in the offing. As recently as November 2005, Vincent Cert wrote a letter to

Congress warning that the major telecoms could take actions to jeopardize the future of the Internet (Yang, 2006). Documents filed with the FCC indicated that Verizon Communications is setting aside a wide lane on its fiber-optic network for the purposes of delivering its own television service. This means that some rivals might be kicked out of the picture, all squeezing into the bandwidth that remains. Verizon contends that it needs to take such measures to earn a return on its network investments. Critics contend that no less than the future of the Internet is at stake. The Internet flourished because entrepreneurs and innovators anywhere, and any size, could reach consumers as easily as rich, large corporations. If Verizon and its peers set up “tools and express lanes,” these innovators and upstarts might not be able to afford the fees.

In some cases, notably the personal computer market, more than one standard emerges. This result has nevertheless been influenced by consumer demand, as Apple is widely seen as meeting the demands of a niche market, while the IBM (Intel/Windows) standard meets the more general needs of the mass market. It is worth noting here that it has been Internet protocols and applications, such as Web browsers and the Java language, that have served to meet the demands of users of IBM and Apple’s respective platforms to interact seamlessly with one another.

A final example of a standard emerging as a result of marketplace forces is the Internet itself. The protocols at the heart of the Internet, TCP/IP, only relatively recently became the dominant standard for networking, at the expense of a number of proprietary and nonproprietary standards, including SNA, DecNet, and X.25.

Although the marketplace is remarkably successful at generating compatible standards, it would be a mistake to conclude that this process is costfree for consumers or firms. Purchasers of Sony’s Betamax VCRs found it impossible to rent or buy movies after the VHS standard won the standards battle, while Sony was forced to concede and begin selling its competitors’ standard. The fax machine market was very slow to mature without a fixed standard, delaying the widespread adoption of a product that soon came to be regarded as almost indispensable for both consumers and firms.

IPv6

The next generation of Internet technology is called version 6, or IPv6. While the conversion to IPv6 is going to take quite some time, it is going to enable millions of devices to communicate information online through unique Internet Protocol (IP) addresses. IPv6 uses an address format of 32 numbers, compared to the current format’s (IPv4) use of just 12 numbers. Technical papers on this subject say that moving from

12 numbers to 32 numbers will increase the number of possible IP addresses from 4 billion to 340 undecillion, a virtually infinite number. This is exactly what X is waiting for.

X Requirements

X systems are collaborative systems, in that parts of them might reside on the Internet, an intranet, a client — including mobile clients, with each component interacting with the others.

While current architectures certainly do enable X systems, it is worthwhile to spend a little time discussing more advanced architectures that might be available. The first of these is P2P (peer to peer), sometimes referred to as a darknet. Perhaps the best description one can offer for how this works is Shawn Fanning's testimony before the Senate Judiciary Committee in 2000 (http://judiciary.senate.gov/oldsite/1092000_sf.htm). Fanning is the inventor of Napster.

I began designing and programming a real-time system for locating MP3 files of other users on the Internet. I designed the Napster software to find MP3s because they are the most compressed format (in consideration of bandwidth) and they were very popular at the time. The system I had in mind was unlike ordinary search engines at that time.

A traditional search engine sends out “robots” to roam the Internet periodically, updating itself every hour or more to remove sites that are down or unavailable. The database created is entirely driven by what the central computer finds by “crawling” the Internet. The indexes become outdated as sites go up or down, a significant problem when looking for MP3s because most of the files were housed on people's home computers.

My idea was to have users list the files they were willing to share on a computer that they all could access. That list would then be updated each time a person logged on to and off of that computer. The index computer would at all times have an up-to-date list of the files people were willing to share, and the list would be voluntarily made by the users as they logged on and off the system. A user searching the index would see all the files shared by users on the network and available to others on the network at that moment.

In contrast to traditional search engines, I envisioned a system that would be affirmatively powered by the users, who would select what information they wanted to list on the index.

Then, when the user exited the application, their portion of the list (their files) would automatically drop from the index. The index was only one part of participating in the community. I also wanted users to be able to chat with each other and share information about their favorite music, so I added these functions to the application...

The Napster system I designed combined a real-time system for finding MP3s with chat rooms and instant messaging (functionality similar to IRC). The chat rooms and instant messaging are integral to creating the community experience; I imagined that they would be used similarly to how people use IRC — as a means for people to learn from each other and develop ongoing relationships. I also added a “hot list” function that enables people to see others’ musical preferences by viewing the files they have chosen to share. This synergy of technologies created a platform for a community of users interested in music with different channels organized by genres of music (again, similar to IRC), and with genuine opportunity for participation, interaction, and individual involvement by the members’ sharing files together.

Napster is a throwback to the original structure of the Internet. Rather than build large servers that house information, Napster relies on communication between the personal computers of the members of the Napster community. The information is distributed all across the Internet, allowing for a depth and scale of information that is virtually limitless.

Napster does not post, host, or serve MP3 files. The Napster software allows users to connect with each other, in order that they may share MP3 files stored on their individual hard drives. The number of song files available at any given time depends on the number of song files that active users choose to share from their hard drives. Users need not share any or all of their files — they can choose which ones to make available to others. MP3 files do not pass through a centralized server. The transfer is directly from computer to computer, known as “peer to peer.” The “peer to peer” or decentralized nature of the technology means that the users, and only the users, determine what is shared....

I believe that the peer-to-peer technology on which Napster is based has the potential to be adopted for a many different uses. People generally speak about the ability to share other kinds of files in addition to music, and indeed, Napster has been contacted by entities such as the Human Genome Project

that are interested in sharing information among specific communities of interest. But peer-to-peer, or distributed computing, also has tremendous opportunity for sharing resources or computing power, lowering information and transaction costs. Peer-to-peer could be used to create a pool of resources in aggregate to solve a range of complex problems.

Peer-to-peer also has the potential to change today's understanding of the relationship between source and site. Think how much faster and more efficient the Internet could be if instead of always connecting you to a central server every time you click on to a Web site, your computer would find the source that housed that information nearest to you — if it's already on the computer of the kid down the hall, why travel halfway around the world to retrieve it? A number of companies, from Intel on down to small start-ups, are looking at ways to develop peer-to-peer technology, and I believe that many of them will succeed. The result will be not just a better use of computing resources, but also the development of a myriad of communities and super-communities fulfilling the promise of the Internet that its founders envisioned.

P2P then, as shown in [Figure 5.1](#), might be worthwhile to investigate as an architectural framework for X-enabled systems. In a P2P architecture, all client computers in a network act as servers and share their files with all other users on the network. Usually, only specific folders in each machine are made shareable for read access only. In an X-architecture, as shown in [Figure 5.2](#), content as well as executables could reside on a variety of platforms (e.g., various clients, servers, etc.), all centrally indexed on the main servers. In this way, a massively large, somewhat parallel universe of knowledge, content, and functionality can be distributed quickly across one or many organizations and end users.

Another possibility is the one used by online gaming systems. Rosedale and Gamasutra (2003) created an elegant, robust architecture for their Second Life virtual world environment (www.secondlife.com). Second Life uses a distributed grid for computing and streaming its virtual game world. The developers felt that this architecture best supported a large, scalable world with an unlimited amount of user-created and editable content. They also felt that this architecture exhibits a level of performance and content complexity that exceeds any static architecture.

Shards, a term originally conceived by Ultima Online, which is another gaming company, divides an online game into a number of parallel worlds through the use of clustered servers. These have identical sets of world content but different sets of users. The advantage of this approach, referred

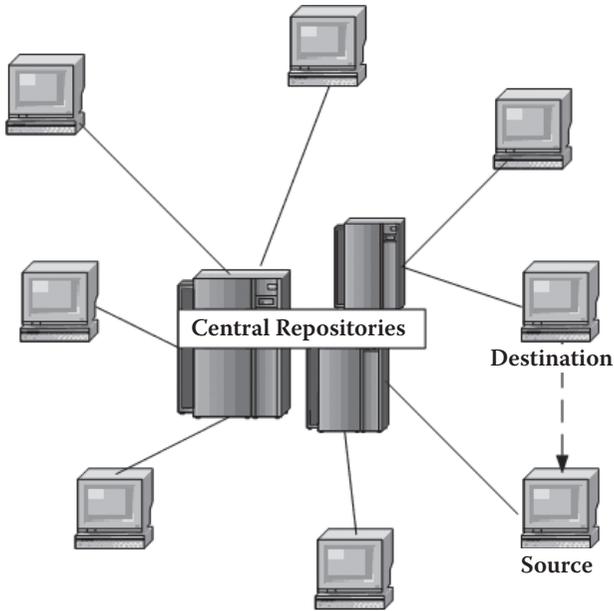


Figure 5.1 Typical P2P architecture.

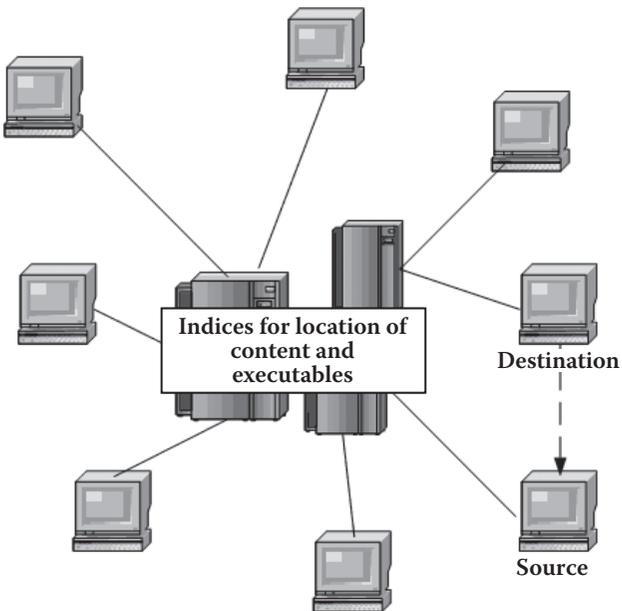


Figure 5.2 A P2X architecture.

to as shard clustering, is that it allows creators to multiply their content development efforts by reusing content across multiple sets of users. A major drawback, however, is that it separates users into separate, noninteracting worlds. This means that friends are not able to collaborate unless they are logged in to the same server. This also has the effect of creating a load-balancing problem.

Putting the entire Second Life worldscape into one world would have presented massive database challenges and would have required at least 100 CD-ROMs to store all content. This challenge was met by the use of a distributed grid approach. This approach is best used when handling a massive number of objects (i.e., an object explosion). Its purpose is to distribute the objects across a large system such that no bottlenecks are created.

Rosedale and Gamasutra (2003) built topologically tiled grids of “nearest neighbor” connected machines. These machines talk only to the four nearest neighbors, so there is no transactional scaling problem as the virtual world becomes very large. The typical bandwidth between adjacent machines is in the hundreds of kilobits per second. Therefore, it is possible that the servers can be in different co-location centers, and do not require connection via gigabit Ethernet or in the same physical cluster.

As objects move around the world, relevant information is transferred from machine to machine as they cross over the edges. Through the use of high-order prediction of the client, which is downloaded to the end user’s client PC, the transitions are not visible to the end user.

Second Life’s grid topology allows for a tremendously large, seamless world, with computational and storage loads spread across many machines. The Second Life viewer client is itself distributed via the Web.

As is evident, there is a wide variety of possibilities when considering the architecture for an X-system network.

Smart X Structures

The U.S. Government was responsible for the creation of the Internet. As everyone now knows, the Defense Department (DARPA) created the precursor to the modern Internet as a method for crisis and emergency management. Because crisis management is still a major focus, it should come as no surprise that the government is quite forward thinking when it comes to tracking down novel, proposed architectures that will serve to advance the state of the modern Internet. Toward that end, the Office of the Manager in the National Communications System (2002) focused on a variety of “smart X structures” (this author’s term) in their discussion of network planning for 21st century intelligent systems and smart structures. This section discusses some of these structures.

Enterprise Nervous System (ENS)

Gartner has introduced the concept of the Enterprise Nervous System (ENS), in which the ENS is an intelligent network that connects people, application systems, and devices — possibly distributed at various locations, within different business units, and using diverse systems — in a real-time, proactive virtual enterprise. Although the ENS is an evolutionary step in system concepts, it is revolutionary in elevating the network to a new level, in that much more of the intelligence will reside in the network, as well as in the applications. The ENS offloads logic from the application systems by supplying higher “quality-of-service” communication, transforming messages, redirecting messages as appropriate (using logical business rules), and sometimes even tracking and controlling business processes (Schulte, 2001).

Intelligent Agent-Based Systems

Forrester’s Radjou (2003) predicts the wholesale adoption of applied software agents in stages. The first stage saw software agents used to interpret environmental data. The agent-managed stage, predicted to be prevalent between 2006 and 2008, will make good use of standards such as BPEL4WS (Business Process Execution Language for Web Services) to power net-resident, Web-service infrastructure offerings. BPEL4WS, developed by vendors such as IBM, BEA, Microsoft, SAP, and Siebel (<http://www-128.ibm.com/developerworks/library/specification/ws-bpel/>) defines an interoperable integration model that should facilitate the expansion of automated process integration in both the intra-corporate and the business-to-business spaces. These same vendors are expected to develop tools that use notation for Agent-Based Unified Modeling Language (AUML) (<http://www.auuml.org/>) such that business agents rather than programmers are able to design agent-based applications. Radjou predicts that by 2009, agent-optimized systems will be common, thus “paving the way for a network of systems and machines, not people.”

Intelligent systems originally emerged from the artificial intelligence community. There was a great deal of interest in knowledge-based systems, which are rule-based intelligent systems (also referred to as expert systems). These systems were considered flexible because rule bases could be easily modified. However, in practice, these early systems were often monolithic rule-based systems, and as the intertwined rule base grew in size with often over 1000 interdependent rules, it often became difficult to understand and maintain. More recently, a much more flexible, distributed, and adaptable paradigm has emerged for intelligent systems, namely that of intelligent agent-based systems.

There are many definitions of agents (Milojic, 2000). The dictionary definition of agent is “one that is authorized to act for or in the place of

another as a representative, emissary, or a government official.” From the standards organizations, the Object Management Group (OMG) defines agents as “computer programs acting autonomously on behalf of a person or organization.” The Foundation for Intelligent Physical Agents (FIPA), another standards organization, defines agents as “computational processes implementing an application’s autonomous communicating functionality.”

Agents vary considerably in their capabilities. While much of the early research was in artificial intelligence, the growth of the Internet has emphasized distributed computing. Java, which has many features for distributed computing, has become a popular language for programming agent-based systems. Distributed Java technologies, in particular Remote Method Invocation (RMI) and Java Intelligent Network Infrastructure (Jini), have provided the distributed application infrastructure for distributed agent-based systems implemented in Java.

An intelligent agent-based system is a highly distributed system in which agents are active concurrent objects that act on behalf of users. Agents are intermediaries between clients, which are typically user interface objects, and servers, which are typically entity objects that store persistent information. Usually, several agents participate in the problem-solving activity, communicating with each other. This leads to a more distributed and scalable environment.

Agents can be categorized in different ways, based on their mobility, their intelligence, or the roles they play in an agent-based system. One categorization is whether the agent is stationary or mobile.

Another categorization is based on the following capabilities:

- Cooperative agents communicate with other agents and their actions depend on the results of the communication.
- Proactive agents initiate actions without user prompting.
- Adaptive agents learn from past experience.

Agents can combine the above three capabilities (Case et al., 2001) as shown in [Figure 5.3](#). Personal agents are proactive and serve individual users — they may also be adaptive. Adaptive personal agents can search for user information in background mode and are often coupled with the World Wide Web. They can refine their search strategies based on how the user reacts to previous searches. Collaborative agents are both proactive and cooperative.

Knowledge Rover Architecture

Another categorization of agents was carried out as part of the Knowledge Rover Architecture, developed in support of the Defense Advanced

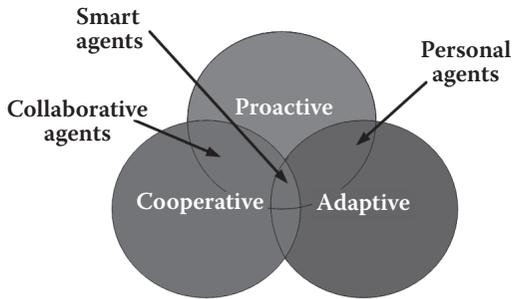


Figure 5.3 Categorization of agents.

Research Projects Agency (DARPA) Advanced Logistics Program. Knowledge rovers are cooperating intelligent agents that can be configured automatically with appropriate knowledge bases (ontologies), task-specific information, negotiation, and communication protocols for their mission into cyberspace. The different agents include:

- *Coordinator agents*: coordinate the activities of a group of agents. They are informed of significant events. A significant event can lead to the activation of new agents. For example, if the enterprise is notified of a payment request, then the coordinator agent would coordinate with other agents in implementing the payment scenario.
- *User agents*: act on behalf of a user and are responsible for assisting users in (1) browsing catalogs and information holdings such as the information repository, (2) the intelligent formulation of queries, and (3) the planning of tasks within a mission-specific scenario such as provisioning logistic support for a disaster recovery effort.
- *Real-time agents*: mission-specific, defined and configured to process incoming data and update the appropriate database or notify the appropriate users. The real-time agents are autonomous, and communicate with each other using a predefined protocol. They are responsible for monitoring the external environment, interacting with other systems, or acting on inputs from users. When an event is detected by a real-time agent, it is signaled to the relevant agents.
- *Facilitation agents*: provide intelligent dictionary and object location services. For example, a facilitation agent might accept a request from the coordinator agent to find all *external* providers of “PCs,” and it might respond with the hardware manufacturers and suppliers for the region in question. Other agents such as knowledge rovers (defined below) could then arrange for the items to be requisitioned, retrieved, and paid for. A knowledge rover

could also post a request for bids, accept responses, make contracts, and provision the requested items.

- *Mediation agents*: configured to assist in the integration of information from multiple data and information sources having diverse data formats, different meanings, differing time units, and providing differing levels of information quality. Mediators (Wiederhold, 1996) are configured to accept queries from the coordinator agent, translate the queries into the query language of the appropriate database system, accept the retrieved result, integrate it with results from other sources, and return the information to the coordinator agent for presentation to the user agent.
- *Active view agents*: created to monitor real-time events from the environment or from databases, and to use these events to initiate actions that will result in the update and synchronization of objects in the data warehouse and also in local views maintained at user workstations. These agents are configured to perform very specialized monitoring tasks and have a collection of rules and actions that can be executed in response to events and conditions that occur in the environment or in the databases.
- *Information curators*: responsible for the quality of information in the information repository. They assist in evolving the data and knowledge bases associated with enterprise information resources. They work with knowledge rovers to incorporate newly discovered resources into the information repositories.
- *Knowledge rovers*: instructed to carry out specific tasks on behalf of the coordinator agent, such as to identify which vendors have a specific item on hand. This would involve obtaining information from several vendors. The knowledge rover dispatches field agents to specific sites to retrieve the relevant information. If the knowledge rover obtains similar information from more than one source, it may ask a mediator to resolve the inconsistency. The knowledge rover reports back to the coordinator agent. The rovers are also responsible for Internet resource discovery. These new information sources and their data are analyzed to determine the adequacy, quality, and reliability of retrieved information and whether it should be incorporated into the information repository.
- *Field agents*: specialized rovers that have expertise in a certain domain (e.g., pharmaceuticals) and knowledge about domain-specific information holdings at one or more sites. For example, a field agent could be tasked to monitor all aspects of a single item, say a “PC” produced by several manufactures and distributed by several vendors. The field agent negotiates with the local systems, retrieves appropriate data, and forwards it to the appropriate requesting agent.

Intelligent Integration of Information

The main problem confronting intelligent integration of information in X-systems is to access diverse data residing in multiple, autonomous, heterogeneous databases, and to integrate or fuse that data into coherent information that can be used by decision makers. To make the problem even more interesting:

- Data may be multimedia (video, images, text, and sound).
- Data sources may store data in diverse formats (flat files, network, relational- or object-oriented databases).
- Data semantics may conflict across multiple sources.
- Data may have diverse temporal and spatial granularities.
- Data that is interesting and valuable may reside outside the enterprise.
- Data may be of uncertain quality, and the reliability of the source may be questionable.

Clearly, one cannot expect to solve information integration and other large-scale system problems with a monolithic and integrated solution. Rather, the system should be composed of smaller components, with each component having the requisite knowledge to perform its tasks within the larger problem-solving framework. Thus, the use of *cooperative intelligent agents* holds promise in helping address, discuss, and understand the issues in building next-generation intelligent information systems.

Bird (1993) has proposed an agent taxonomy based on two client/server classes. They are *mobile agents* (clients) for content, communications, and messaging services, and *static agents* (servers). Bird notes that distributed intelligent systems share many of the same characteristics of multidatabase systems (Sheth and Larson, 1990), in particular, distribution, heterogeneity, and autonomy. Knowledge and data would be distributed among various experts, knowledge bases, and databases, respectively, thus making problem solving a cooperative endeavor.

There are several facets to the heterogeneity of information in systems, including:

- *Syntactic heterogeneity* refers to the myriad of knowledge representation formats (Gruber, 1993), data definition formats to represent both knowledge and data.
- *Control heterogeneity* arises from the many reasoning mechanisms for intelligent systems including induction, deduction, analogy, case-based reasoning, etc.
- *Semantic heterogeneity* arises from disagreement on the meaning, interpretation, and intended use of related knowledge and data.

Table 5.1 Three Layer Internet Information Architecture

<i>Information Layer</i>	<i>Layer Service</i>
Information Interface Layer	Users perceive the available information at this layer and may query and browse the data. This layer must support scalable organizing, browsing, and search.
Information Management Layer	Responsible for the replication, integration, distribution, and caching of information.
Information Gathering Layer	Responsible for the collecting and correlating the information from many incomplete, inconsistent, and heterogeneous repositories.

A third characteristic of intelligent systems is that of autonomy. There are several aspects to autonomy — in the control structure of an agent, in the extent to which an agent shares information with other agents, the manner in which an agent associates with other agents, and structural autonomy in the way an agent fits into an organization of agents for problem solving.

Bowman et al., (1994) describe a three-layer architecture for scalable Internet resource discovery, proposed by the Internet Research Task Group. Table 5.1 denotes the three-layer architecture that provides access to heterogeneous repositories, including those on Web servers.

The Intelligent Integration of Information Program identified a collection of services to support the following types of activities:

- *Information Interface Layer*: thesaurus services, information search services, query decomposition and semantic query optimization services, and information presentation services
- *Information Management Layer*: information integration, real-time subscription and notification services, and information mediation services (e.g., knowledge rovers, facilitators, and brokers), and federation services
- *Information Gathering Layer*: wrapper services, query execution services, and data harvesting services

Conclusion

X systems are naturally heterogeneous. Data and function must be distributable across a variety of devices. This is growing in importance as our society becomes far more mobile. For example, over 300 million camera phones were sold in 2005 alone. According to Hewlett-Packard,

worldwide penetration is estimated at more than a billion. Tim Kindberg is a senior researcher at HP Labs in the United Kingdom. Kindberg and his associates imagine the camera phone operating like a computer mouse, making the camera phone easier to access mobile content. HP Labs created a camera-based code reader. Similar to scanners at the grocery store, this software enables the camera to read data-rich codes. When scanned, the codes trigger a variety of services. For example, they can open Web content, send text messages, access help lines, or download discounts. The codes themselves can be located in newspapers, magazines, signs in stores, or even on billboards.

Dedicated plug-and-play appliances are also getting some traction in this X-enabled world scene. In 2005, a niche technology called the XML acceleration appliance began to pique everyone's interest. This technology moves the load of XML processing from an application server to a dedicated plug-and-play piece of hardware (Goth, 2006). X, it seems, is going to come in a variety of flavors.

References

- Bird, S.D. (1993). Toward a Taxonomy of Multi-Agent Systems, *International Journal of Man-Machine Studies*, 39, 689–704.
- Bowman, C.M., Danzig, P.B., Manber, U., and Schwartz, M.F. (1994). Scalable Internet Resource Discovery: Research Problems and Approaches, *Communications of the ACM*, 37, 98–107.
- Case, S. Azarmi, N. Thint, M., and Ohtani, T. (2001). Enhancing E-Communities with Agent-Based Systems, *IEEE Computer*, 34, 64–69.
- Goth, G. (2006). XML: The Center of Attention Up and Down the Stack, *IEEE Distributed Systems Online*, 7(1).
- Gruber, T. (1993). A Translation Approach to Portable Ontology Specifications, *Knowledge Acquisition*, 5, 199–220.
- Kende, M. (2000, September). The Digital Handshake. Connecting Internet Backbones. Office of Plans and Policy Federal Communications Commission Working Paper No. 32. Retrieved from: http://www.fcc.gov/Bureaus/OPP/working_papers/oppwp32.pdf.
- Milojic, D. (2000). Agent Systems and Applications, *IEEE Concurrency*, Vol. 8. Office of the Manager. National Communications Systems. (2002, February). Network Planning for 21st Century Intelligent Systems and Smart Structures: Advanced Concepts in Telecommunications for National Security and Emergency Preparedness. Retrieved from: http://www.ncs.gov/library/tech_bulletins/2002/tib_02-1.pdf
- Radjou, N. (2003). Software Agents in Business: Steady Adoption Curve. Forrester Whole View TechStrategy Research.
- Rosedale, P. and Gamasutra, C.O. (2003, September 18). Enabling Player-Created Online Worlds with Grid Computing. Retrieved from: <http://www.cs.ubc.ca/~krasic/cpsc538a-2005/papers/rosedale.pdf>

- Schulte, R. (2001). Designing the Agile Enterprise. Presented at *Application Integration: Moving Toward Total Business Integration*, San Francisco, CA.
- Sheth A. and Larson, J. (1990). Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, *ACM Computing Surveys*, 22, 183–236.
- Wiederhold, G. (1996). Foreword to Special Issue on the Intelligent Integration of Information, *Journal of Intelligent Information Systems*, 6(2/3), 93–97.
- Yang, C. (2006, February 6). Is Verizon a Network Hog?, *Business Week*.