Articles & White Papers

Object-Oriented and Component-Based Development

CIOs are under pressure to implement objects;
here's a guide through the maze of vendor claims and object categories

The Object Infrastructure Challenge, PART 2

b y m a x d o l g i c e r

Corporate information systems (I/S) organizations are under pressure to implement objects. Indeed, most CIOs in large corporations have already received orders to implement "distributed objects everywhere." Difficulties arise as the CIO faces implementing such a policy.

Microsoft and the Object Management Group, Framingham, Mass., both have defined object infrastructures (see **reference**). The question remains, however, how do operating systems, hardware platforms and OLE/COM integration fit into an object infrastructure?

Platform requirements

A key requirement of middleware in general is to provide developers with the same functionality across multiple operating systems and communication protocols. In theory, the operating system should not matter. In reality, however, support of an object middleware for a specific range of platforms, as well as tight integration with the operating system, can influence corporate purchasing decisions.

In some cases, corporations choose between a consistent cross-platform implementation that may be the lowest common denominator and a system that offers maximum exploitation of the services of a specific operating system. Tight integration with a specific operating system sometimes comes at the expense of cross-platform consistency.

We have yet to see a product offering the best of both worlds. So it is inevitable that the choice of tactical and strategic operating systems will have some influence on the selection of a distributed object infrastructure. It is unrealistic to ignore the issue of operating system support.

Windows 3.1, 95 and NT

If a corporation's applications are mostly Windows-based, then it is hard to justify any object-oriented (OO) middleware beside Microsoft's own Distributed Common Object Model (DCOM). Windows NT has found rapid acceptance in corporate I/S shops. Increasing numbers of sites are deploying applications on NT servers. Many are even bypassing Windows 95 to standardize on NT as the corporate client operating system.

The huge corporate installed bases of Windows 3.1 and 95 can utilize Windows NT-based server applications and integrate them with several Microsoft desktop productivity tools like the Excel spreadsheet and Microsoft Word word processor. In addition, more than 600 commercial third-party application packages are OLE/COM enabled, making that technology a very strong contender for new application development on these platforms.

But what about distributed applications? With Windows NT 4.0 and DCOM, one of the most attractive features incorporated in NT 4.0, shipped ahead of schedule in August, Microsoft's ability to offer a distributed object infrastructure has become a reality.

Although several products based on Corba run on the Microsoft platforms, the selection of DCOM as the object middleware of choice at a Windows-only site is almost inevitable. In addition to the fact that DCOM is well integrated with the operating system, and that its programming model has great appeal to NT programmers, it comes bundled in the operating system without charge.

However, DCOM does lack support for the same middleware services as most Corba-based products. Those services include transactions, distributed naming, message queuing, load balancing and failover. Users should remember that DCOM just started shipping and can hardly be qualified as a mature product.

Windows and Unix

Over the past few years, Windows-based clients and Unix-based server components have been a very popular platform combination for building distributed, OO applications. Such applications have been implemented using Corba products when object middleware and a non-object middleware system like Message-Oriented Middleware (MOM), distributed TP monitors, Distributed Computing Environment (DCE) from Open Group, Cambridge, Mass., or any of the database gateways are selected.

All of the Corba-compliant products support Windows and a number of Unix platforms. Observers do not expect a Unix implementation of DCOM until at least early 1997.

For companies that have a substantial investment in Unix and plan to continue using Unix as a strategic platform, Corba products are the choice by default -- at least for now. Corba also remains the choice for those companies requiring a consistent middleware environment across multiple operating systems. As a result, some companies have opted for products such as Orbix from Iona Technologies, Marlborough, Mass., or ObjectBroker from Digital Equipment Corp. to connect distributed objects running on Windows NT and Unix platforms. These products can provide the same functionality across a multitude of operating systems, including Windows NT and Unix.

OLE/Corba Integration

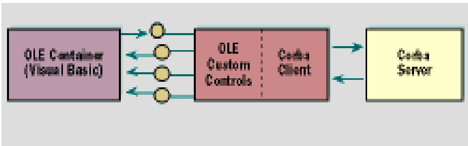

Fig. 1 Source: International Systems Group Inc.

OLE/COM and Corba integration allows the implementation of client applications using OLE interfaces, whereby the OLE-based requests are internally translated and forwarded to Corba objects residing on Unix. This approach allows popular software packages like PowerBuilder, Visual Basic and Excel to access objects developed using Corba-compliant products.

Last year one feature that established itself on the "must have" list of most Corba vendors was support for OLE Automation. This technology allows the implementation of client applications using OLE interfaces, whereby the OLE-based requests are internally translated and forwarded to Corba objects residing on Unix. This approach allows popular packages like PowerBuilder from the Concord, Mass.-based Powersoft unit of Sybase Inc., Emeryville, Calif., and Microsoft’s Visual Basic and Excel software to access objects developed using Corba-compliant products. (See **Fig. 1**.)

This concept will be addressed in more detail later, in the section on "OLE integration with Corba."

Mainframe (MVS)

IBM is certainly making a strong case for the development or deployment of new OO applications on MVS-based mainframes. IBM provides languages like C++ and Smalltalk, and its Corba product, System Object Model (SOM), for MVS, an appealing offering for some large corporate I/S organizations. Generally, these companies are very sophisticated, leading edge technology customers that tend large I/S budgets and are convinced that MVS is still a strategic operating system; that consistent middleware functionality across all strategic operating systems is a must; and that object technology can provide them with a competitive advantage.

For these companies, implementing a consistent object middleware infrastructure across client and server operating systems like Windows, Unix and MVS is a potential solution. This approach offers consistency and can minimize operational issues that can have a direct effect on the bottom line. However, we believe that these companies do not represent a significant piece of the corporate world.

Combining OO and Non-OO Middleware

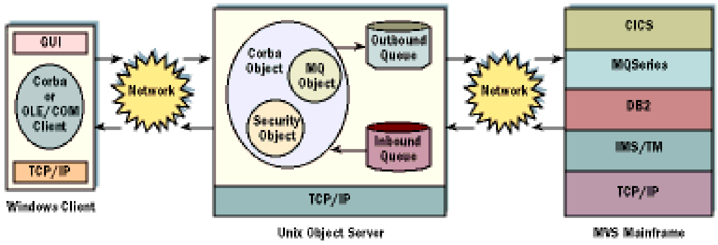

Fig. 2 Source: International Systems Group Inc.

MVS-based applications that are not object-oriented can be linked via non-object-oriented middleware like DCE or MOM to objects that were developed using either Corba- or OLE/COM-based products.

For most corporations already endorsing distributed objects, the prospect of developing objects on MVS is not appealing. In fact, most MVS developers do not possess the skills required to work in the object paradigm. For these corporations, gateway approaches are a very viable option: MVS-based applications that are not OO can be linked via non-OO middleware like DCE or MOM to objects that were developed using either Corba- or OLE/COM-based products. Such a solution requires an adapter, or "wrapper," to map object requests to a traditional middleware product. (See **Fig. 2**.)

However, the number of mainframe Corba products is limited. The list of mainframe Corba products includes IBM's SOM and Digital's ObjectBroker, which lacks support for key MVS subsystems like IBM's Customer Information Control System (CICS) and its Information Management System (IMS). Iona promises to ship an MVS version of Orbix next year, but there is concern about how well Iona can support an MVS-based version.

Finally, Microsoft signed an agreement about a year ago calling for Software AG, Reston, Va., to port OLE/COM to non-Microsoft platforms, including MVS. Software AG promises to begin shipping the MVS version during 1997.

Other platforms

At least one Corba-compliant product can or will run on virtually any operating system. For example, Digital's ObjectBroker product runs on VMS and IBM has announced that it is developing a version of DSOM for the AS/400.

Microsoft has said it plans to support some proprietary operating systems, but has yet to disclose any availability dates.

In sum, we believe that distributed objects on operating systems like VMS and AS/400 will have limited appeal. For the most part, companies running such proprietary operating systems will opt to use non-OO middleware to build an enterprise-wide distributed infrastructure.

Development environment

In the middleware selection process, corporate I/S officials must look closely at a product's ability to integrate with development tools. The selection comes down to a matter of priority. If an integrated application development tool is selected first, buyers must look at how a product can integrate with object middleware products. If the group is looking to first build a flexible infrastructure, it must evaluate what range of development tools is supported by any OO middleware candidate.

Several potential development language and tool choices can be made with the selection of object middleware. These are as follows:

- The integration of 3GLs and object middleware is driven primarily by the ORB vendors. Most object middleware products support C and C++, while some support Smalltalk. Generally, the selection of a language or an object middleware technology does not restrict the number of available 3GL options.

- Companies that opt for 4GLs are usually focusing on building a specific application, rather than building an enterprise-wide infrastructure for new development efforts. Some 4GL tools, such as Forté from Forté Software Inc., Oakland, Calif., incorporate built-in middleware capabilities. Such middleware can be substantial, but it is also embedded and proprietary, thus restricting the choices for more standard object middleware. On the other hand, this approach requires that application developers require only minimum middleware skills.
- The rush to the Internet and the fast increasing popularity of the Java development language from Sun Microsystems Inc., Mountain View, Calif., has also impacted object middleware. Several makers of Corba-compliant tools are adding Java support. And Sun's own Java Objects Everywhere (JOE) product is said to allow client Java applets to connect to NEO objects. However, JOE can only interoperate with Sun's own NEO objects today, although Sun officials say future versions will support other ORBs. Some ORBs, such as Iona's Orbix, now support Java.
- Finally, we believe that the OLE/COM enabled tools, which include Visual Basic, PowerBuilder and Delphi from Borland International Inc., Scotts Valley, Calif., deserve special attention. Visual Basic in particular is widely installed due to its simplicity and short learning curve. Unfortunately, these tools only work on the client end of distributed applications unless Windows NT is running on the server. None of these tools should be dismissed, however.

OLE/COM-Corba integration

The application architecture is another important factor to consider in the search for object middleware. Three-tier application architectures are becoming the preferred system design for many new development projects. Therefore, heterogeneous operating systems can be used for the different tiers.

A typical application can be comprised of: a Windows-based presentation layer using Visual Basic or another tool in combination with OLE/COM; a business logic layer based on Unix, using C++ to develop business objects that can be implemented as Corba objects; and a data access layer implemented as either a logical layer residing on the same system as the business logic layer, or as a physically separated server.

This typical application implies the use of OLE/COM and Corba. In fact, many developers are taking advantage of this approach, which grants Visual Basic programmers access to business objects developed in more advanced languages such as C++. Such an approach requires that few people within an I/S organization be dedicated to developing and deploying business objects. The remaining staff can access and use the objects from client applications developed in easy-to-use languages like Visual Basic. The importance of OLE/COM and Corba integration should not be ignored.

Advantages

The advantages of OLE/COM and Corba integration are that: developers get the freedom to choose the best tool for a specific job; personnel skilled in lower end tools like Visual Basic are readily available; developers can utilize the inheritance and portability offered by Corba; and Corba's multiplatform support can be utilized.

Disadvantages

Although you can split the required skills for OLE/COM and Corba among different development teams, using an integrated system forces an I/S organization to embrace two separate distributed object models.

Most Corba-compliant products do provide support for OLE Automation, but the extent of support can vary widely among the offerings. For example, other products require that users code most of the OLE Automation server logic, while Iona's Orbix product set includes tools that generate the OLE Automation code based on the definition of the Corba server applications.

Other options

Corporate I/S organizations could wait until the vendors of Corba-compliant products implement the OLE/COM-Corba interoperability specifications as defined by the OMG, or they could evaluate the Object Bridge solution from Visual Edge Software Ltd., St. Laurent, Quebec.

COM/Corba Bridge

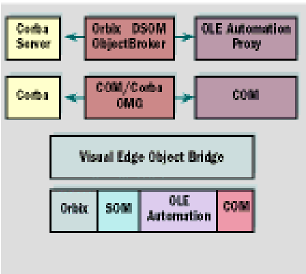

Fig. 3 Source: International Systems Group Inc.

An adapter such as the one from Visual Edge Software can provide an object system with automatic interoperability with all other object systems, including OLE Automation and COM, which already have adapters in place. In fact, the bridge acts as a "clearing house" between the different object systems.

Conceptually, Object Bridge is based on Visual Edge's Object Systems Adapter (OSA) specification. The OSA specification could be viewed as an engine capable of converting requests from one object model to another. Visual Edge Software has developed adapters for OLE Automation and COM, and is working with various middleware vendors like Digital, Iona and IBM to develop adapters for ObjectBroker, Orbix and SOM, respectively. (See **Fig. 3**.)

The Visual Edge product offers advantages over other integration solutions. For example, Object Bridge provides bi-directional communication between OLE Automation and Corba, or COM and Corba. The OLE Automation support implemented in a number of Corba-compliant products is uni-directional. Therefore, these products allow OLE Automation to call Corba objects, but not vice versa. Object Bridge also includes more comprehensive, robust errors and exception handling than the Corba-compliant offerings.

Standards

Corba and OLE/COM can both be called standards, although such designations come from different perspectives. Corba and its related specifications are selected and adopted by the OMG through an open process. The key to the OMG standards is the Interface Definition Language (IDL), a standard accepted throughout the industry and by other standards bodies. Therefore, any service provider can publish interfaces in a well-defined way.

The Corba standardization process brings tangible benefits to developers. For example, a client application programmer can use IDL descriptions to understand how to use the service. Then the programmer can generate stubs via the IDL compiler provided with a preferred Corba-compliant product.

However, a number of unresolved issues in the Corba specifications remain. For example, despite OMG promises to the contrary, there are still no compliance test suites. So corporations have obvious difficulties determining whether a Corba-based product being evaluated complies with the standard specification.

The general purpose of any standard is to ensure portability and interoperability of applications that use the environment. While Version 2 of the OMG's Corba specification produced a first step in providing the means for interoperability between multivendor ORBs, there are still several limitations to the standard, including a lack of coordination of naming and security systems across heterogeneous ORBs, and the widely varying mechanisms in the different Corba-compliant products for binding a client to a server.

Finally, vendors must still address issues like the management of a distributed object infrastructure. Unfortunately, the OMG has yet to set a standard management specification for Corba products.

Microsoft, of course, claims that OLE/COM is a standard, although it is a stretch to describe an infrastructure technology controlled by a single vendor as an open system. Further, developers of COM objects must build code that completely adheres to Microsoft specifications.

However, because COM meets two key selection criteria -- portability and interoperability -- it is hard to ignore the technology. No matter how much the technology is criticized, no one can argue that OLE/COM components cannot interoperate with other OLE/COM components. And the market penetration of OLE/COM makes the technology a de facto standard, at least in the Windows environment.

In addition, most of the top software vendors now support or have promised to support OLE/COM. Several OLE/COM-compliant business components covering a wide range of industries are already available. Therefore, OLE/COM could allow the assembling of separate business components produced by different vendors using different programming languages into a cohesive business application. So a single vendor's control over a key piece of technology could have some advantages for I/S organizations. Such a single vendor scenario can provide better integration of product suites, like Microsoft did with its Microsoft Office system, and can create a large component market. In addition, the existence of only one reference code base controlled by a single company guarantees a portability and interoperability of applications.

Going the object way

Selecting a distributed object infrastructure is clearly not an easy task. Before embarking on the selection process, I/S organizations must be absolutely sure of their objectives and whether they can be met using the technology under evaluation. Special considerations must be given to the fact that many I/S operations do not have large numbers of personnel with the skills required to design, develop and implement OO applications. Therefore, companies must first and foremost be realistic about their in-house skills and must budget realistically to bring in outside talent.

Second, object infrastructures must be selected for the right reasons. To use any of the Corba products just to connect different tiers of an application is a typical misuse of the technology. This is especially true when there are other, more mature middleware products that can better do the work. Selecting Corba for its ability to define higher level interfaces using IDL and for its ability to facilitate reuse of application components through support for inheritance are valid reasons. The same considerations are also key when looking at OLE/COM technologies.

Companies must also remember that neither object model possesses all the services required to be an infrastructure technology. Any decision comes down to how much additional in-house development is required to "plug in" the missing pieces. Our consulting firm has been working with corporations that are spending hard dollars to plug in those missing pieces.

Corporations can also look at integrating object middleware with other, more mature non-OO middleware technologies. The non-object middleware can supply some services not currently incorporated in object middleware. The expected integration of Iona's Orbix with IBM's MQSeries and Orbix with the publish/subscribe technologies of Tibco, Palo Alto, Calif., are examples of providing new functionality to Orbix developers.

In sum, object technology is maturing, but it is still missing pieces that are required for it to become a comprehensive infrastructure technology out of the box. Combining object middleware and non-object middleware can potentially produce a "best-of-breed" solution. Such a solution can be a good foundation for developing enterprise distributed applications.

With such a foundation in place, corporate developers can truly focus on developing business applications rather than having to deal with infrastructure issues. We believe that the world will see more best-of-breed solutions in the immediate future. These solutions will require cooperation among vendors, consultants and corporate I/S organizations. After all, no single vendor can do it all. At least for now.
