CORBA: Catching the Next Wave

June 5, 1997

This white paper presents CORBA - its history, features and benefits, and practical applications. It also describes Netscape's plans to integrate CORBA across its product line, enabling the next generation of applications for intranets, extranets, and the Internet.

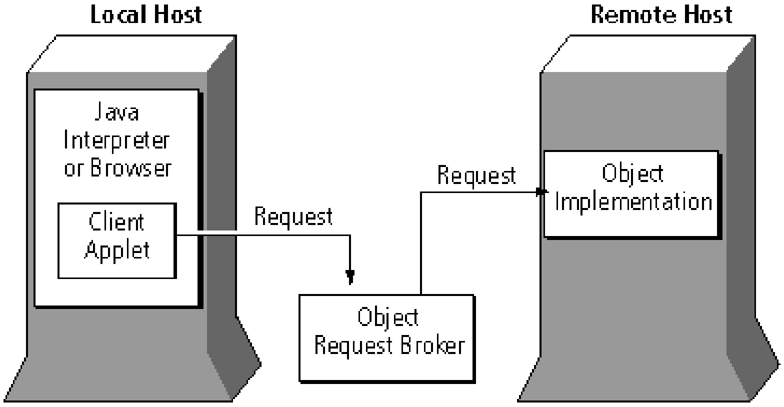
Contents

back to top

What is CORBA?

Distributed objects are the next wave in Internet innovation. CORBA, the Common Object Request Broker Architecture defined by the Object Management Group (OMG), specifies how software objects distributed over a network can work together without regard to client and server operating systems and programming languages.

CORBA is a complete distributed object platform. It extends applications across networks, languages, component boundaries, and operating systems. A CORBA Object Request Broker (ORB) connects a client application with the objects it wishes to use. The client application does not need to know whether the object resides on the same computer or on a remote computer elsewhere on the network. The client application needs to know only two pieces of information: the object's name and how to use the object's interface. The ORB takes care of the details of locating the object, routing the request, and returning the result.



HISTORY

As CORBA grew, three influential trends were taking shape:

1. The software development community realized the importance of *object-oriented programming* techniques, even though they'd been around for the better part of two decades.
2. Industry leaders-including IBM, Microsoft, and Apple-were advocating new application models based on small, *task-specific components*. Because they are smaller than general-purpose applications, components are easier to write and update. Developers can, conveniently and cheaply, upgrade only those parts of their software that are out of date, and users don't have to purchase new versions of the entire package. Also, small components can be distributed over a network more easily than their monolithic ancestors. With component architectures comes the promise of a cross-platform future where components running on different systems work together seamlessly.
3. Just after the release of CORBA 2.0 in early 1995, the *Java programming language* burst forth, as if to validate the concepts underlying the vision of an object-oriented, component-based future. Java provided an object-oriented development environment for producing small software components that could run on any operating system with a Java virtual machine. And the components could be used over the Internet, which by then was surging in popularity.

CORBA fits the component-based and Internet-based approaches to building and using software. It defines a way to divide application logic among objects distributed over a network, some on clients, others on a variety of servers. It also defines a way for those objects to communicate and use each others' services. CORBA supplements Java with a rich set of services that includes introspection, dynamic discovery, transactions, security, naming, and more. CORBA links the Java mobile code environment and the world of interoperable objects.

FEATURES AND BENEFITS

CORBA defines an object-oriented approach to creating software components that you can reuse and share between applications. Each object encapsulates the details of its inner workings and presents a well-defined interface, reducing application complexity. The CORBA approach also reduces development costs, because once an object is implemented and tested, it can be used over and over again. CORBA's platform independence lets you run and invoke the object from any platform; you can run an object from the platform that makes the most sense for that object.

CORBA's language independence lets you reuse existing code and leverage your existing programming skills.

Netscape envisions 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 information technology (IT) organizations are significant:

Choice. CORBA is based on an open, published specification. It is implemented on and supported by numerous hardware and operating system platforms. CORBA Java objects are portable. You can build objects on one platform and deploy them on any other supported platform.

Interoperability. CORBA objects are fully interoperable because they communicate using the Internet Inter-ORB Protocol (IIOP). An enterprise IT organization need consider only its functional needs when selecting an ORB, CORBA services, and software objects, even when they are developed by different vendors. In addition, software bridges enable communications between CORBA objects and objects developed using Microsoft's DCOM technology.

Modularity. CORBA objects interact via interfaces. Because interface and implementation are separate, developers can modify objects without breaking other parts of the application. Changing an object's implementation does not affect other objects or applications because that object's interface stays the same.

Compatibility. CORBA protects your investment in existing systems. You can encapsulate a legacy application, module, or entry point in a CORBA IDL wrapper that defines an interface to the legacy code. This object wrapper makes the legacy code interoperable with other objects in a distributed computing environment.

Security. CORBA provides security features such as encryption, authentication, and authorization to protect data and to control user access to objects and their services.

back to top

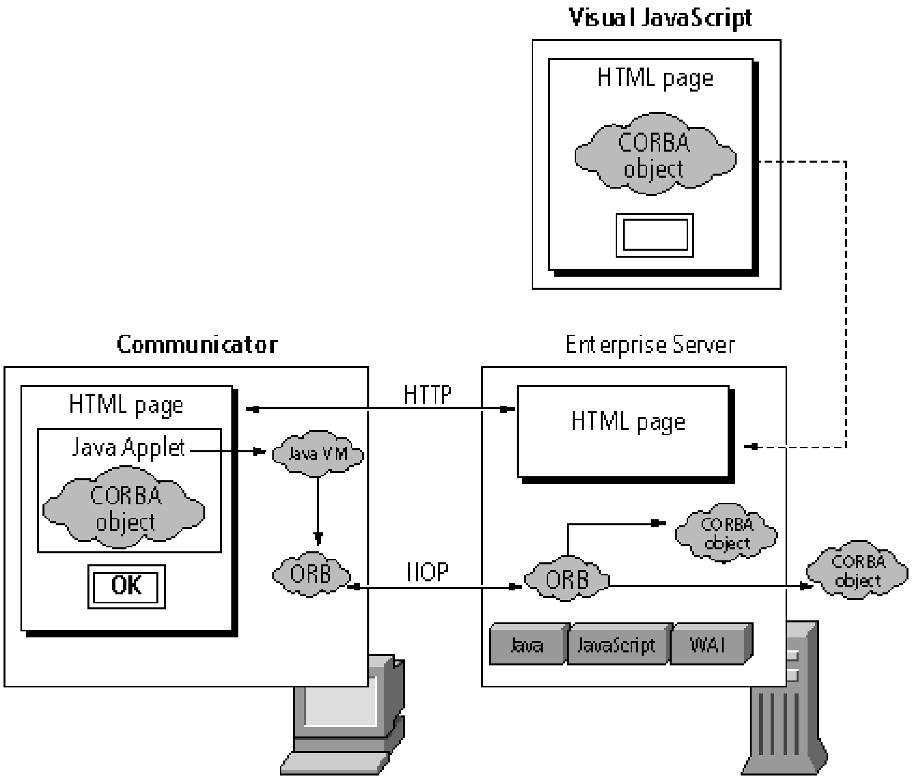
CORBA EVERYWHERE

Netscape and its partners want to make CORBA and the Internet Inter-ORB Protocol (IIOP) as ubiquitous as HTML and HTTP, making Internet-based services as widely available as Internet-based content is today. The 700-plus members of the OMG-including Sun, IBM, Oracle, Hewlett-Packard, Digital Equipment Corporation, Apple Computer, Novell, and BEA-have already endorsed CORBA, and many companies are currently implementing CORBA-based information technology solutions. CORBA has already been implemented by many organizations, and thousands of CORBA applications are currently in use, mostly behind corporate firewalls.

Netscape is making CORBA a cornerstone technology, integrating it with every client and server solution they sell from now on. Netscape Technology Senior Vice President Marc Andreessen briefly described the company's CORBA vision in a Techvision column about the CORBA network protocol, IIOP. As Marc emphasizes, CORBA enables access to a range of services-using databases, tracking inventory, keeping medical records, controlling corporate finances, searching for information-in addition to the kinds of content and applications available today. These services are implemented as objects stored on servers of all types; CORBA lets objects written in virtually any language interact. These objects are small software components rather than large applications. For this reason, objects can be updated and distributed far more quickly and easily than older, traditional software.

The first step of Netscape's CORBA strategy is to make CORBA an integral part of every client and server across its installed base. Netscape includes the following with its new Communicator and SuiteSpot product lines:

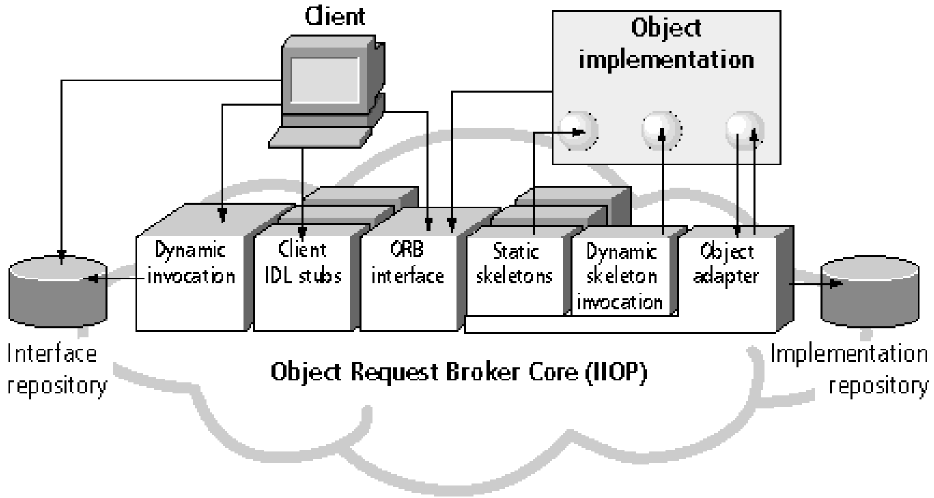
- **Netscape Communicator** will ship with the Netscape Internet Service Broker (ISB) for Java-a Java ORB-making it possible for clients to use the services of CORBA-compliant objects starting with Communicator Preview Release 3 (PR 3).
- **Netscape Enterprise Server 3.0**, the core server component of the SuiteSpot 3.0 server suite, includes both Java and C++ ORBs, Netscape ISB for Java and C++, starting with the beta 3 release. Included with the server is a single-user developer license and all the tools that come with it, including IDL compilers, network agents, and other utilities. Enterprise Server 3.0 also includes the Web Application Interface (WAI), which augments the Netscape server application programming interface (NSAPI). WAI has an IDL specification and lets server plug-ins and other server-side software objects communicate with CORBA objects on the network.
- **Netscape's Visual JavaScript** development tool can integrate CORBA-compliant objects into an HTML page. Then users can work with these objects interactively at design time and programmatically (via JavaScript) at run time, just as they do with any other JavaBean component. With CORBA, users can interact not only with subjects in HTML pages but also for objects that can perform services.



back to top

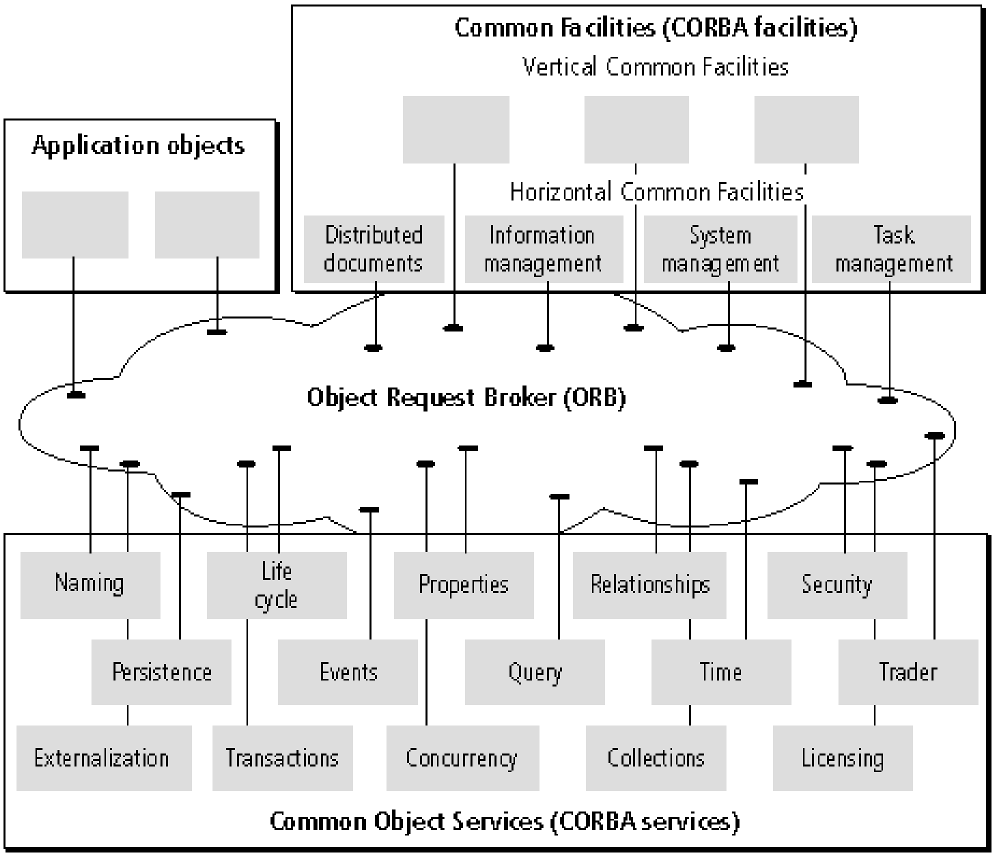
HOW CORBA WORKS

A CORBA Object Request Broker (ORB) is the middleware that establishes the client-server relationship between objects. Using an ORB, a client object can invoke a method on a server object that can be on the same machine or across a network. The ORB intercepts the call and finds an object that can implement the request, pass it the parameters, invoke its method, and return the results. CORBA, like SQL, provides both static and dynamic interfaces to its services. The client does not have to know the object's location, its programming language, its operating system, or any other system aspects that are not part of an object's interface. Also, the client and server roles are dynamic: an object on the ORB can act as either client or server, depending on the occasion.



Together, ORBs and the CORBA architecture provide the mechanism for CORBA objects to communicate. The objects are small software components that provide some kind of a service, such as access to a database, account management, or inventory tracking. Fundamental to the architecture are the ORBs. For any client or server to be a part of the CORBA scheme, it must include an ORB to help it find and communicate with other CORBA objects. Once outfitted with an ORB, a client or server can use the services of any CORBA object on any server or host on the network. An object can be both a client and a server: when it provides services to another object, it's a server; when it requests services from another object, it's a client.

The CORBA architecture provides a set of services that help objects interact with each other. (In the CORBA world, *services* means both the services provided by the CORBA architecture to help objects communicate and the functionality provided by the objects themselves.)



The services are among the efficiencies provided by CORBA: You write your object's code, and CORBA takes care of how your object identifies itself, finds other objects, learns of network events, handles object-to-object transactions, and maintains security. The services include the following:

- A *naming service* allows a CORBA client to locate a named CORBA object. It maps a human-readable name to a CORBA object reference.
- An *event service* enables objects to dynamically register and unregister interest in specific events. Publishers and subscribers each connect to an event channel; publishers send messages (events) to the event channel, and subscribers receive these messages (events) asynchronously.
- A *transaction service* allows transaction contexts between objects to be transparently propagated via IIOP. It coordinates two-phase commits of transactions (flat or nested) among objects.
- A *security service* provides features including encryption, authentication, and authorization to protect data and to control user access to objects and their services. It also manages the delegation of credentials between objects.

CORBA also defines the Internet Inter-ORB Protocol (IIOP) to govern how objects communicate over the network. Unlike other CORBA solutions based on proprietary protocols, Netscape's implementation is completely open. It makes strict use of IIOP, an open protocol that runs on top of TCP/IP. Unlike HTTP, IIOP allows state data to be preserved across multiple invocations of objects and across multiple connections.

Current versions of the Netscape Internet Service Broker support the CORBA naming service, and

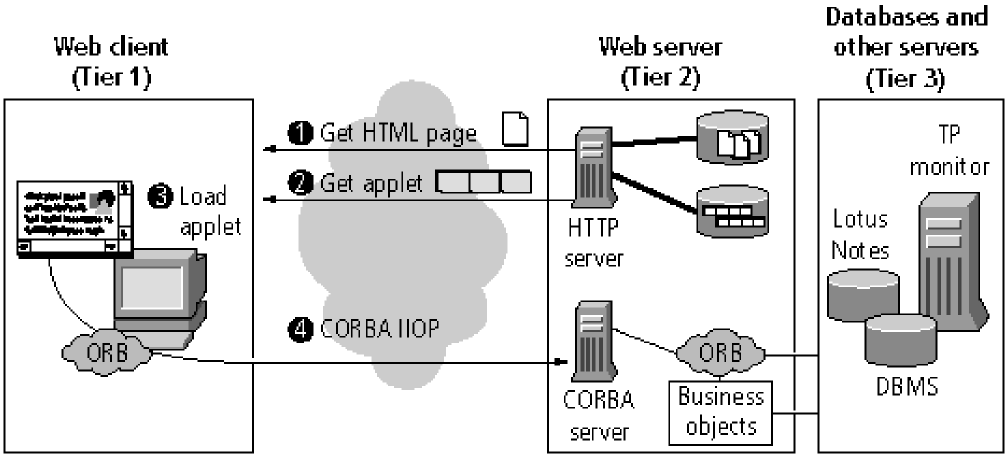
Netscape engineers are working on updates that will include support for the event, transaction, and security services for future releases of Communicator and Enterprise Server. Further, Netscape and several of its partners are extending the naming service for easier interoperability between different vendors' ORBs.

back to top

CORBA IN ACTION

With an ORB-enabled browser, a user can access services of objects from a multitude of servers and hosts. A CORBA object, using IIOP and the client ORB, can access the services of an object on a Web server (through the server ORB), which in turn can access one or more objects on relational databases or legacy systems (as long as each system includes an ORB).

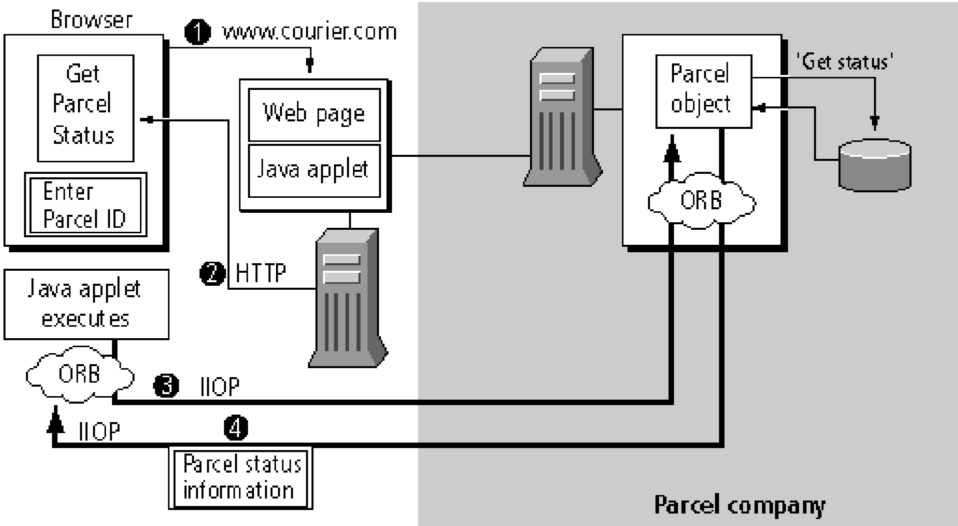
A client object requests the services of other objects through an ORB that resides on the client system. Using IIOP, the client ORB then reaches across the network, looking for ORBs on other systems and the server objects that can provide the requested services. Each object has a unique name, provided according to the CORBA naming service, which identifies it and its services to other objects. Once the ORB has found the requested service/object, the client object communicates with the server object, still using IIOP. Each object's IDL interface tells other objects how to use its services and the results those services generate.



As a practical example, consider a parcel-tracking application. (This example is based on a similar example used by Visigenic Software in its white paper about CORBA, Distributed Object Computing in the Internet Age, which is available from Visigenic as a PDF file.) Suppose an Internet user wants to track delivery of a package. From a browser, the user enters a URL that points to the delivery company's Web server. The server uses HTTP to send a Web page to the user; that page contains an embedded Java applet, which is the client component of the parcel-tracking system. Note that so far, nothing is different from a traditional HTTP Web transaction.

Next, the user enters a parcel-tracking number into the appropriate field in the Java applet. Using the client-side ORB, the applet then generates and sends an IIOP message across the network, looking for a specifically-named server object that can obtain the status of the parcel. The ORB on the server with the appropriate parcel-tracking object picks up the message and invokes the object's status-finding method. Through the server ORB, the object then generates another IIOP message looking for an object on a legacy system that contains all the parcel company's data.

Upon obtaining the last request through its ORB, the mainframe containing the data-tracking object determines that status of the parcel from its database. The mainframe returns an IIOP message with the information to the server object. The server then routes the information-again with IIOP and the ORBs-to the Java applet running on the client, which displays the results to the user.



CORBA objects do not have or need information about each other's implementation details. They communicate only through their published interfaces, and the ORB manages these communications transparently to users. The client could be a PC running Windows, a Macintosh, a workstation, a network computer, a handheld personal computer, or even a set-top box. Server objects can be programmed in a variety of languages and may include new and legacy code. Because they communicate with the objects via IIOP, clients can access functionality directly, without going through Web server software and without the performance costs of processing a CGI script for each user access.

CREATING CORBA OBJECTS

You can create CORBA objects in one of two general ways:

- Write and compile an interface definition language (IDL) specification.
- Use `java2iiop` (also called Caffeine), provided with Netscape's Enterprise Server. This utility processes Java bytecode and generates files for CORBA objects.

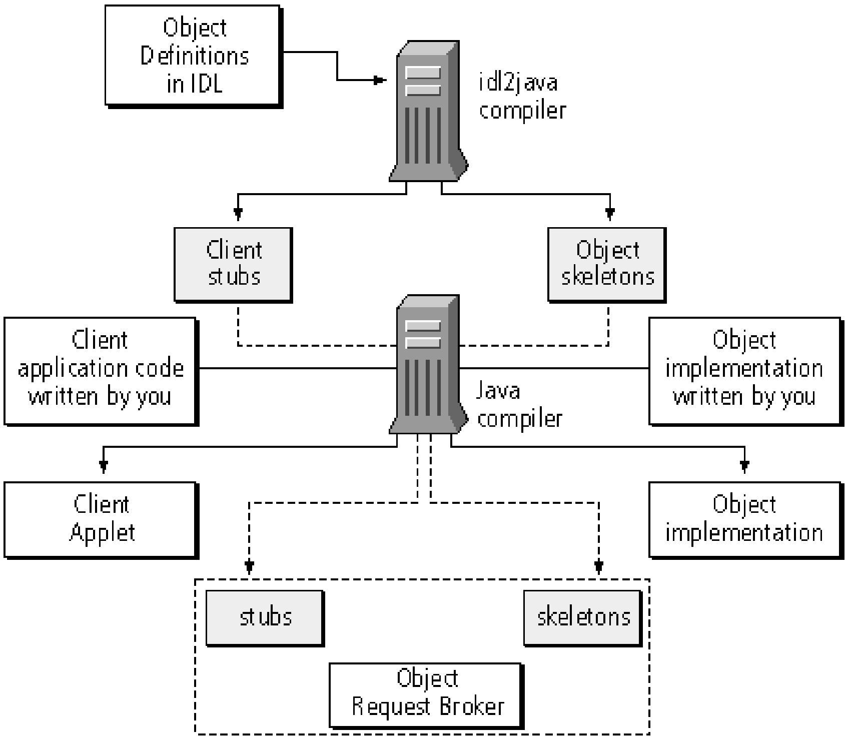
USING IDL

CORBA defines an interface definition language (IDL) that provides a language-neutral way to describe a CORBA object and the services it provides. IDL lets components written in different languages communicate with each other using IIOP and the rest of the CORBA architecture. CORBA objects can reside on different types of systems, including Windows or UNIX servers and

IBM 3090 or DEC VAX mainframes. They can be written in different languages (Netscape provides tools for developing in Java and C++). As long as interfaces to their services are written in IDL, the objects can communicate and use each others' services through ORBs on clients, servers, database systems, mainframes, and other systems on the network.

Here are the basic steps:

1. Write a specification for the objects using IDL.
2. Using the appropriate compiler, compile the IDL code into Java or C++ code stubs and skeletons. The stubs are used by the client, and the server skeletons provide the framework that you fill in with the code for the service your object is to provide.
3. Write the Java or C++ code to implement the service.
4. Compile the code created in the previous step.

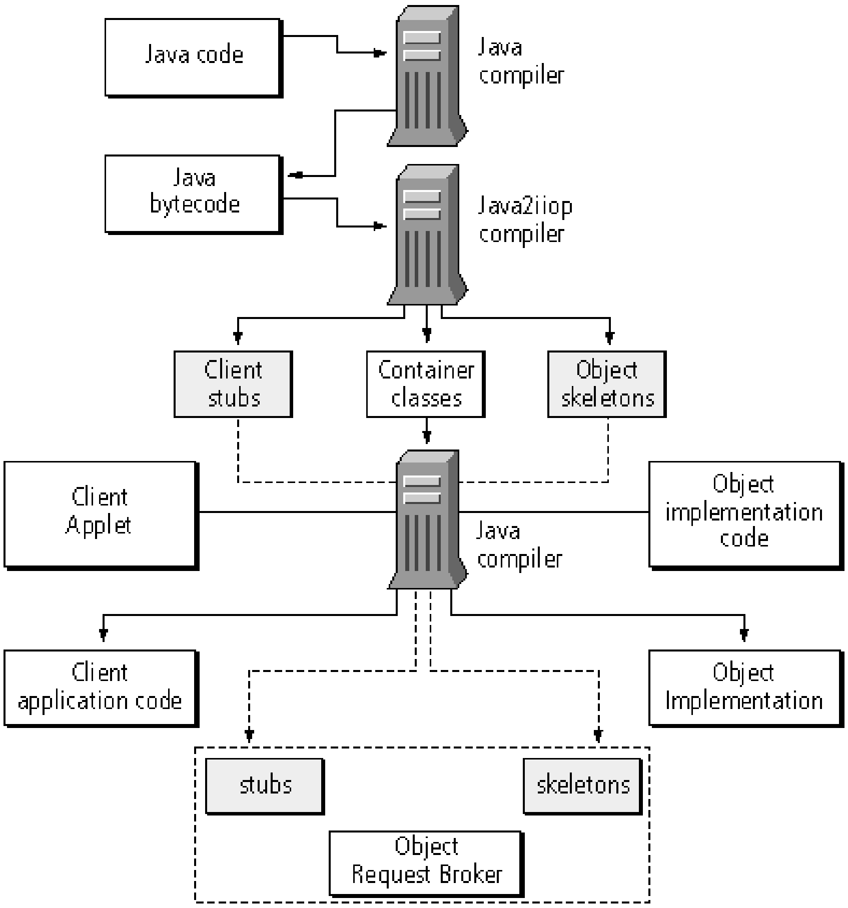


Whether you use Java or C++ depends on your preferences and the solution you're trying to develop. Java is probably more flexible, because a Java object will work on any operating system with a Java virtual machine. However, C++ may give better performance, depending on your server environment.

The specific techniques you'd use to create new Java and C++ CORBA objects are described in the Netscape Internet Service Broker documentation that's part of the current Enterprise Server 3.0 release. Start with the *Netscape Internet Service Broker for Java Programmer's Guide* and the *Netscape Internet Service Broker for C++ Programmer's Guide*.

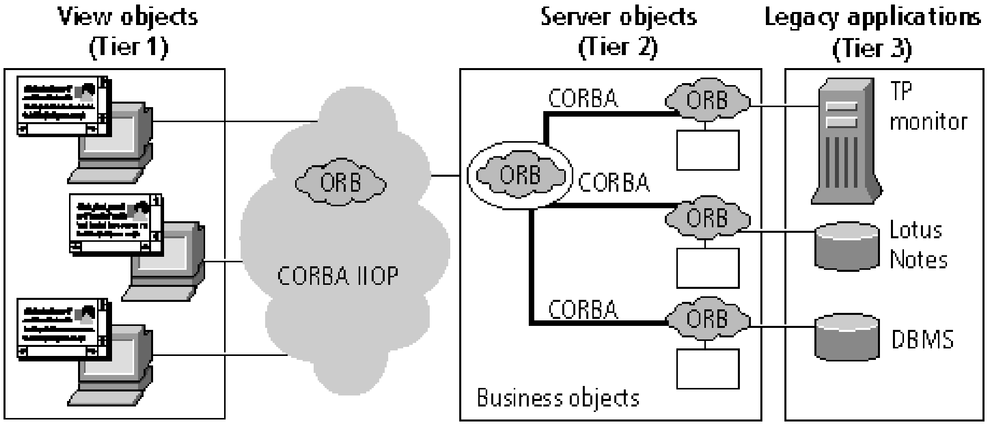
CAFFEINE: CORBA WITHOUT IDL

Netscape's Enterprise Server 3.0 includes a java2iiop compiler (also called Caffeine) for defining CORBA-compliant interfaces using Java instead of IDL. By using java2iiop, you can generate the necessary container classes, client stubs, and server skeletons from Java bytecode. Caffeine makes it easy for Java programmers to write objects that interoperate across Java VMs using Java language semantics. You don't have to learn CORBA IDL to make Java objects remotable and accessible via a CORBA IIOP ORB. Caffeine also lets you pass objects by value. Finally, it provides a naming service that maps CORBA object references to URLs. The following figure shows the development process.



REUSING EXISTING CODE AND LEGACY APPLICATIONS

CORBA provides a path for advancing legacy systems and other existing code bases into the cross-platform, Internet-based present and future. You can encapsulate a legacy application, module, or entry point in a CORBA IDL 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. Unlike many competing schemes, CORBA bridges the past of proprietary code with the present of open, Internet-based applications; it also bridges the present with the future of distributed Internet-based object computing.



Divide your legacy code and other older code into smaller objects and write an IDL specification for them. This will expose the object's services to other CORBA objects across the Internet, making it possible for other CORBA objects to communicate with your objects. With CORBA, you can preserve your current code base and give it an interface (through IDL) that makes it accessible via Internet technology.

back to top

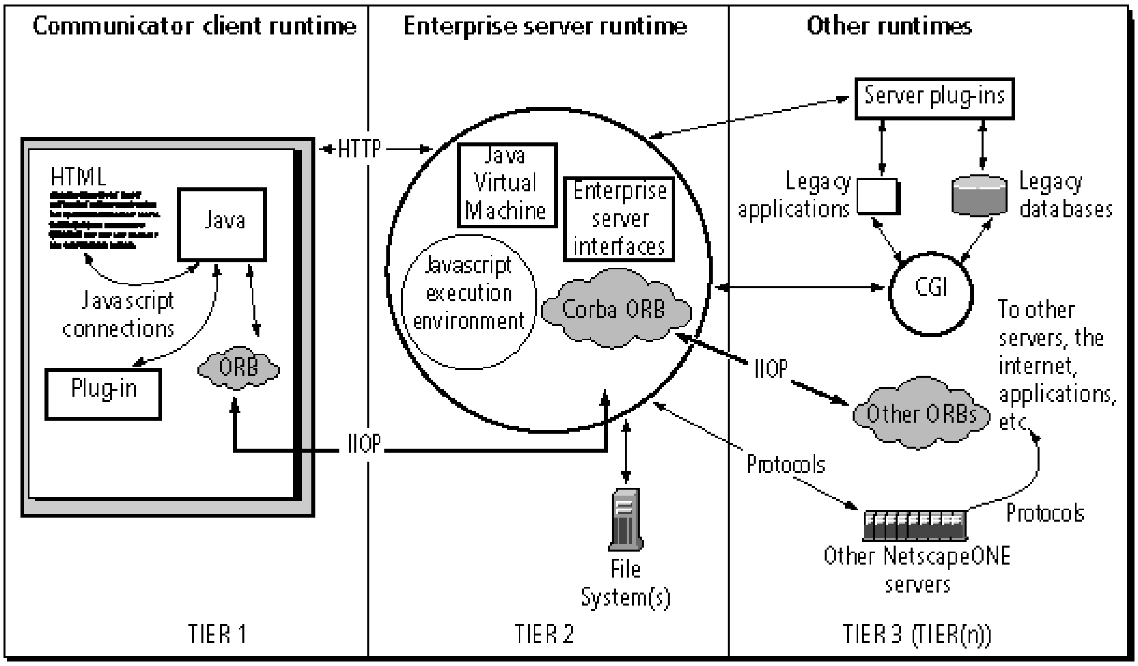
THE NETSCAPE ADVANTAGE

The foundation of Netscape's business and technology strategy is to provide full capabilities across all platforms, including multiple versions of the same platform. Netscape plans to integrate CORBA into all its products, enabling your Netscape ONE applications-and your users-to take advantage of CORBA through

- Crossware
- Netscape Internet Service Broker
- Scriptable CORBA

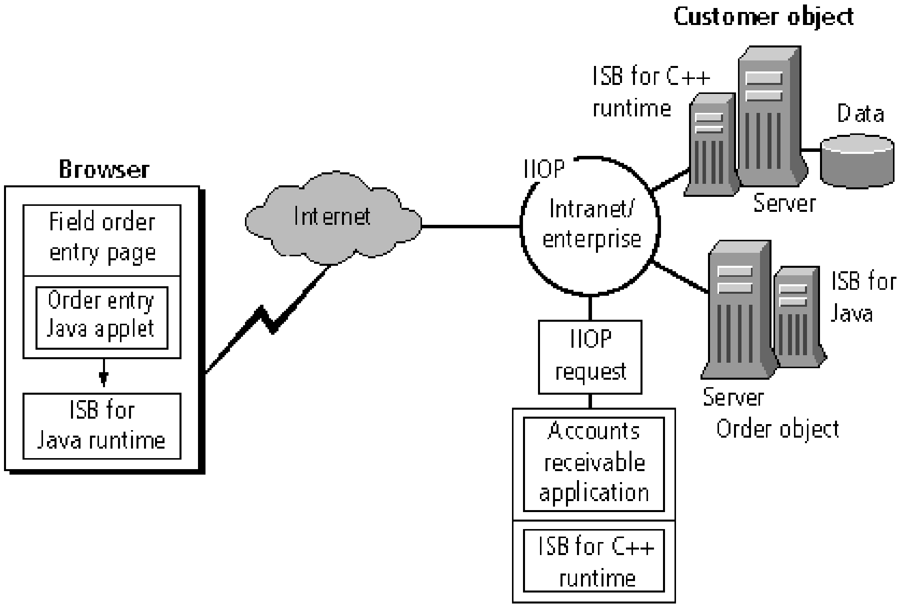
CROSSWARE

Marc Andreessen describes *crossware* in his vision paper *The Networked Enterprise*. Crossware describes on-demand applications that run across networks and operating systems, and are based entirely on open Internet standards like HTML, Java, and JavaScript. Think of crossware as next-generation client-server applications, residing partly on the client and other partly on any number of servers or hosts connected to the network, inside and outside of corporate firewalls. You write each part without regard for the server platforms where it may reside or the clients that may want to use it. The components communicate using currently-available Internet protocols that are open and platform-independent. Through CORBA and IIOP, this vision becomes reality.



NETSCAPE INTERNET SERVICE BROKER

As announced at the October 1996 Netscape Developers' Conference in New York City, Netscape has licensed technology from Visigenic Software, Inc., to speed the integration of CORBA across the Netscape product line. This technology is available to developers as the Netscape Internet Service Broker (ISB). Netscape Communicator provides runtime support for ISB for Java, and Enterprise Server 3.0 supports development and deployment of ISB for Java and for C++. Through ISB, Communicator and SuiteSpot can use the thousands of CORBA applications that already exist, and the many thousands of CORBA-compliant software objects that Netscape expects to be written in the near future.



In addition to providing the basic ORB features defined in the CORBA specification, Netscape's ISB for Java offers enhancements that increase application performance and reliability.

Fault tolerance	ISB can determine if the connection between your client application and an object server has been lost because of a server crash or network failure. When it detects a failure, it tries to restart the server or to connect your client to a suitable server on a different host.
Optimized binding	When your application binds to an object, ISB selects and establishes the most efficient communication mechanism. Depending on the platform and the location of the requested object, the bind may be established through a pointer reference or a TCP/IP connection.
Dynamic Invocation Interface	Client applications can obtain an object's interface and dynamically construct requests to act on the object.
Interface repository	The interface repository (IR) maintains information about ORB objects, such as modules, interfaces, operations, attributes and exceptions. An IDL interface enables your client applications to query the IR to obtain language binding information or to discover new interfaces. The ORB also accesses the IR when it needs to check the type of values in a client request or to verify the correctness of an interface inheritance graph.
Web Naming service	Web Naming allows you to associate URLs with objects, allowing an object reference to be obtained by specifying a URL.
Caffeine: Defining interfaces without IDL	The `java2iiop` utility (also called Caffeine) allows you to use Java instead of IDL to define interfaces. Use `java2iiop` when you want to adapt existing Java code to use distributed objects or if you do not have the time to learn IDL. It generates the necessary container classes, client stubs, and server skeletons from Java code.
Enhanced thread and connection management	ISB provides two thread policies: thread-per-session and thread pooling. You choose a policy, then ISB automatically selects the most efficient way to manage connections between client applications and servers.

Dynamic Skeleton Interface	The Dynamic Skeleton Interface (DSI) provides a mechanism for creating an object implementation that does not inherit from a generated skeleton interface. Normally, an object implementation is derived from a skeleton class generated by the `idl2java` compiler. DSI allows an object to register itself with the ORB, receive operation requests from a client, process the requests, and return the results to the client without inheriting from a skeleton class.
Enhanced `idl2java` compiler	The `idl2java` compiler includes its own Java-based preprocessor, which supports standard preprocessing arguments.

SCRIPTABLE CORBA

Netscape makes CORBA objects scriptable through its new development tool, Visual JavaScript. You can query the CORBA interface repository for an object to use in an HTML page. This tool wraps the object in JavaScript code and displays it in the tool palette where you can inspect it, script its behavior, and use it like any other JavaBean component.

back to top

COMPETITION

This section presents competing technologies, that is, technologies developed to do what CORBA does.

HTTP and CGI	The CGI/HTTP protocol is clumsy, stateless, and extremely slow-much slower than CORBA IIOP. Programming Internet client-server applications with CGI is a very poor choice. The bad news is that CGI is the premier three-tier client-server application model for the Internet today. The good news is that the leading Internet architects, well aware of CGI's shortcomings, are migrating to alternative technologies. Netscape will do its part by bundling a CORBA ORB with every browser.
RMI	First, RMI does not provide language-neutral messaging services. In other words, RMI objects can talk only to other RMI objects. With RMI, you cannot invoke objects written in other languages or

	vice versa. Second, RMI does not support dynamic invocations and interface repositories. Third, RMI does not provide a wire protocol for security and transactions. RMI is both proprietary and lightweight. It was not designed to interoperate with other ORBs or languages. Unlike CORBA's IIOP, RMI is not a suitable backbone for the Internet or intranets; it lacks services IIOP provides.
DCOM	A DCOM object is not an object in the object-oriented programming sense; a DCOM object does not have a persistent object reference that lets you reconnect to the same object at a later time. In other words, DCOM objects do not maintain state between connections. This can be a big problem in environments where you have faulty connections - for example, the Internet. The current implementation of DCOM does not support distributed naming services; it is based on the NT registry. Configuring DCOM and installing type libraries is tedious and labor-intensive. DCOM is also Windows-centric. Very few implementations of DCOM run on non-Windows platforms. Finally, for DCOM to scale on the server side, it requires the Microsoft Transaction Server (MTS). Microsoft has no immediate plans to port MTS to non-NT platforms.
RPC	With an RPC, you call a specific function (the data is separate). In contrast, with an ORB, you call a method within a specific object. Different object classes may respond to the same method call differently. Because each object manages its own private instance data, the method is implemented on that specific instance data. ORB method invocations are precise. The call gets to a specific object that controls specific data and then implements the function in its own class-specific way. In contrast, RPC classes have no specificity: all the functions with the same name are implemented the same way.

back to top

CONCLUSION

CORBA provides an object bus that is the foundation for the next generation of Internet applications. Any CORBA object can be both a client and a server. CORBA lets you interact with server objects from browsers in an object-oriented way. It represent a major advance over stateless protocols such as HTTP. CORBA lets you leverage existing code and turn it into objects. Most importantly, CORBA provides a solid server-to-server backbone for creating the Object Web. For example, server-side objects from Netscape, Oracle's NCA, IBM's Component Broker, BEA's Iceberg, and Sybase's Jaguar can seamlessly interoperate over a common IIOP ORB. CORBA provides an inter-vendor infrastructure that mirrors the way the Web will be used to create the next generation of business-to-business and consumer-to-business applications. In addition, vendors such as Netscape, Borland, Symantec, IBM, Sybase, and Forte are adding CORBA support to their visual tools. This will make CORBA accessible to the programming masses. With CORBA, you are limited only by your imagination.

back to top

FURTHER READING

The authors would like to acknowledge the contribution of Paul Dreyfus, whose article *CORBA Theory and Practice* (listed below, and available in View Source magazine) was an important source of information.

We would also like to thank Robert Orfali for his contributions and editorial assistance.

- IIOP and the Distributed Objects Model
- The Networked Enterprise: Netscape Enterprise Vision and Product Road Map
- Netscape Internet Service Broker for Java Programmer's Guide
- Netscape Internet Service Broker for C++ Programmer's Guide
- CORBA Theory and Practice
- CORBA 2.0/IIOP Specification
- Distributed Object Computing in the Internet Age
  (Visigenic CORBA white paper. Available only in PDF format)
- Client/Server Programming with Java and CORBA
  Robert Orfali and Dan Harkey. 1997, John Wiley and Sons, Inc.
- Instant CORBA
  Robert Orfali, Dan Harkey, and Jeri Edwards. 1997, John Wiley and Sons, Inc.
