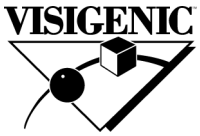


DISTRIBUTED OBJECT COMPUTING IN THE INTERNET AGE

FEBRUARY 1997

VISIGENIC SOFTWARE, INC. 951 MARINER'S ISLAND BLVD. SAN MATEO, CA 94404 TEL: 415-286-1900 FAX: 415-286-2464 WEB: WWW.VISIGENIC.COM

DISTRIBUTED OBJECT COMPUTING IN THE INTERNET AGE

Visigenic Software is helping to create the foundation on which tomorrow's mission-critical applications will be built—an open, distributed, object-based architecture for the new global enterprise. The leading supplier of embedded distributed object technology to the software industry, Visigenic develops pioneering products that enable IT organizations to protect their investments in legacy applications while moving to distributed object computing and embracing the new opportunities presented by the Internet.

In this white paper, we discuss recent trends in distributed object and Internet computing technologies. We provide a brief overview of the Common Object Request Broker Architecture, an infrastructure for distributed applications based on distributed objects, and the Internet Inter-ORB Protocol, the CORBA protocol that is emerging as the business application messaging standard for the Internet. We give a technical overview of Visigenic's VisiBroker Object Request Brokers—the ORB implementation adopted by Netscape Communications Corporation, Oracle Corporation, Hitachi and other major corporations— and review the advantages of developing distributed applications with these industry-leading products.

Distributed Object Computing in the Internet Age

Distributed Object Computing: A Challenge for the '90s

As the old adage goes, "The more things change, the more they stay the same." Businesses in the '90s face many of the same challenges they have faced for decades. Survival or success requires continual improvements in customer service and product quality. There is perpetual pressure on the bottom line. Technology plays an increasingly important role in creating a competitive edge. Business cycles continue to shorten. These realities translate into mandates for Information Technology (IT) organizations. Rapid response is critical to the success of the enterprise. For IT, however, the landscape is filled with obstacles that make rapid response difficult.

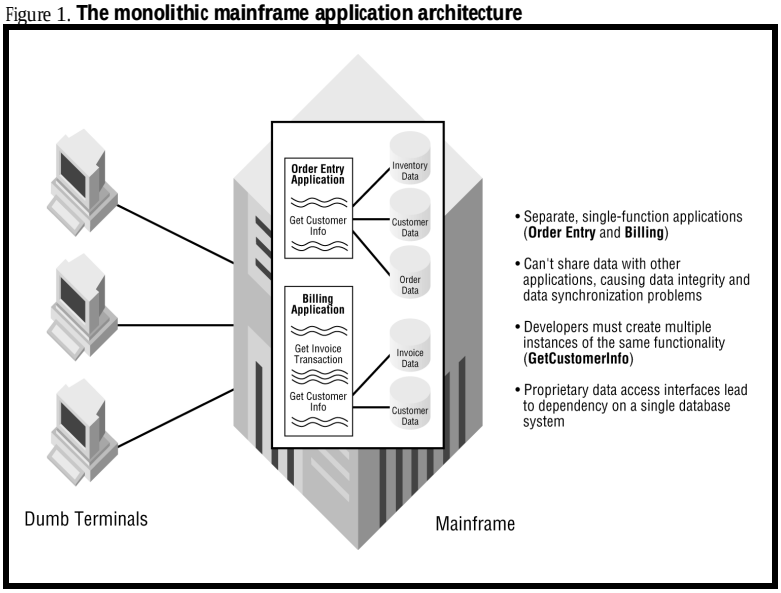
In the latter half of the '90s, most central IT organizations are under considerable pressure to deliver more value at lower cost. Most are struggling to manage complex, heterogeneous environments that incorporate disparate hardware, software, applications, networks, and database systems. Many must integrate legacy systems originally developed ten or twenty years ago with the latest client/server and Internet technologies. Driven by the need to deliver faster and more cost-effectively, many IT organizations have had to adopt a tactical, short-term approach and, as a result, are finding it increasingly difficult to be proactive in setting the long-term technical direction for the enterprise.

How Did We Get Here?

The last 15 years have seen much change in how we design, develop, and maintain enterprise information systems. This period began with monolithic mainframe systems. Each of these systems contained all of its own presentation, business, and data access logic. They could not

share data with other systems, and so each had to store a private copy of its data. Because different systems needed access to the same data, organizations had to store redundant copies on multiple systems.

Figure 1. **The monolithic mainframe application architecture**

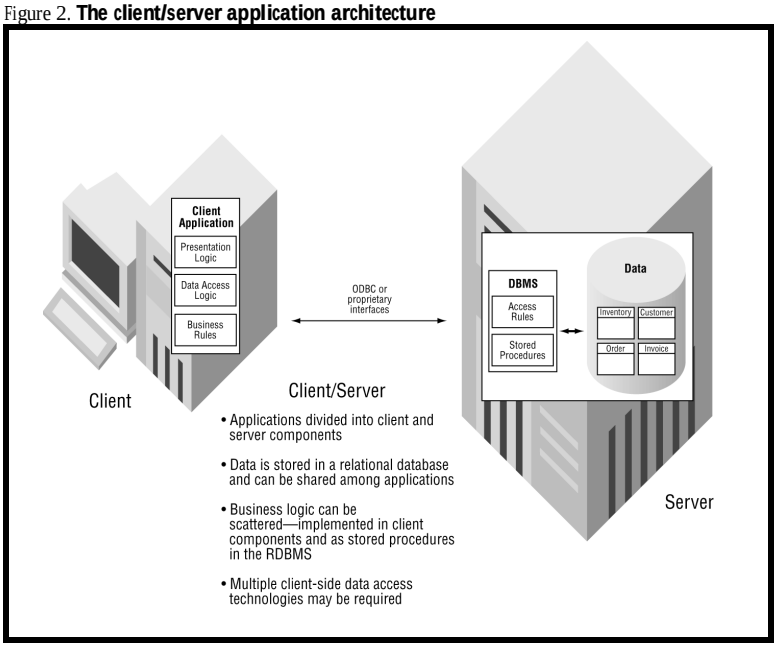


These monolithic applications were inefficient and costly, and they soon gave way to relational database technology and the client/server model. Made possible by a convergence of technologies—networks, low-cost PCs, graphical user interfaces, and relational databases—client/server computing promised to simplify the development and maintenance of complex applications by separating centralized, monolithic systems into components that could be more easily developed and maintained.

Applications were partitioned into client components, which implemented the application's presentation logic and contained much of the business logic, and server components containing business logic in the form of stored procedures. Data access logic was handled either by the client or the server, depending on the implementation strategy. In the end, many client/server solutions simply created two monolithic systems where there once had been one. Today, it remains difficult to build, maintain, and extend mission-critical client/server applications.

Throughout this period, development teams have had to create the same functionality over and over again. Reusing code was difficult. Usually it meant copying a segment of code, modifying it, and then deploying the modified copy. Over time, a proliferation of similar modules had to be updated and maintained separately. A change to one module had to be propagated to similar modules throughout the enterprise. Because this often proved unmanageable, functional inconsistencies crept into enterprise information systems.

Figure 2. **The client/server application architecture**



The Distributed Object Revolution

Distributed object technology fundamentally changes all this. Coupled with a powerful communications infrastructure, distributed objects divide today's still monolithic client/server applications into self-managing components, or objects, that can interoperate across disparate networks and operating systems.

The component-based, distributed-object computing model enables IT organizations to build an infrastructure that is adaptive to ongoing change and responsive to market opportunities. In the age of global competition and increasingly narrow market windows, companies that can initiate rapid change—not just respond to it—are better prepared to capitalize on opportunity, and are more likely to succeed.

"By 1997/98, object technology will become the predominant paradigm for conceptualizing, building, and using applications."
— The META Group

Distributed applications provide an opportunity to establish and maintain a competitive advantage by creating a flexible IT infrastructure. However, they also bring new requirements. To operate in today's heterogeneous computing environments, distributed business applications must work on a variety of hardware and software platforms. They must integrate old technology with new and make use of existing infrastructure. Furthermore, suitability for enterprise-class applications calls for capabilities beyond conventional Web-based computing—scalability, high availability, ease of administration, high performance and data integrity.

Since 1989, the Object Management Group, a consortium of platform vendors, ISVs and end-users, has been busy specifying the architecture for an open software bus on which objects written by different vendors can interoperate across networks and operating systems. The OMG is the world's largest software consortium, with over 700 member organizations, and its specifications enjoy broad support throughout the standards community and the software industry. The

International Standards Organization (ISO) endorses OMG's standards. OMG standards typically are also endorsed by X/Open as part of the Common Application Environment (CAE) specification.

The CORBA Specification

The focus of the OMG's effort has been to specify the Common Object Request Broker Architecture (CORBA). The CORBA specification describes a software bus, called an Object Request Broker (ORB), that provides an infrastructure for distributed object computing. It enables client applications to communicate with remote objects, invoking operations either statically or dynamically. In late 1994, the OMG approved the CORBA 2.0 specification, which includes a protocol for ORB interoperability called the Internet Inter-ORB Protocol (IIOP).

A recent survey of IT professionals "concluded that 57.9% are currently developing, or plan to develop mission-critical object-oriented applications over the next 12 months."
— The Standish Group (9/96)

IIOP is robust, scalable, and transaction-oriented. It runs on top of TCP/IP, requires no special configuration, and is fast becoming the standard for communication between and among distributed objects running on the Internet and enterprise intranets. Leading vendors of Internet tools, including Netscape and Oracle, have fully embraced IIOP as the basis for future product offerings. CORBA 2.0 compliant objects are fully interoperable because they use IIOP to communicate.

CORBA Services

The ORB backbone is extended with modular, add-on, system-level services that complement the functionality of the ORB and provide building blocks for business applications. The OMG has defined a set of common object services, including:

- A *naming service* that enables objects to find each other by name
- An *event service* that enables objects to subscribe to an event channel and be notified of specific events
- A *transaction service* that defines transactional disciplines, coordinating two-phase commits among objects
- A *security service* that provides authentication, authorization, encryption and auditing functions to protect sensitive data and to control user access to applications and services

CORBA services provide functionality that is essential for many enterprise applications. Since developers do not have to implement these core functions in each system, they can focus instead on implementing their own application and business logic.

Benefits of CORBA

The OMG and its members envision a near-term future in which software objects with defined interfaces interoperate across corporate intranets and across the Internet. The benefits for application developers and IT organizations are significant:

- Choice. CORBA is an open solution based on a published specification. It is implemented and supported on a wide variety of hardware and operating system platforms.
- Plug-and-Play Flexibility. A CORBA-compliant software object has a defined interface. All communications go through that interface. Changes to the implementation of the object do not affect other objects, so long as the object's interface remains the same. Devel-

opers can code to the defined interface knowing that modifications will not break other parts of the distributed application.

- COEXISTENCE WITH EXISTING SYSTEMS. ORB technology protects your investment in existing systems. A legacy application, module, or entry point can be encapsulated in a C++ or Java "wrapper" that defines an interface to the legacy code. Creating such an object wrapper gives the legacy code a CORBA-compliant interface, making it interoperable with other objects in a distributed computing environment.
- INTEROPERABILITY. CORBA-compliant software objects communicate using the Internet Inter-ORB Protocol (IIOP) and are fully interoperable, even when they are developed by different vendors who have no knowledge of each other's objects. Software bridges enable communications between CORBA-compliant objects and objects developed with Microsoft's ActiveX/DCOM technology. Enterprise IT organizations can select an ORB, CORBA services, and software objects based on the functionality they provide—even if they were developed by different vendors.
- PORTABILITY. Software objects that comply with the CORBA standard are portable. That is, objects built on one platform can be deployed on any other supported platform.

"On average, 70% of the work for developing new applications is spent on building infrastructure, with only 30% focusing on the specific needs for the application to fulfill. A standards-based distributed computing model, such as CORBA, solves this problem by providing a reusable infrastructure that allows developers to focus on the specific business problems at hand."
— The Standish Group

THE INTERNET EXPLOSION

As the CORBA 2.0 specification was being developed and finalized, the Internet and the World Wide Web began the phenomenally rapid expansion that continues unabated to this day.

Transcending its roots in government agencies and educational institutions, the Internet has become the most significant new medium for communication between and among businesses, educational and government organizations, and individuals. Growth of the Internet and enterprise intranets will continue at a rapid pace through at least the end the century, leading to global interconnectedness on a scale unprecedented in the history of computing.

KEY INTERNET TECHNOLOGIES

Several Internet-related developments are particularly significant for enterprise IT and are playing a key role in enabling IT organizations to benefit from distributed-object technologies:

- JAVA. An object-oriented programming language developed by Sun Microsystems. The widespread adoption of the "write once, run anywhere" Java language makes it possible to develop truly platform-independent client applications.
- JAVA APPLETS. A small program, written in Java, that travels from the Web server to the requesting client where it then executes. An advantage of Java applets is that IT has more control over what version of an applet runs on any client environment, thereby eliminating the version incompatibilities that exist where application software must be loaded and updated on each client machine. Java applets allow "thin" clients, which are not loaded with large, difficult-to-maintain application software.
- "SMART" WEB BROWSERS. Web browsers on millions of desktop computers can now run applets that communicate and interact

with applications running on remote systems elsewhere on the Internet.

"We expect most organizations will realize the impracticality of trying to converge on a single technical architecture, and will recognize that a mixture of technical architectures will be the norm through 2000 and beyond. For most organizations, this realization will become apparent with the arrival of the Internet."
— The META Group

- NETWORK COMPUTERS. Network computers—like Sun's JavaStation, the "zero administration" NetPC from Intel and Microsoft, and Oracle's NC—promise dramatically lower costs for organizations whose users access server-based software objects from downloadable applets running in a Web browser. (The Gartner Group has estimated the five-year cost of owning and supporting a single Windows PC at $60,000, up 100 percent since 1988; Oracle claims that system administration costs will be 70 percent lower for network computers.)
- SECURE FIREWALLS. Firewalls now provide security for enterprise information systems, making it possible for applications behind the firewall to interact safely with applications running on systems outside the firewall.

In the Internet age, it is no longer possible to think of creating homogeneous computing environments. Enterprise information systems *must* be able to communicate and interoperate with applications and systems outside the firewall. In this way, enterprise IT can leverage the Internet by using it as a *de facto* WAN that links government agencies, businesses, educational institutions, and individuals worldwide.

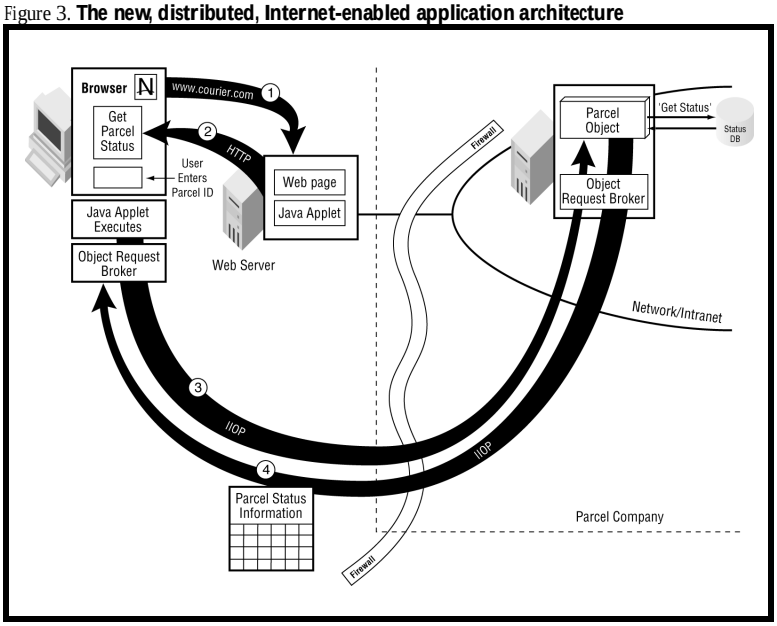
A NEW APPLICATION PARADIGM FOR THE INTERNET AGE

The concurrent revolutions in Internet and distributed object computing are on converging paths. As a communications framework, the Internet provides an ideal platform for distributed object applications and therefore fosters their growth. At the same time, distributed object technology is improving the quality of Web-based applications, adding value to the Internet and enterprise intranets.

This symbiotic relationship is creating a paradigm shift in the way we conceptualize, design, develop, deploy, and maintain business applications. In the age of the Internet, a new kind of application will predominate—an application built from new and legacy code encapsulated in software objects, running on systems both inside and outside the firewall, and interacting with each other through an Object Request Broker using defined interfaces.

How does this new application work? Figure 3 shows an example—a distributed, Web-based parcel tracking application. The application lets users check the status of a delivery by entering a parcel tracking number. The user enters the tracking number in a form displayed by a Java applet that runs in the user's Web browser. The Java applet uses the Object Request Broker to communicate with a software object running behind the package delivery company's firewall; this object queries the company's database and returns status information through the ORB to the Java applet, which displays the information for the user.

Figure 3. **The new, distributed, Internet-enabled application architecture**



The parcel-tracking application works as follows:

1. In a Web browser, the user enters a URL that points to a page on the package delivery company's Web server maintained outside the enterprise firewall.
2. The Web server receives the user request and uses HTTP to return a Web page to the client's browser. The Web page contains the Java applet that makes up the client component of the distributed parcel tracking system.
3. With the Java applet running, the user enters a parcel tracking number, and clicks the "Get Parcel Status" button displayed by the Java applet. The applet sends an IIOP message through the Object Request Broker to a server-side software object (Parcel), invoking the object's GetStatus method and passing the user's parcel tracking number. The Parcel object queries the enterprise database for the status of the user's parcel.
4. Upon receiving query results from the database, the Parcel object returns the results to the client-side Java applet through the Object Request Broker. The applet receives the results and displays them in the user's Web browser.

The client might be a PC running Windows, a Macintosh, a workstation, a network computer, a handheld personal computer, or even a set-top box such as WebTV. Server objects may be written in Java, C++, COBOL, SmallTalk, or other languages and may include new and legacy code. Objects do not have or need information about each other's implementation details. They communicate only through their defined interfaces. In the background, transparent to the user, the Object Request Broker manages the communications among the various objects that make up the distributed application. Because they communicate with the objects via IIOP, clients are able to invoke business functionality directly, without going through Web server software and the delays inherent in processing through a CGI programmer script for each user access.

"Today, there is little doubt that conducting business over the Internet is the direction the world is taking."

— The Standish Group

Implications for the Enterprise IT Organization

The new application paradigm has much to offer enterprise IT. Organizations that move quickly to adopt distributed object computing and embrace the Internet will achieve a substantial edge over their competitors. The benefits include:

- Flexibility to mix and match interoperable software objects from multiple vendors
- Centralized management and administration of software objects
- Lower costs and shorter development schedules resulting from reusable objects and simplified integration of legacy code
- A dramatic reduction in the costs for acquiring, configuring, and maintaining client systems, brought about by storing applications centrally
- The stability of widely supported industry standards, such as CORBA and IIOP, that ensure interoperability, portability, and a wide choice of object vendors

CORBA is simply a specification for a software bus. The suitability of a CORBA-based application for enterprise computing is governed by the implementation of the ORB. Because of their design and completeness, Visigenic's VisiBroker products are recognized as the leading solution for distributed applications. After comparing ORB products, the industry's leading technology vendors—including Netscape, Oracle and Hitachi—have selected VisiBroker for their implementation of CORBA and IIOP.

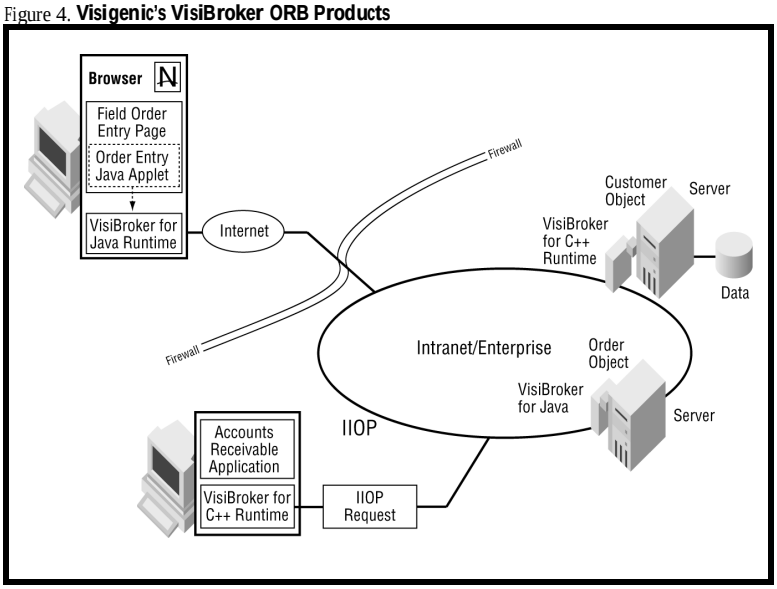
The VisiBroker Solution

Visigenic's implementation of the CORBA specification addresses the requirements of organizations wishing to deploy enterprise-class distributed business applications. Visigenic's VisiBroker product family allows systems developers—both commercial solution providers and enterprise IT organizations—to develop, deploy, and manage distributed object applications. The VisiBroker product family includes:

- **VisiBroker for C++** is a complete CORBA 2.0 compliant Object Request Broker runtime and supporting development environment for building, deploying and managing distributed C++ applications that are open, flexible, and interoperable across multiple platforms.
- **VisiBroker for Java**, the first client- and server-side CORBA 2.0 compliant Object Request Broker written completely in Java, is a runtime and supporting development environment for building distributed Java applications for the Internet and intranets. VisiBroker for Java can be used for client applets that run in Java-enabled browsers such as Netscape Navigator and Microsoft Internet Explorer, as well as objects that reside on servers.
- **VisiBroker Naming Service for C++ and Java**, a full implementation of OMG's CORBA Naming Service specification, enables developers to register object names at runtime. Naming contexts are organized into a hierarchical namespace. Client applications can then use the Naming Service to discover the names and object references for objects they wish to use by simply traversing the naming context hierarchy.

- **VisiBroker Event Service for C++ and Java**, a full implementation of the OMG's CORBA Events Service specification, allows "supplier objects" to communicate via an *event channel*, notifying any number of "consumer objects" when events they "subscribe to" take place. This supplier-consumer model reduces server traffic and improves scalability over the alternative of a polling scheme.

Figure 4. **Visigenic's VisiBroker ORB Products**



Software objects built with VisiBroker can be accessed by Web-based applications that communicate using IIOP, including Java applets developed with VisiBroker for Java. VisiBroker's native implementation of IIOP enables truly interoperable distributed applications for the Internet and enterprise intranets.

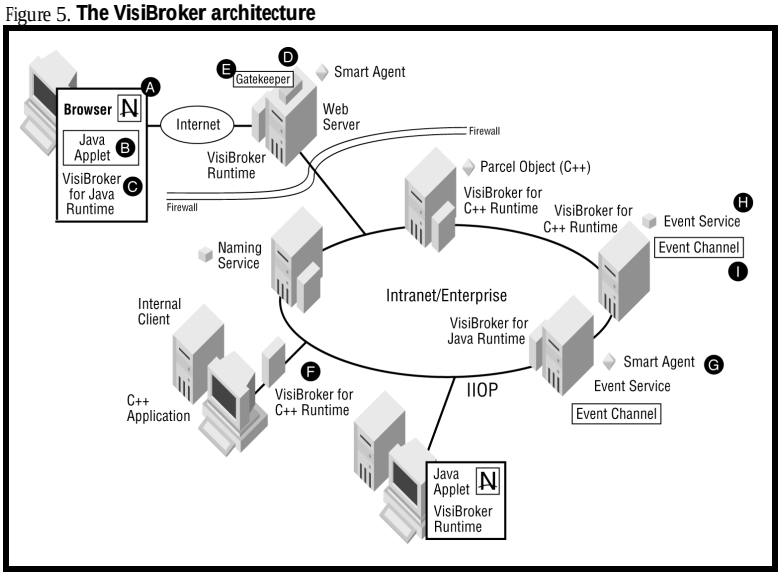
The VisiBroker ORB products were originally developed by PostModern Computing, which marketed them as ORBeline and BlackWidow. PostModern Computing had long been considered a technology leader in the ORB market. Visigenic merged with PostModern in May 1996 and reintroduced the products as VisiBroker for C++ and VisiBroker for Java.

"VisiBroker for Java is a technology leader in the brand new field of Java ORBs."
— The META Group

VisiBroker Architecture

Visigenic's VisiBroker products establish a communication infrastructure for enterprise-class, distributed-object applications. Figure 5, based again on the example of a parcel tracking application, illustrates the VisiBroker architecture and its software components.

Figure 5. **The VisiBroker architecture**



The VisiBroker architecture comprises the following components:

A JAVA-ENABLED WEB BROWSER. The Java applet using VisiBroker runs inside any Web browser that supports Java.

B JAVA APPLET. The client component of the application is a Java applet that is delivered by the Web server and runs in the user's Web browser.

C VISIBROKER FOR JAVA RUNTIME. The VisiBroker for Java runtime is embedded in the browser (such as Netscape Navigator 4.0) or accompanies the downloaded client applet.

D WEB SERVER. The Web server sits outside the firewall. It feeds the Java applet to clients and hosts the Gatekeeper.

E GATEKEEPER. The Gatekeeper runs on a Web server outside the firewall. The Gatekeeper enables client-to-object communication through firewalls that do not accept IIOP messages. It also allows clients to "bind" to objects on servers other than the server from which the applet was downloaded.

F VISIBROKER FOR C++ RUNTIME. The VisiBroker for C++ runtime enables objects written in C++ to be accessed by other objects written in C++, Java, Smalltalk and other languages.

G SMARTAGENTS. SmartAgents are components of the VisiBroker runtime. One or more SmartAgents running on the enterprise intranet dynamically locate requested objects in the distributed object environment. This dynamic object location capability gives VisiBroker significant flexibility and ease-of-administration.

H VISIBROKER EVENTS SERVICE. The VisiBroker Event Service enables objects to notify any number of clients and other objects when significant events take place.

I EVENT CHANNEL. The event channel component of the event service receives information from supplier objects and distributes it to consumers (objects and clients interested in the new information). This asynchronous interaction reduces server traffic and increases system scalability.

VisiBroker: A Better ORB by Design

VisiBroker delivers a robust, high-performance implementation of the CORBA 2.0 specification, making it the ideal choice for enterprise computing environments. Its strengths include:

- High Availability and Load Balancing
- Interoperability
- Ease-of-Administration
- Scalability and Performance
- Suitability for the Internet

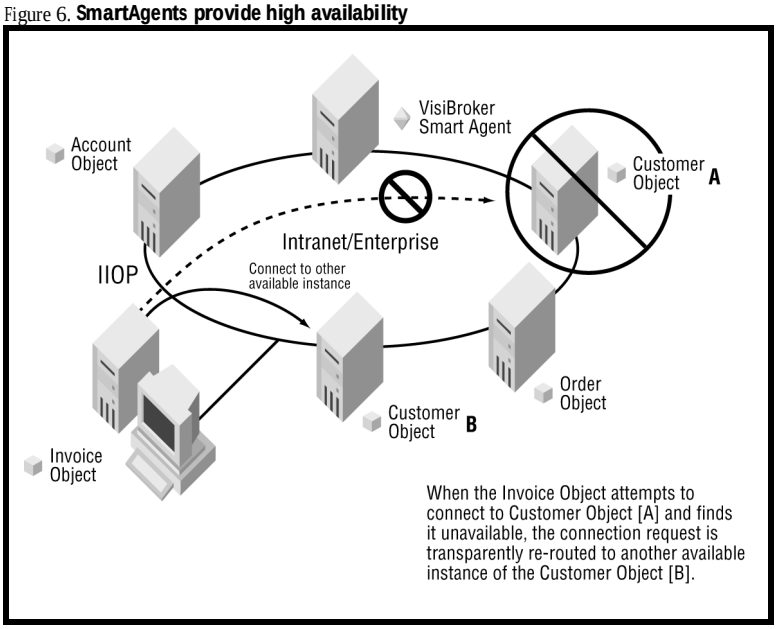
High Availability and Load Balancing

VisiBroker provides high availability and load balancing through its agent-based architecture, which is implemented as two software components: SmartAgents and Object Activation Daemons.

The SmartAgent provides a distributed directory service that dynamically tracks objects that are installed or active in the VisiBroker ORB. Server objects register with the SmartAgent at startup. A client application seeking to communicate with a server object of a particular type obtains an *object reference* from the SmartAgent and thereafter uses the object reference to communicate directly with an instance of the server object.

VisiBroker also features an Object Activation Daemon, which runs on each machine that hosts server objects. It cooperates with the SmartAgent to ensure that server objects are activated on demand and are shut down when they are no longer needed. Use of the Object Activation Daemon is optional.

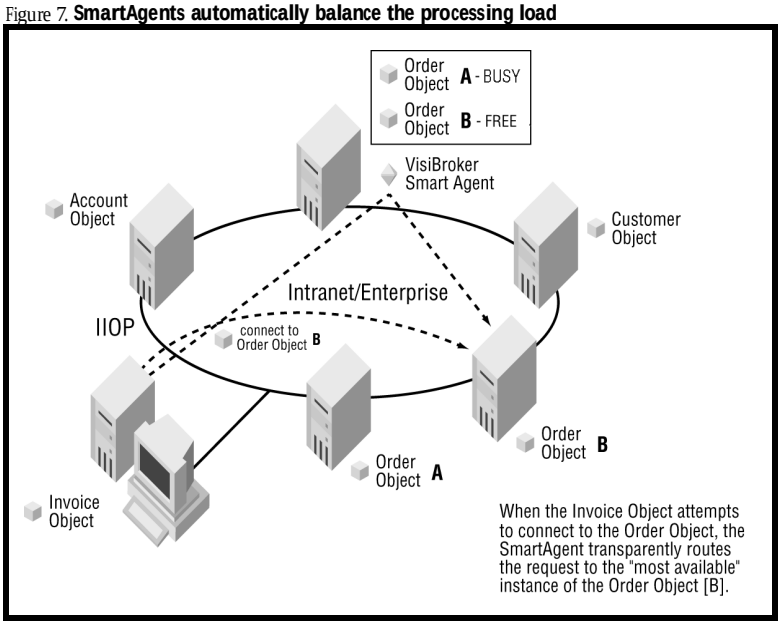
Figure 6. **SmartAgents provide high availability**



High Availability. It is sufficient to run a single SmartAgent on a given local area network. However, running multiple SmartAgents provides higher availability, with no additional effort on the part of the application developer.

- Multiple SmartAgents cooperate with Object Activation Daemons to ensure uninterrupted service even if the SmartAgent or a server object becomes unavailable during processing. If the host running a SmartAgent becomes unavailable, objects registered with that SmartAgent are re-registered with another available SmartAgent. This is transparent to the applications.
- If a server object becomes unavailable, its clients cooperate with the SmartAgent to reestablish a connection, whether this means locating another instance of the server object or using the Object Activation Daemon to start a new instance. In either case, service to clients proceeds uninterrupted.

Figure 7. **SmartAgents automatically balance the processing load**



LOAD BALANCING. Load balancing is accomplished using a simple round-robin load balancing algorithm. SmartAgents allocate client location requests to multiple object instances, ensuring that no single object instance becomes overloaded and providing a consistent level of service to client objects. With the 3.0 release of VisiBroker, more sophisticated load balancing algorithms can be added by the application developer.

INTEROPERABILITY

Applications developed with VisiBroker are interoperable with other CORBA 2.0 compliant objects, with Web-based applications, with legacy applications, and with ActiveX/DCOM objects.

WITH OTHER CORBA OBJECTS. VisiBroker provides a complete CORBA 2.0 Object Request Broker that uses IIOP as its native communications protocol. Unlike other ORBs, VisiBroker has no proprietary internal protocol and does not require a gateway or a translator protocol. Objects developed with VisiBroker are interoperable with other objects running on a CORBA 2.0 compliant ORB. It should be noted, however, that objects managed or accessed by an ORB other than VisiBroker would not be participants in VisiBroker functionality, such as the high availability made possible by the SmartAgent architecture.

WITH OTHER LANGUAGES. VisiBroker enables objects written in C++ or Java to be accessed by other objects written in other languages, including C++, Java, Smalltalk and others.

WITH LEGACY SYSTEMS. Objects created with VisiBroker can easily be integrated with legacy systems through object wrappers. Object wrappers define object-oriented interfaces for legacy applications and enable them to function as software objects that communicate with other objects through the ORB.

WITH WEB-BASED APPLICATIONS. Applications created with VisiBroker for Java execute in any Java-enabled Web browser. Netscape has embedded the runtime component of VisiBroker for Java in version 4.0 of its Navigator product and in its SuiteSpot server products. For users of other Java-enabled browsers, such as Microsoft Internet Explorer, the runtime module is automatically downloaded when the user accesses a VisiBroker for Java applet. VisiBroker development software will also be bundled with Netscape SuiteSpot and FastTrack server software.

"Enterprise customers want to leverage Intranets to build a new generation of network applications. By integrating the Visigenic ORB in Netscape Enterprise Server, Netscape is making it easier for developers to leverage the thousands of CORBA applications in place today to build multi-tiered distributed applications."

— Srivats Sampath

Vice President, Server Product Marketing

Netscape Communications

WITH OTHER OBJECT MODELS VIA VISIBRIDGE. Visigenic has developed a software bridge that enables communication between COM/DCOM objects and CORBA objects. VisiBridge allows ActiveX controls implemented in Web pages, Visual Basic applications, or OLE-enabled applications to interoperate with CORBA objects. Developers using ActiveX-enabled tools or ActiveX-enabled development environments can build applications that transparently access CORBA objects over the Internet, intranets or within enterprise networks without having to know CORBA.

EASE OF ADMINISTRATION

The VisiBroker ORB products are easy for system administrators to manage. VisiBroker is a dynamic system. Each object registers with a SmartAgent automatically at startup. Clients find objects dynamically by querying the SmartAgent.

EASIER FIREWALL ADMINISTRATION. VisiBroker also simplifies firewall administration. Typically, firewalls are configured to permit HTTP traffic and block messages sent using other protocols, including IIOP. When communicating through a firewall that is not configured for IIOP traffic, VisiBroker automatically wraps IIOP messages in HTTP packets. This ensures that software objects can communicate through the firewall without any special firewall reconfiguration.

NO CONFIGURATION FILES. With VisiBroker, there are *no* configuration files. Other CORBA ORBs require that a software daemon be installed and configured on every node that needs access to the ORB. VisiBroker is a completely dynamic system. It eliminates the overhead required to install and manage components on all client and server systems.

MONITORING AND MANAGEMENT. The 3.0 version of VisiBroker includes features that facilitate monitoring and management of the ORB and the implementation repository.

- The *Interface Repository Manager* displays in hierarchical form all known interfaces for objects and data structures that the objects might use.

- The *Implementation Repository Manager* provides a visual interface to Object Activation Daemons.
- The *Location Service Manager* provides a visual interface to SmartAgents.
- The *Name Service* Manager delivers a visual interface to the naming service and enables administrators to manage this CORBA service more effectively.

Scalability & Performance

Distributed object applications must be highly scalable. VisiBroker's architecture provides for high performance and scalability as a result of its multi-threaded architecture, connection management facilities, SmartBinding, dynamic object creation, event handing, and use of IIOP.

Multi-threaded Architecture. VisiBroker is based on a multi-threaded architecture that ensures efficient use of system resources. It offers a choice of server-side thread management models: *thread-per-session* and *thread pooling*. With the thread-per-session model, a worker thread is created for each incoming connection request. The worker thread performs all tasks originating from the client and exits when the connection is closed. With thread pooling, a pool of allocated threads is available to service client requests. Because threads are reused, overhead is further reduced.

VisiBroker offers superior connection management. Should a client request connections to multiple objects residing on the same server, the requests are multiplexed over a single network connection. Developers can also streamline system traffic by limiting the number of connections for each server object. When the connection limit is reached, connections that have gone the longest without use are broken before new connections are created. This allows large numbers of clients to access the service provided by a given object.

SmartBinding. SmartBinding ensures that the optimum transport mechanism is chosen whenever a client binds to a server object. If the object is local to the client process, the client performs a virtual function call. If the object resides in a different process on the same host, the client uses an optimized inter-process communication mechanism. If the object resides on a different host, the client uses IIOP over the network.

Dynamic Object Creation. Dynamic object creation further enhances scalability and performance. Objects are registered with VisiBroker's Object Activation Daemon, which can run on each machine that hosts server objects. If a client attempts to bind to an object that is not currently running, the activation daemon can start an instance of the object. Objects can also be started manually. Use of the Object Activation Daemon is optional.

Scalability of IIOP. The combination of HTTP and CGI that drives many of today's first-generation Web applications does not scale well. CGI requires a separate server-side process to handle each client request, resulting in inefficient use of system resources and poor scalability. IIOP is both more robust and more scalable and therefore much better suited to developing business applications.

Suitability for the Internet

VisiBroker is the ideal environment for developing distributed, object-based applications for use over the Internet or enterprise intranets.

The Leading Java ORB. Netscape selected VisiBroker to enable its Communicator suite of client software (including version 4.0 of the Netscape Navigator) to support IIOP. Netscape will also bundle VisiBroker development software with its SuiteSpot server products. As a result, developers of Web-based applications can make use of a communication infrastructure that is already integrated into their Web servers and users' browsers.

Access Objects behind Firewalls. The VisiBroker Gatekeeper enables an application object outside the enterprise firewall to access objects running on machines behind the firewall. For instance, users on the Web can access the functionality of objects that are part of an internal enterprise system.

HTTP Tunneling. Most firewalls do not allow IIOP traffic, but are configured to allow inbound and outbound HTTP traffic. The Gatekeeper enables IIOP messages to pass through firewalls by encapsulating IIOP packets in HTTP.

OMG IDL Not Necessary. A Java-to-IIOP translator utility relieves Java developers from having to write OMG Interface Definition Language (IDL) for Java objects. The utility is able to infer the IDL object interfaces and automatically create the object stubs and skeletons. It enables Java developers to build applications that use CORBA objects without having to learn IDL or adopt C ++ programming conventions.

Web Naming Convention. Objects can be named through the URL naming convention. VisiBroker intercepts the request, looks in the file (specified by the URL), extracts the proper CORBA object reference from that file, and then binds to that object.

"Visigenic's leading technology combined with Oracle's Web Application Server will make it easier to build application cartridges and will extend the strength of Oracle's database, Web server and object development tools to new audiences."

— Jerry Held

Senior Vice President

Server Technologies

Oracle Corporation

Summary

In the age of global competition and increasingly narrow market opportunities, companies that can initiate and adapt to change have a significant competitive advantage. Those organizations that embrace the converging distributed object and Internet technologies will thrive and dominate their markets because they will offer superior customer service, respond rapidly to changing market opportunities and adapt quickly to new market conditions.

The OMG has specified an architecture, CORBA, for an open software bus on which objects can interoperate across networks and operating systems, communicating via IIOP. With its VisiBroker products, Visigenic is playing a prominent role in the shift to CORBA-based distributed applications. The ORB implementation of choice for the industry's leading technology vendors—including Netscape, Oracle and Hitachi—VisiBroker enables developers to build, deploy and manage robust, high-performance distributed business applications for enterprise use.

For more information on Visigenic and the VisiBroker product family, please visit our World Wide Web site at www.visigenic.com.

About Visigenic

Visigenic is playing a key role in creating the foundation on which mission-critical applications of tomorrow will be written—the open, distributed, object-based architecture for the new global enterprise. Building on its leadership and expertise in standards-based distributed object and data access technologies, Visigenic is a pioneer in managing distributed business logic and its access to data.

Visigenic distributed object products include VisiBroker for Java and VisiBroker for C++, which are both based on the Common Object Request Broker Architecture (CORBA), and utilize the Internet Inter-ORB Protocol (IIOP). Visigenic data access products include VisiChannel for ODBC, VisiODBC Drivers, the VisiODBC Software Development Kits, and the upcoming VisiChannel for JDBC.