PART I

A Simple Introduction to Complex Event Processing

Highlights in Part I:

- *The challenges involved in keeping the human in control of today's electronic information systems*
- *The basic concepts of CEP*
- *How CEP can be employed to meet the challenges*

Chapter 1

The Global Information Society and the Need for New Technology

- *Events everywhere in our information systems*
- *The Internet and the growth of global communication spaghetti*
- *Layers upon layers in enterprise system architectures*
- *Global electronic trade—understanding what is happening*
- *Agile systems—future reality or just a dream*
- *Can an open electronic society defend itself?*
- *The gathering storm—global coordination or global chaos*

Information processing using computer systems on a global scale has become the foundation of twenty-first-century life. It runs our governments and industries, our transportation systems, hospitals, and emergency services. It is the foundation for global economics and global electronic trading in the new millennium.

Information processing systems have grown up around the globe at "*Web speed,*" only slightly slower than "warp speed" in science fiction. The Internet or "Web" has been a driving force. The technology to build these systems, to make them faster, capable of handling and routing larger and larger amounts of information, has developed to help drive this growth. New

applications are appearing all the time to entice us into new ways of using IT systems.

But there is no similar development of foundational technology for monitoring and managing the information that flows through global information systems. And the fact is, if we do not know what is going on in these systems—and I mean "know" in the sense of human understanding—we cannot protect them, and we cannot use them to maximum advantage. This chapter describes our event-based world and some of its challenging problems.

1.1 Distributed Information Systems Everywhere

Typical examples of distributed computing and information systems are systems that automate the operations of commercial enterprises such as banking and financial transaction processing systems, warehousing systems, and automated factories. The Internet has promoted and speeded the growth of distributed information processing beyond the single enterprise, across the boundaries between enterprises. Information is shared between different enterprises and provides the foundation for trading partnerships and the automation of business collaborations.

Figure 1.1 shows a multi-enterprise financial trading system. These systems are distributed over various networks worldwide and often use the Internet as one of the networks. When viewed at a macro level, the various enterprises and organizations are simply components of the system. Each of them has its own internal information processing system. The figure shows these components, including the stock market information system, brokerage houses and online customers (or more accurately, their workstations), the Federal Reserve, investment banks, and the networks through which all these components communicate. Messages (or "events") flow across networks between these enterprises. They react to the events they receive and issue new events that are sent to other components. The system is *"event driven"*—it lives or dies based upon the messages flowing across its IT networks. This is a rather large system with high volumes of messages flowing through its networks. For example, in 2001, 5,000 or 10,000 messages per second flowed through a single large brokerage house's information technology layer. Soon, that number will be higher.

Financial trading systems are but one example of distributed IT systems. Generally speaking, the business operations of any global coporation are supported by a widely distributed, message-based computer system. Figure 1.1, by the way, shows tools for complex event processing added to the distributed system—we will explain this later. In fact, that's what this book is about.

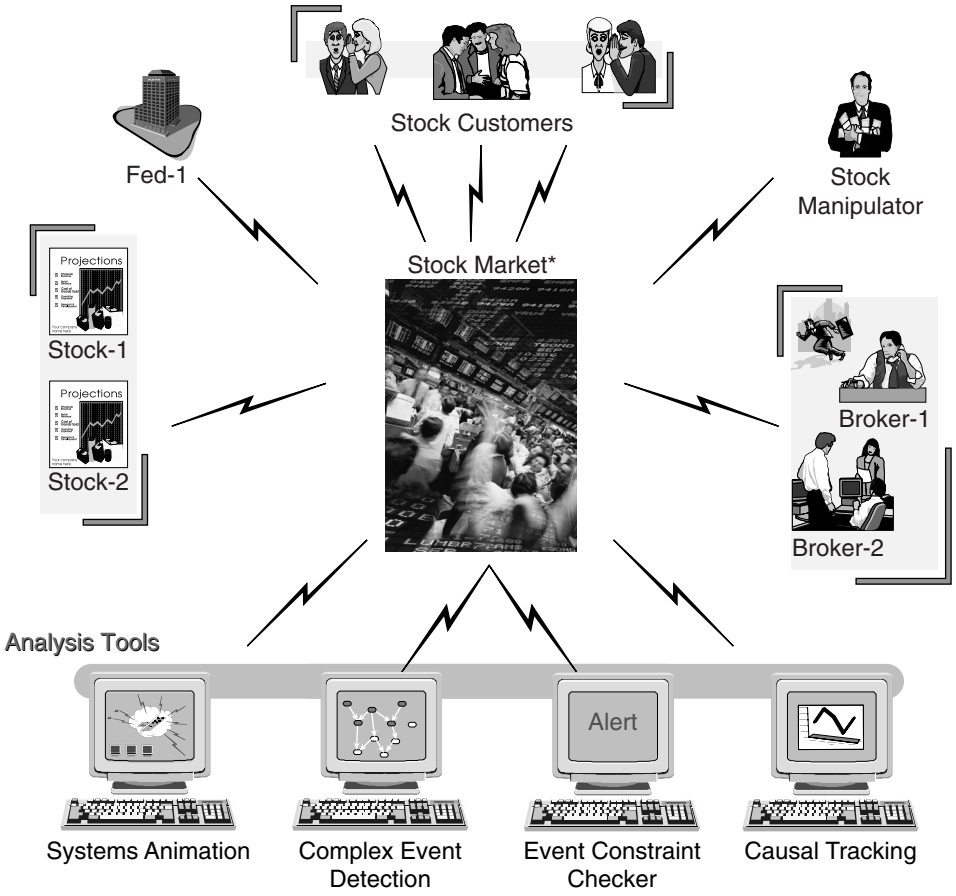

Figure 1.1: A system of distributed communicating enterprises (*photo by Wonderlife USA Corporation)

Government and military information systems are also distributed systems. Figure 1.2 depicts a typical military command-and-control system that links command centers, intelligence-gathering operations, and battlefield units of all services—a so-called C^4I (command, control, communications, computers, and intelligence) system. At first look, a military system may seem to be entirely different from a commercial one. Certainly, the military goals and operational conditions are very different, and many of the types of component objects in military systems are different from commercial systems. But military systems also contain a lot of commercial components, such as operating systems and databases, and often parts of their IT layer use the same publicly available networks, such as the Internet.

In fact, all three kinds of systems have a lot in common. In all cases—commercial, government, and military—the underlying architectural

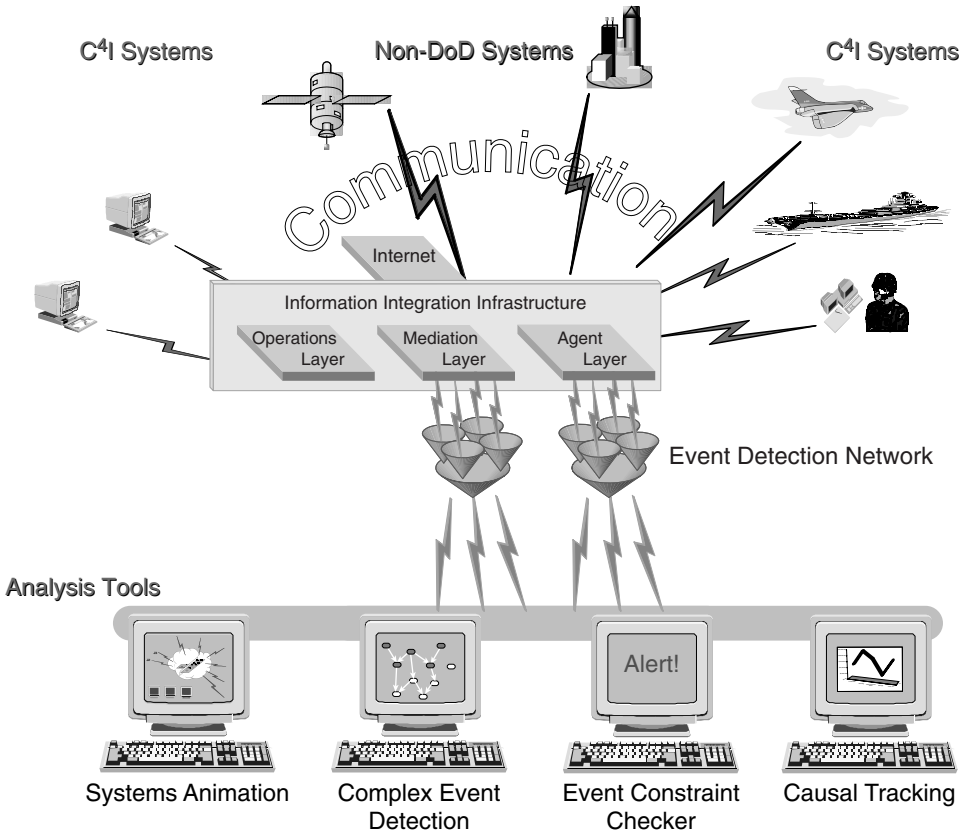

Figure 1.2: A command-and-control system monitored by an event processing network

structure is the same: a distributed information system consisting of a widely dispersed set of several thousands or hundreds of thousands of application programs (or *component objects,* as they are often called) communicating with one another by means of messages transmitted over an IT layer containing various kinds of media.

All these systems have come to be collectively called "*enterprise systems.*" They all have a common basic problem. Their activities are driven by the events flowing through their IT layers. And they produce zillions of events per hour or day. But there is no technology that enables us to view those events and activities that are going on inside these systems in ways that we humans can understand. To be sure, given the primitive tools we have at the moment, we can *see* the events. But making *sense* of them is the problem!

Enterprises invest as best they can in tools to monitor events in the basic networks that carry information. So we are told, for example, that "the router in Hong Kong is overloaded." We have to figure out that the router problem may be holding up completion of an important trading agreement between our New York office and our partner's Tokyo office. We need to be able to answer questions about events that are not simply low-level network activities, but are high-level activities related to what we are using the systems to do—so-called business-level, or strategic-level, events. We need answers to questions like these:

- "What caused our trading system to sell automobiles (a business-level event) to a customer in Texas?" The answer could involve several other trading transactions.
- "Is the system, at this moment, under a denial-of-service attack?" The answer requires real-time recognition of complex patterns of events, the patterns that indicate such attacks.
- "What is causing the system to fail to execute this trading agreement?" The answer could be a missing set of earlier business-level events, like failures of several suppliers' systems to respond within time limits.

These questions are about *complex events,* which are built out of lots of simpler events. Answering them means that we need to be able to view our enterprise systems in terms of how we *use* them—not in terms of how we *build* them, which is the state of the art in enterprise monitoring technology today.

1.2 The Global Communication Spaghetti Pot

The World Wide Web and all large-scale networks are constructed on well-understood network engineering principles, protocols, and conventions. They allow the ultimate flexibility in communication whereby any two computers have the potential to communicate with one another. This flexibility, or "openness," allows communication to take place, and information to flow, in patterns we do not understand.

Consider for a moment what happens when you visit a Web site today. Suppose you are the client in Figure 1.3 visiting a Web site, say, www.anymumble.com. You click on something on the Web site screen. You get a screen full of banner advertisements, blinking applets, links to other sites, and, maybe if you're lucky, the information you were looking for. Is all this stuff coming back at you from www.anymumble.com? Very unlikely. What you see on your computer screen may actually be coming from several

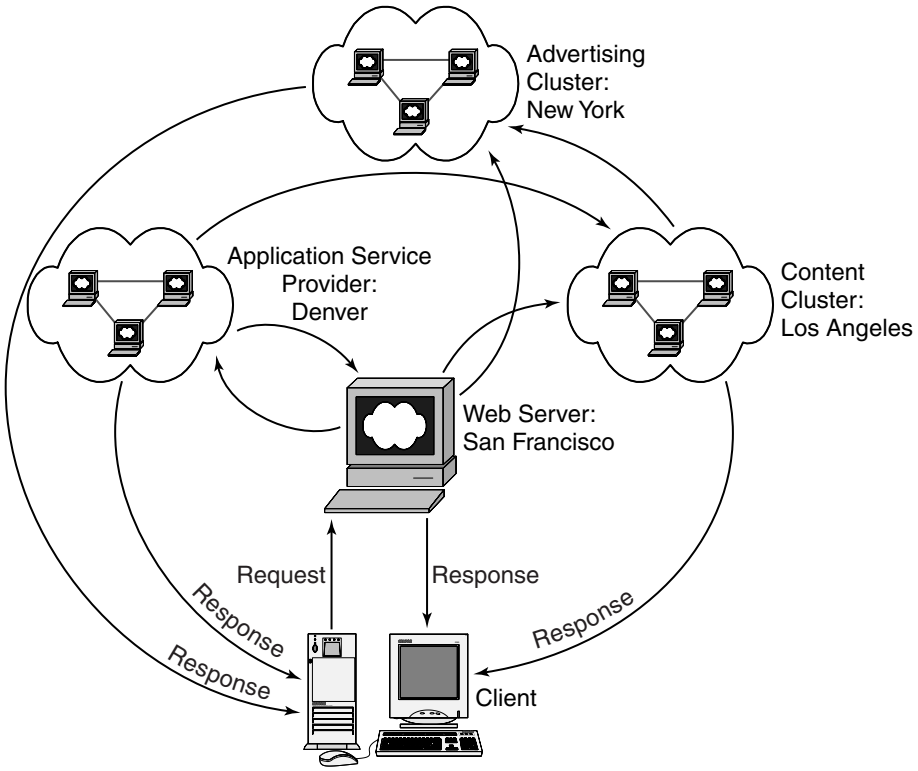

Figure 1.3: Communication spaghetti across the Internet

different places on the Internet, as Figure 1.3 shows. And the next time you visit www.anymumble.com, what you get may be coming from a totally different set of Web sites. In fact, what you get and where it comes from may depend upon who the anymumble site thinks you are and what you did the last time you visited it. That's communication flexibility for you!

Another simple example of communication flexibility is the now very common e-mail virus attachment. The "I Love You" virus in 2000 was a fine example. Released in the Phillippines as an attachment to e-mail, within a few days it wrought havoc in computer systems across the world, estimated in the billions of dollars. The virus even took down chunks of computers within the U.S. Department of Defense, supposedly protected from cyber attack behind communication firewalls. How did it travel so far so fast? Through the simple communication flexibility of forwarding e-mail across the Internet, together with the address book features in our mailer programs that let us do that easily.

Communication flexibility is one of the great powers of the Internet. We certainly don't want to restrict this flexibility in any way. On the positive

side, it is enabling rapid growth in new kinds of business activities. One example is *outsourcing,* which enables a Web site that supplies services to subcontract with other Web sites to supply some of those services. When you go to a Web site that offers home mortgages, for example, it communicates with lenders over the Internet to get you the latest bids that fit your requirements. If you apply for a mortgage, it may outsource a credit check to a credit agency. All this happens in a second or two. Another example is *automated trading Web sites,* or "eMarketplaces" as they are called. These let many enterprises, buyers and suppliers of goods, communicate to complete multistep transactions in a matter of minutes. An automobile manufacturer can reduce inventory costs by tracking parts suppliers' inventories and prices, and placing orders "just in time" that are spread across several suppliers to reduce costs. eMarketplaces are connecting together and may eventually morph into the "global eCommerce Web."

The flip side is that the communication flexibility of the Internet makes it difficult to track the causes of events. Because events can come flying at us from anywhere on the Web, we often cannot figure out why they happened or what other events caused them. For example, when a trading transaction fails to complete, there is nothing to tell us why. Whenever we need to know the causes of a message, we have to resort to searching through very large, low-level, network log files and, often, lots of them. It is tedious, expensive, and very primitive. And often fails. We have no *causal* tracking that can tell us what events caused another event to happen. There is no technology that lets us detect when events at different places—say, Web sites in New York and London—at different times, have a common cause, or if complex patterns of events happening globally are repeating.

For example, at a staff meeting of a large commercial Web site, the CEO might ask the IT manager, "There was an unusually high number of complaints from customers in the New York area about delays in getting pages on their screens—why?" A reply might be, "We think it was the XYZ advertisement server we outsourced to for those banner ads directed at the Northeast, but XYZ says their demand loads were light. And the ISPs aren't taking any blame either."

Communication flexibility on the Internet has led to a very dynamic communication architecture in which the links between various computers are constantly changing. At any given moment in the life of any large IT network, we don't have a global view of who's communicating with whom, nor what is causing what, and we have to work very hard to find out. This is communication traffic we do not understand.

What we have today is *global communication spaghetti*. We can't tell where it starts and where it ends, we can't unravel it, and most of the time we don't know how it happened.

The challenge is *not* to restrict communication flexibility, but to develop new technologies to *understand* it.

The technology of monitoring and managing events in IT systems has been completely overrun by the technology of communicating events. This is a global problem in search of new ideas on how to solve it.

1.2.1 Event Causality

A key to understanding events is knowing what *caused* them—and having that causal knowledge at the time the events happen. The ability to track event causality is an essential step toward managing communication spaghetti.

Events are flowing all the time through our Internet-based systems from one part of the world and causing events in another part of the world. For example, suppose I send you an e-mail message, say, M, and you reply with another message, say, N. Then my message, M, *caused* your message, N. This is so obviously important to my being able to understand your reply, which may come back days later when I have forgotten my M, that the "reply" feature of mailers usually attaches M inside of N.

Another example is pointing a browser at the www.anymumble.com location, which is an event. This event then causes all those applets and banner advertisements to appear on our screen—that is, lots of other events. This example is not so simple because there are a lot of intermediate events flowing between the anymumble Web site and other Web sites before we get the events on our screen. So tracking how we got the stuff on our screen is not so easy.

In examples like these, the causal relation between the events is sometimes called *horizontal causality* to emphasize that the causing and caused events happen on the same conceptual level in the overall system. In the e-mail example, N is an event caused by M, and both events happen at the level of using e-mail applications. So the causal connection is "horizontal"—that is, at the same level. Events can also be components of other events, in which case the component events are thought of as happening at a lower level. The component events of an event also cause that event. This kind of causal relation between events is called *vertical causality,* as described in the next section.

1.3 Electronic Archeology: Layers upon Layers

Enterprise systems are *distributed, event-driven systems*. But they all have another property in common. They are *layered systems*. Layering is a design technique for controlling complexity. But it is also a side effect of unplanned or uncontrolled growth. In some systems the situation bears analogy with the multiple layers of civilizations of different historical periods, one on top

of another, that the archeologist Heinrich Schliemann discovered at Troy, back in the 1870s.

Layered IT systems present another dimension in the search for new ways to understand the events that happen in them.

1.3.1 A Layered Enterprise System

Figure 1.4 shows a common layered structure of an enterprise system.

Applications and Users at the Top

The top layer is called the *business level* or *strategic planning level* because it is the layer at which an enterprise plans and transacts its business. It contains the applications that people use to transact business and do their

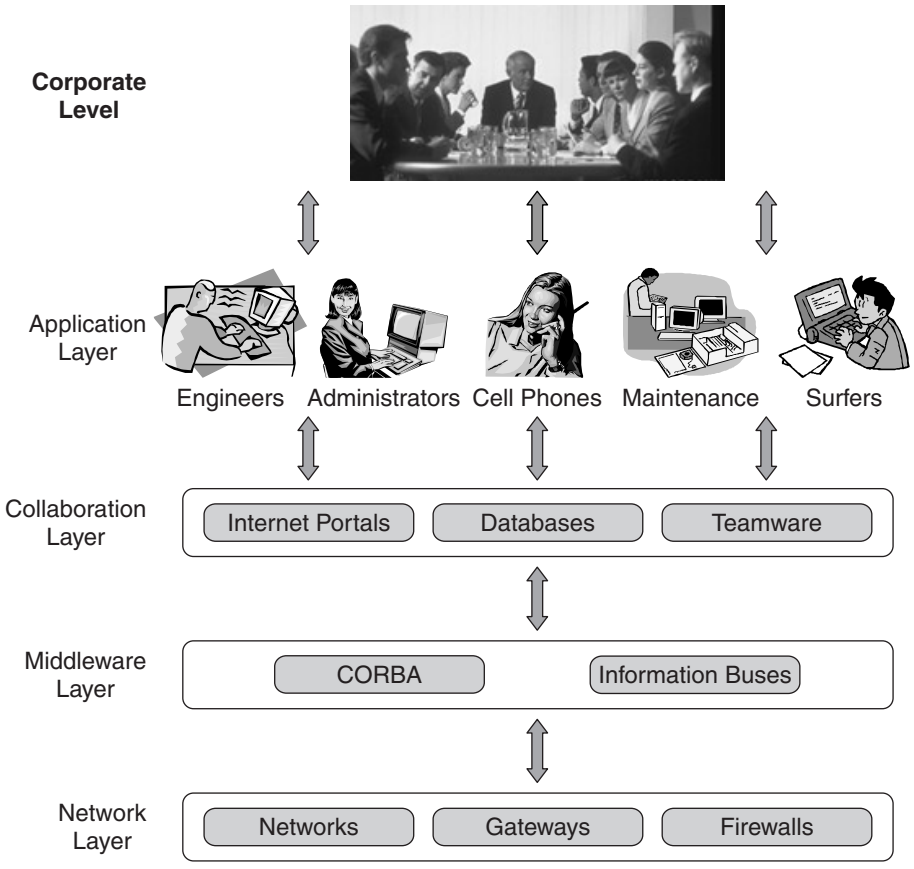

Figure 1.4: Typical layers in an enterprise system

work. This layer contains user interfaces to the enterprise's services, planning and forecasting applications, inventory and accounting applications, spreadsheets, calendars, Web browsers, document preparation tools, e-mail, communication devices such as cell phones, system administration tools, and so on. The contents of this layer are expanding rapidly. Different kinds of applications are appearing all the time, each requiring different types of information.

The application layer is the level at which human users work and conceptualize their goals—for example, "I want to send e-mail to so-and-so," or "I want to initiate a process to purchase item X from the cheapest supplier." Events that signify activities at this level are the ones humans can understand most easily. These events result from using applications.

The high-level events that are of greatest significance are not explicitly generated by users using applications, but they are *consequences* of sets of other high-level events. For example, a single *business-level* event could consist of a sequence of events indicating uses of various applications. It could signify a completed supply chain transaction with business partners, negotiated by messages across the Internet. It is a *virtual* event signifying a very real activity with contractual and economic implications. These inferred events are aggregated from sets of other high-level events and are the events of interest to the business level or strategic planning level of the enterprise.

Another example of a complex business level event is a stockbroker, or a group of brokers, in a financial trading enterprise trading stocks on many different accounts, including those owned by the trading house itself. Some patterns of trading activities on various accounts might violate Securities and Exchange Commission (SEC) regulations. But high-level events signifying trading violations have to be inferred from the explicit trade events generated by the brokers when they make trades. It depends on an ability to recognize *complex patterns* of events involving timing and relationships between the events. We understand a "violation event" when it is pointed out to us—no problem! But it isn't actually generated. That's the problem! It is a *virtual event* signifying a very real activity. It has to be recognized from the morass of actual events.

There are myriads of similar situations going on in enterprise systems all the time that have the potential for similar kinds of violations of the enterprise's policies—for example, in online Internet-based auction houses and in "dot-com" retailing, where violations of the regulations of various states may happen.

Conceptually, the events of interest to the highest layer of an enterprise, or a system of multiple cooperating enterprises, are virtual events comprising sets of application layer events. The *strategic planning layer* must be fed aggregations of application-level events.

But application-level events are not subjected to a battery of monitoring and analysis tools as the network-level events below it are. This is unfortunate but understandable, because such tools would need a new kind of technology that deals with complex patterns of events to be effective.

So, in a layered system, understanding at the highest level what is going on in the enterprise, at every layer, is one problem. But there are more problems to come because causality between events crosses layers.

Collaboration and Enabling Layer

Next in this figure comes the collaboration layer, which contains components that help make applications available to users. Here we find components that we have come to expect, such as databases and servers for e-mail groups, news, and chat rooms. We might even go so far as to include the operating systems, such as Windows and Linux, in this layer, although it could be argued that they belong in layers below. The collaboration layer now contains a growing number of products that enable more sophisticated uses of distributed information processing. For example, Web sites that service applications (so-called application service providers, or ASPs). And business rules engines that enable trading agreements and correlation of information. The collaboration layer contains more and more tools that enable users and applications to interact intelligently on complex projects, like building software systems or putting together multistep trading deals. This layer is the target of a growing industry in business-to-business products and services (often called B2B).

The dividing line between the collaboration layer and the application layer is not a definite, rigorous line. Sometimes, for example, a database would be more appropriately viewed as an application rather than as an enabler of applications, in which case we would put it in the layer above.

Middleware

The next layer down is the *middleware layer,* which lets all the stuff above communicate. Included are Object Request Brokers (ORBs) and messaging systems such as information buses. This level of communications sits on top of the basic networks and lets all the applications and application servers talk to one another. It contains the system components that are often categorized as Enterprise Application Integration (EAI) products. It is called "integration" because it helps link business-level applications by letting them communicate. And it is called the middleware layer because its components are viewed as lying "in the middle" between applications and networks.

Before 1990 this layer and the products in it didn't exist. It came about by design, to make it easier to get applications to communicate over networks. Middleware is a go-between, translating the message communication

between applications into low-level messages called packets that networks are designed to carry.

Communications middleware has become an industry in itself. Middleware for distributed applications includes ORBs and various kinds of message-oriented middleware based upon message queues, publish/subscribe (pub/sub) services, and so on. These products provide a layer of communication protocols, together with Application Program Interfaces (APIs), that the enterprise business applications use to communicate. At present there are several widely used commercial middleware products that form the basis for the EAI industry.

The Under-the-Hood Layer

At the bottom is the network layer—the essential plumbing for transporting information from one point to another, both within an enterprise and across enterprises. The kind of networks we are talking about here first appeared in the 1970s. Before that, applications had no way to communicate with one another.

The network layer is the core of the information technology layer (IT layer) in an enterprise system. Newspaper articles refer to it as the "under-the-hood" part of an enterprise's IT system, or the Internet for that matter. It is generally looked upon as something the common man should not know about and certainly not tinker with—it is a source of many system problems. And when it collapses in one of many well-known or not so well-known ways, the whole system grinds to a halt. We often hear "The network is down." Network crashes can become a critical concern to the higher-level echelons in a distributed enterprise. And they certainly affect business. Spectacular network crashes make newspaper headlines—for example, when online stock-trading systems clog up and stop the customers from trading, or when hackers orchestrate a denial-of-service attack on Yahoo and other popular Web sites.

So the network layer has become the domain of a powerful new kind of expert, the specialist in network and IT systems management. Often this person has a title, like IT director. The job is to keep the network layer running at all costs—otherwise, the enterprise is out of business. It's the first point of pain for a distributed enterprise.

Consequently, network management products have proliferated. There are all sorts of tools for the IT director to use. They track resource consumption (CPU use and memory or disk space use) on every machine in the network and provide statistics and summaries in every color in the rainbow. They log alerts or warnings from network components such as routers and servers, slice and dice them by the minute or hour, raise the alarm, and page the IT director out of bed at any hour of the day or night.

But keeping the network running is not the IT director's only problem. Additionally, there are issues of security and privacy. And *cyber warfare* is an increasing headache for all enterprises. Everyone knows it is going to increase. Plenty of tools and products try to help the IT director solve cyber warfare problems, but again, they don't do too well.

The IT director's problem is making sense of all this network-level information. The importance of this task is well understood by enterprises today. They invest heavily in whatever technology is available to help. For example, the network operations room in a large global banking enterprise represents a billion-dollar investment these days.

1.3.2 Vertical Causality: Tracking Events up and down the Layers

Although there are no general layering standards for categorizing enterprise systems, the picture in Figure 1.4 is typical and captures the general idea.

- The applications that humans use and understand (sometimes) are at the top.
- Network plumbing that carries the messages and information is at the bottom.
- Several layers are in between, depending upon the complexity of the system.

Activity at each layer is translated into activities at the layers below and conversely. Those lower-level activities must complete successfully in order for the higher-level activities to also complete successfully.

So, we have a general principle of layered enterprise systems: An activity at the top *causes* activities at successively lower levels, which in turn *cause* other activities to happen at the top.

Layering adds another dimension to understanding events. Discovering the causal relationships across layers, between high- and low-level activities, especially in real time, is another outstanding issue in enterprise systems. We call this *vertical causality*.

An ability to track vertical causality in a layered system has at least two important uses in managing our enterprises. First, knowing how a business-level event is broken down by your system into sets of lower-level events is very helpful in understanding properties of that high-level event, such as its timing—for example, why it took so long or failed to complete—and how to improve performance at the business level. Sometimes we may only need to understand the breakdown of the business-level event into calls to applications, while at other times, we may need a breakdown all the way to the network level. This is the kind of information that might have helped

the IT manager answer the CEO's question about what caused customers in the New York area to complain about slow service.

Second, if we were able to track vertical causality, we could use it to group events at lower levels according to the high-level events they signify. That would help us make sense out of the mountains of low-level events that network monitoring tools give us.

Tracking vertical causality is not a simple problem because there is almost as much dynamism in the layers of our enterprise systems as there is in the communication between them across the Internet. For example, the ability of a service provider to load-balance incoming requests may depend on the loads of lower-level servers that are continually changing. A high-level complex event, such as defaulting on a service-level agreement, can happen in many different ways. It may result from lots of different sets of lower-level events. Moreover, our enterprise systems are dynamic in their composition and structure. The sets of components at each level are changing all the time, as are the kinds of activities the system is being employed to do. So the kinds of events that can happen at every level are changing too. Keeping up with what our systems do is not an easy task.

1.3.3 Event Aggregation: Making High-Level Sense out of Low-Level Events

What about those strategic-level complex virtual events that we talked about earlier? Virtual events are not actually generated in the system—nobody sends a message saying, "I violated law XYZ." The activities signified by high-level virtual events have to be recognized from among sets of events at the application layer or below. These policy and contractual events are top-level events of immediate interest to the corporate level of the enterprise.

We have just discussed the potential usefulness of an ability to track vertical causality. But there needs to be a high-level event in order to track its causes. Often, all there is at the highest levels is a perception, a suspicion, a list of complaints. But the business and strategic implications are very real.

The complementary operation to downward tracking of vertical causality is the aggregation of sets or groups of lower-level events into a single higher-level event that expresses the meaning of the lower-level events, taken together. These are some examples:

- A group of stock-trading events, related by accounts, timing and other data, taken together constitute a violation of a policy or regulation.
- A large set of network-level operations, originating from the same machine and repeating similar accesses on different target machines, may signify an attempt to intrude into a network.

- In an application service provider, groups of events signifying accesses to distributed applications, such as those illustrated in Figure 1.3, together may imply a violation of a service-level agreement with a customer.

Recognizing or detecting a significant group of lower-level events from among all the enterprise event traffic, and creating a single event that summarizes in its data their significance, is called *event aggregation*. The aggregate event will appear at a higher level in the enterprise's operations.

Event aggregation is a capability that needs to be developed. It is a very powerful technology for monitoring and managing our enterprises. It will in turn depend upon technology for recognizing patterns of events in large amounts of lower-level event traffic, in real time. And it depends first on an ability to express patterns consisting of multiple events together with their common data and timing. If event aggregation is implemented properly, it can give us the ability to track the lower-level events that were aggregated to create a high-level event—so-called *drill-down* diagnostics for tracking vertical causality.

1.4 The Gathering Storm of New Activities on the Web

There is already an urgent need for new technologies to help us understand the information in our IT systems, at every level from low to high. This need is going to become even more urgent. In the next three sections, we look at some of the newer activities hosted on the Internet.

We pick out three main themes among the activites now developing in the Internet world:

- Global electronic trade and the emergence of the *open enterprise*
- Agile systems—new kinds of highly flexible information-gathering systems
- Cyber warfare and cyber defense—the battle for the information society

Each of these themes, we shall argue, is making increasing demands for new technology for information gathering and delivery. Widely distributed fragments of information must be gathered and pieced together into forms appropriate for electronic processes to tackle the problems at hand, and for humans to understand high-speed business situations and stay in control.

Demands for new design technology also abound in each of the themes. We must be able to deal with increasingly complex systems, demands for high reliability, and frequent, rapid modifications. Demand for a design

technology that applies across the complete system lifecycle will appear in each of these themes, systems of electronic business processes that span multiple collaborating enterprises, agile information-gathering systems, and cyber defense systems.

1.5 Global Electronic Trade

The Internet is fast becoming the ether by which enterprises transact all forms of business. This is the next stage of Internet development. It goes beyond the electronic retailing and trading that we have all come to know, and love or hate, as the dot-com industry.

Global electronic trade involves collaboration between enterprises to transact business electronically on a scale unforeseen a few years ago. This is the *electronic collaboration* model of doing business. The stakes are high—reduced costs, lower inventory levels, faster time to market—all adding up to increased profitability and increased competitiveness.

The various visions and predictions for how this will happen can all be captured in the idea of the *global eCommerce Web,* shown in Figure 1.5. It is a grand scheme.

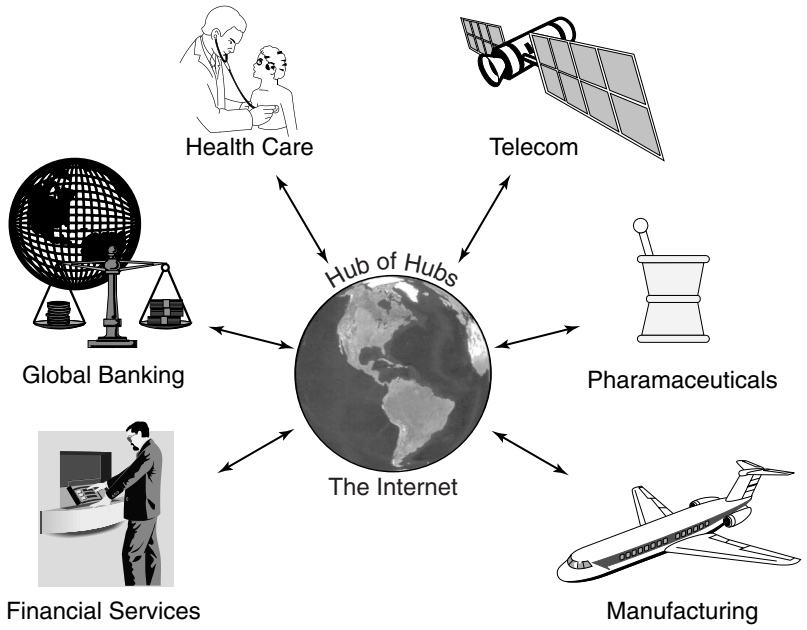

Figure 1.5: The global eCommerce vision

At present special Web sites called trading hubs, or eMarketplaces, are being developed for particular industries, such as retailing, automotive manufacturing, aerospace, financial products and services, telecom provisioning, and the like. Each of these trading hubs provide services that let enterprises in that industry collaborate by integrating various business processes—for example, supply chain processes between buyers of automobile parts and the suppliers.

A trading hub is set up to enable electronic commerce between enterprises in a particular industry. It provides standard formats for electronic trading documents used in an industry, and a range of services to support eCommerce between enterprises in that one industry. Services include, for example, demand forecasting for the industry, inventory management, partner directories, transaction settlement services, and so on. The role of the trading hub is to serve as an integration medium for the electronic management processes of enterprises that wish to collaborate in business.

In electronic commerce, trading documents are contained in messages that drive the steps in a trading process. There are standards for representing the trading documents within the industry. So the processes in each enterprise can easily understand documents from another process and respond intelligently by taking the next steps in a mutual trading process. The current trend is to base these standards on Extensible Markup Language (XML). Various services available on the hub may be supplied by different individual members. Then there are *content managers,* which are applications such as a distributed master catalog. Each participant manages its own content, and the master catalog keeps a constantly updated view of all the inventories of all participants in the hub. Anyone can get a globally consistent view of the "content" of the hub. Global content management is a critical facilitator to encouraging electronic trading agreements on the hub. The highest-level services are *transaction managers* of electronic business processes that automate the trading agreements between enterprises. These applications are process workflow engines that automatically initiate the next step in a trading process each time a message completing the previous step is received. Electronic trading across a hub can take many forms, from collaborative integration of individual business processes to auctions and exchanges of goods (electronic barter). The hub provides aggregation and settlement services.

The advantage of trading hubs is reduction in the costs of doing business. For example, procurement costs of a Fortune 100 manufacturer can run annually into billions of dollars. Just-in-time purchasing on the hub saves a significant percentage of these costs.

Now, the grand vision is that the trading hubs for all the various industries will be linked together across the Internet into the global eCommerce Web—that is, a global trading *hub of all hubs.* It will be event driven, of

course. As one visionary puts it:

> The traditional linear, one step at a time, supply chain is dead. It will be replaced by parallel, asynchronous, realtime marketplace decision making. Take manufacturing capacity as an example. Enterprises can bid their excess production capacity on the world eCommerce hub. Offers to buy capacity trigger requests from the seller for parts bids to suppliers who in turn put out requests to other suppliers, and this whole process will all converge in a matter of minutes.[1]

To what extent will this vision ever happen? It implies an awful lot of very complex high-level business events flying around, causally dependent upon sets of other high-level events. Will these global trading processes "all converge"? Or will there be a lot of timeouts and thrashing about with no convergence? This kind of real-time marketplace decision making will be driven by events. It is clearly going to need a new technology, beyond master catalogs and workflow engines, that enables us to detect event patterns that signify market situations of interest. These patterns will need to specify sets of business- or trading-level events and their data, timing, and causal relationships. What kinds of "situations of interest"? Here are some examples.

- *Risk mitigation.* Certainly, in the real-time marketplace, automatically recognizing and avoiding risky situations is going to be necessary. For example, an enterprise may want to abort a multistep purchase process in midstream if events happening in the hub's current context signify purchases by other parties that are likely to drive up the overall cost by the time the process has completed.
- *Policy enforcement.* Enterprises will need to maintain strict limitations on real-time trading processes because of the speed with which they could get out of hand. Each enterprise will have its own private policies. Policies could range from constraints on the data in events that are driving the trading process, such as "no parts manufactured in country XYZ," to avoidance of processes that can be indirectly influenced by competitors. The former kind of policy is a constraint on the data in events, while the latter may require detecting patterns of events happening in other trading alliances.
- *Opportunistic trading.* The flip side of risk mitigation is recognizing when trading processes should be speeded up or changed in midstream to take advantage of favorable situations. Again, such situations may

[1] Jay Marty Tennenbaum, EE 380 Seminar Lecture Series, Stanford University, May 2000.

often be signified by patterns of events, not directly related to the executing process, but happening in the global trading context.

- *Regulatory monitoring.* This is the same kind of issue as policy enforcement, but on a global scale as agreed to by all participants. Electronic trading will be subject to regulations placing limitations on trading processes, perhaps local to one industry's hub, or perhaps the result of international trade agreements across all hubs. It is well beyond the scope of this book to try predicting what the regulations of the global eCommerce hub might be. But what can be predicted is that the technology to monitor compliance will be event based, will involve recognition of complex patterns of events happening in the total global context, and will need to be flexible enough to be changed frequently without disrupting the monitoring functions.

Real-time marketplace decision making may well become a cybergame of cat-and-mouse, where the advantage goes to those with the best event monitoring and event pattern recognition capabilities. Players in such a game will need to analyze and adjust trading processes frequently. Tools to help analyze multilevel trading processes will need to track causal relationships between events, both horizontally and vertically. So we can expect that analysis tools incorporating causal tracking and complex event pattern detection will be used to monitor the business event layer and the layers below. This kind of monitoring and viewing activity (we will discuss "viewing" later) will need to have the same real-time scalability at higher levels that network-level tools have today.

1.6 Agile Systems

An *agile system* is defined as a system that is able to adapt rapidly to changes in its environment. What people have in mind, particularly the military, is large systems of small autonomous agents that share information across the Web and other networks. An agent is a small program or software object that peforms one task with the information it is fed, usually a stream of events, and passes on the results. The agents perform their individual tasks and are in constant communication with one another to achieve some overall objective—such as guiding a missile to its intended target or making a profit on arbitrage between several stock markets. As the environment changes, which may involve part of the system being destroyed, the community of agents adapts, spawns new members, and continues as best it can to achieve its objectives.

This kind of system model is appropriate not only for military command-and-control systems, but it could also be a model for business intelligence gathering, collaborative stock-trading strategies, and all manner of cyber warfare activities as well.

The major activity of agents in agile systems will be to process events fed to them from various sources, including but not limited to the Internet. In military systems like the one shown in Figure 1.2, other sources of event inputs will be special networks and sensors (radar, unmanned aerial vehicle [UAV], and so on). There will be a layered organization of communicating agents. Sets of agents will be organized into layers, with lower layers performing information gathering, filtering, and aggregating streams of events, and higher layers performing analysis and decision making. A second dimension of organization, orthogonal to layering, is also envisioned. Communicating agents will also be organized into enclaves or cells, to achieve specific goals and to improve the security and resilience to destruction of the system as a whole.

Agile systems are expected to deal with quite challenging situations, well beyond the scope of workflow engines and business process integration. Intelligence gathering requires the ability to correlate events from different domains of activity, such as manufacturing and transportation statistics on one hand and satellite image processing on another. Events may arrive from sources that are not normally associated as being related or relevent, and the events may arrive at the processing agents in a different time order from the order in which they actually happened. Furthermore, agile systems must make decisions based upon incomplete information. In the battlefield, information gets fragmented or destroyed en route. Nevertheless, global views of a situtation must be constructed in real time and actions taken.

If the agents are to be sophisiticated enough to be the building blocks of agile systems, they will need to be based upon a new technology of event processing. Each agent will perform one or more functions, such as recognizing a complex pattern of events, or aggregating lots of lower-level events that match a pattern into higher-level intelligence events, or detecting relationships between events that differ widely in their source and time of arrival at the agent. Agents will be needed that can perform these functions on any level of events, from low-level network events to high strategic-level events. So the underlying technology must apply generally across all layers of events, actual or virtual—a technology that can only process network-level events, for example, will not be adequate.

An agile military system is expected to carry out military doctrine, strategies, and processes, which may be dynamic and change on the fly during operations. The underlying technology must allow for continuous construction of new classes of agents that can recognize different patterns of events and perform new functions. It must enable us to easily add them into a running agile system without disrupting its operations. And it must let us build agile systems that have the ability to respond to certain situations by adding agents and reconfiguring themselves automatically.

The agile system vision has more of an artificial intelligence (AI) flavor to it than the global eCommerce vision. But it is a safe bet that AI will play a role in a successful solution to either vision.

1.7 Cyber Warfare and the Open Electronic Society

What would happen to us if our computer systems suddenly stopped working for a few days? Nobody knows! Of course, given the redundancy and tolerence to failures in our computer systems, that is very unlikely to happen. Or is it? It is quite surprising what havoc a rather unimaginative virus attack can cause. And hackers seem to be able to invade corporate Web sites and government computers almost at will. The reality is that we do not know what is happening in our information systems minute by minute or day by day.

Cyber warfare involves the development of new ways to defend our IT layers against an increasing set of criminal and destructive activities. The battleground is the Internet and every corporate IT layer. These are some of the current problems:

- Intrusions into computers. An intruder is someone who is not an authorized user and who anonymously gains privileged (or "root") access to a computer. The purposes for an intrusion vary widely and include the following examples:
  - Using a computer anonymously to set up a chat room for drug trafficking
  - Accessing databases (supposedly secure) to steal information such as credit-card numbers
  - Using a network server to launch another kind of attack at some other site in the network
- Denial-of-service attacks on Web sites and networks
- Computer viruses, propagated from machine to machine by file transfers and e-mail messages
- Spam, unsolicited e-mail, and other nuisance activities

Present security technologies such as encryption, firewalls, network-level intrusion detectors, and virus scanners defend against well-known, "textbook," criminal activities on the Web. When a new kind of attack is discovered, the IT security managers scramble to include it in their log file search scripts, and the security tools manufacturers try to add a defense against it to their products. But new tricks and techniques to exploit vulnerabilities in the defenses are being devised all the time. This kind of crude

rearguard response we have at the moment—fix the defenses *after* we know what happened—is totally inadequate for dealing with the coming wave of electronic crime.

It is a total myth that it takes an expert or genius to break into a computer. Breaking and entering an operating system is something that's taught nowadays—it is a very good way to learn how operating systems and network protocols work. There are chat rooms on how to break in and scripted attacks that can be downloaded. The majority of those breaking into systems do not understand how their exploits work any better than a clerk understands how the operating system running his inventory program works.

When a new kind of intrusion is devised, it does take an understanding of the detailed workings of an operating system, or a network protocol, or an application program. From then on, that method of intrusion can be applied using scripts that carry out the steps of the method automatically.

Experts often publish new attacks on the Web to encourage manufacturers to plug the holes in their operating systems and application programs. But of course, the "crackers" and "kiddies" can read the expert's techniques too! And even when holes are plugged by the manufacturers, the patches are often not downloaded to many of the computers on the network. A vulnerable computer is usually the weak link into a subnet or a whole IT layer.

At present it is generally admitted by anyone in the know, including those who build or manage IT networks or advise on Internet policy, that our present abilities to defend our IT infrastructure against attacks, both old and new, are dismal. Let's list a few of the problems.

1. We don't have powerful enough technology, even at the low level of network events, to detect nefarious activities in real time and cut them off. And new ones are springing up all the time.
2. At the network level, the current crop of detectors have absurdly high false alarm rates. We can't correlate the outputs from various network-level detection products to get a more accurate picture of whether or not an intrusion attempt is taking place. Correlation of the various detectors produced by the security industry would need a lot of planning and agreements. First, we need standards and conventions for the formats of detector outputs, so-called alerts. This first-step problem has been recognized by the Internet Engineering Task Force (IETF) and other organizations. We may have some standards for alerts and alarms soon. Then we need event processing technology beyond what is presently available from the security industry to do real-time correlation of alerts.
3. Some kinds of attacks are difficult, if not impossible, to detect at the network level. They would be easier to detect if we had application-level

monitoring—but we don't. We don't have adequate monitoring to provide defensive tools with the kind of information they need. To do this, we need detectors that work at all levels of an IT system.

4. New Internet applications and activities, such as global eCommerce, for example, present new opportunties for electronic theft and destruction. We need technologies that enable us defend at the highest levels in enterprise operations. Just imagine being able to mess with your competitor's electronic trading processes—insert a few false events supposedly coming from a partner at the "right" time, or intercept and divert process events between trading partners, or relay process events to competitors.
5. We, as a society, have become far too dependent upon the Internet. Most of our essential services, from telecommunications and financial transactions to food distribution, now operate over the Internet and IT layers that are accessible from the Internet. In many cases there is no backup mode of operation, should the Internet be taken down. A denial-of-service attack on the Federal Reserve's Fedwire funds-transfer system could, if successful, bring the banking system to a standstill.
6. Traditional security technology—firewalls, for example—and open eCommerce are in direct conflict.

The first four problems all indicate the need for event processing technology that applies across all levels of enterprise operations and applies uniformly to a wide range of the security industry's products. The fifth problem is a political and national policy problem, beyond the scope of this book.

But the sixth problem is one of the most perplexing problems with the Internet explosion—that is, the tension between traditional security based upon locking things up and denying access (firewalls, virtual private networks, and so on) and the need for open commerce. Industry analysts have written report after report on this problem, but the technology to allow openess, whereby anyone can communicate with anyone else and still have secure operations, just doesn't exist.

The sixth problem could become the great stalemate in developing visions like the global eCommerce vision. Unfortunately the "real" problem is not the sixth problem itself, but the lack of attention to it by the developers of grand visions. They are focused on making the vision happen. I have no doubt that the sixth problem can be solved. The question is whether a solution must be part of the infrastructure of the grand vision from the earliest designs, or whether it can be added piecemeal afterward when the cyber warfare really begins—rearguard actions all over again, but the stakes will be higher than ever.

There may be a lesson to be learned from the DARPAnet, the precursor of the Internet, which was built on open principles and little thought about misuse. The focus was on getting distributed communication networks to work. Could the DARPAnet have been designed differently, in hindsight, to make our present security problems easier to track, compartmentalize, and handle? Some experts feel that authentication should have been included in name servers from the very beginning, and that the messaging protocols should have been designed to always indicate the actual source of any message, no matter how many "hops" it makes in getting to its destination.

Following up on this early DARPAnet lesson, should the IT systems of the future be built with monitoring facilities and event feeds at every level from the network to the strategic level? And should they have backup duplication of essential monitoring and tracking facilities? This will be expensive, just like having security checkpoints at airports today. But it may become part of the cost of running the information society.

1.8 Summary: Staying Ahead of Chaos

The recurring theme of this chapter is that our event-driven global information society is heading into a lot of trouble if we don't pay attention to tackling and solving a number of related problems.

- Monitor events at every level in IT systems—worldwide and in real time.
- Detect complex patterns of events, consisting of events that are widely distributed in time and location of occurrence.
- Trace causal relationships between events in real time, both horizontally within a level of system activity and vertically between high and low levels of system activity.
- Take appropriate action when patterns of events of interest or concern are detected.
- Modify our monitoring and action strategies in real time, on the fly.
- Design our systems to incorporate autonomous processes for applying levelwise event monitoring and viewing and for taking appropriate actions.

What we are talking about here is developing an *event processing technology* applicable to every level of system operations in all our IT layers. Such a technology is a key foundation to managing the electronic enterprise and is critical to developing capabilities to defend our IT systems.