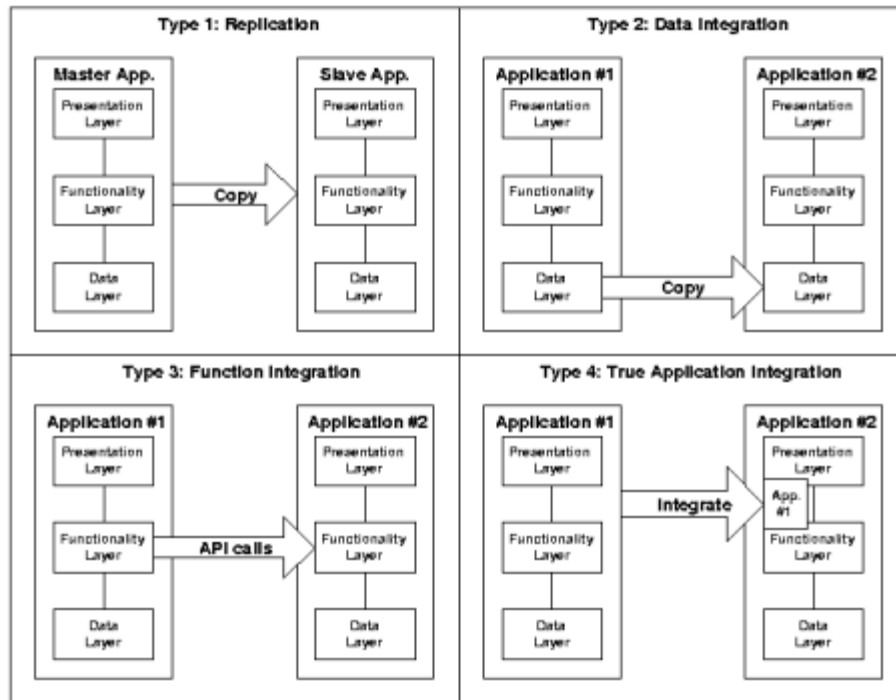


Web Services: The Next Evolution of Application Integration

Applications are typically built to automate business processes that are part of a discrete business function. In most cases, the users of an application fall within a single functional area in a company. However, company operations often involve multiple business processes across functional areas. This results in a requirement for integration between applications. A substantial amount of IT resources is spent each year on developing and maintaining integration points between applications. Vendors have recognized this fact, and a large market exists for application integration tools.

As the market has developed, four types of application integration have evolved: simple replication, data integration, function integration and, finally, true application integration. Replication is nothing more than simple duplication of all or portions of an application, creating a single master with multiple slaves. Data integration describes the use of tools that move data from a master application to one or more slave applications. Function integration involves one application programmatically invoking code that lies in another application. And true application integration results from making one application available within the context of another without actually duplicating the application itself. The following diagram describes these four types of application integration:

Four Types of Application Integration



True application integration will provide the underlying tools to build new business processes, creating new opportunities for collaboration. In addition, the integration of applications will be streamlined and simplified, reducing implementation and support costs. The double benefit of creating new value at reduced cost should not be ignored.

Theory Meets Reality

To illustrate the four types of application integration more clearly, let's use an example from two common desktop applications: Microsoft Word and Microsoft Excel. Often, spreadsheets that are built in Excel need to be embedded into a Word document. One way to do this involves copying a portion of the spreadsheet to the clipboard and pasting it as a graphic into your Word document. This is a simple method of integration, but one that must be repeated each time the spreadsheet changes.

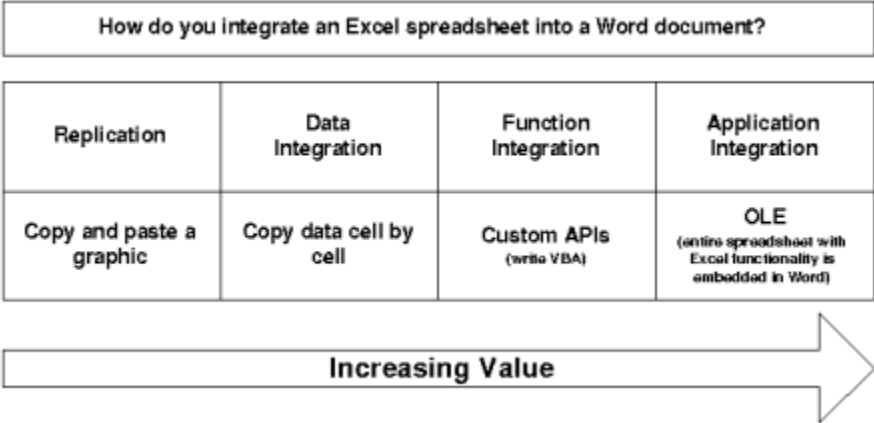
Another method involves creating a table in your Word document and copying data from each cell in the spreadsheet to the corresponding cell in the Word table. This allows for very fine-grained control of the data moving across, but it is very labor-intensive. A third method involves writing VBA (Visual Basic for Applications) code that copies the

appropriate content from the spreadsheet into the Word document. This method would even allow access to Excel macros and functions from within the Word document but again is rather labor-intensive.

The final and most common method involves embedding the Excel spreadsheet directly into the Word document using OLE (Object Linking and Embedding). The advantages with this method are ease of use, real-time data and the ability to access Excel functionality directly from the Word document itself. The only requirement is for Microsoft to build the underlying OLE technology, which it has already done.

The following diagram describes how each method described maps to each of the four types of application integration:

Application Integration on the Desktop



Methods of integrating enterprise applications are similar to those for desktop applications. Various methods exist for replicating all or portions of an application. This includes installing multiple instances of the application either manually or in an automated fashion with software distribution tools. Data integration is commonly achieved through a custom data extract, file transfer and load process. Extract, transform and load (ETL) tools exist to automate this process.

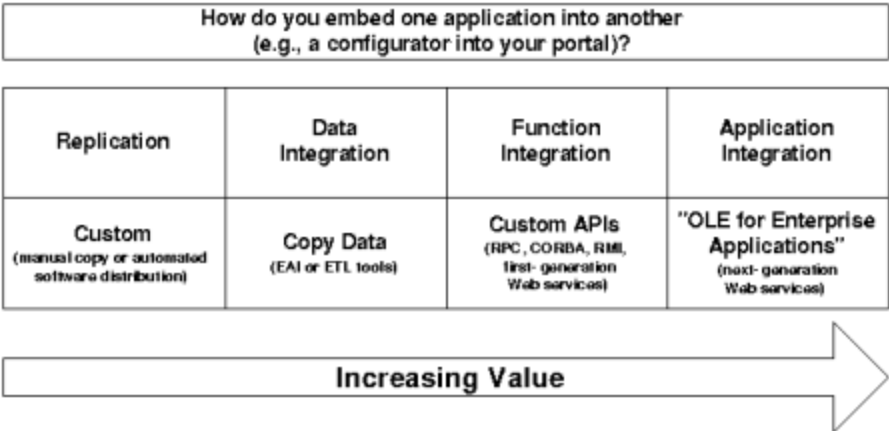
In addition, most enterprise application integration (EAI) tools actually perform nothing more than intelligent, reliable data integration. With EAI tools, changes in a data

element in the source application trigger a small extract that is sent and updated in the target application while applying the target application's business logic. EAI tools represent a solid, proven solution, but they are really nothing more than a form of advanced data integration.

Multiple attempts at function integration have been developed that allow one application to invoke an application programming interface (API) of another application. Various options include DCE RPC (Distributed Computing Environment Remote Procedure Call), CORBA (Common Object Request Broker Architecture), RMI (Remote Method Invocation) in Java and, more recently, Web services using SOAP (Simple Object Access Protocol). However, few solutions exist for true application integration or rendering a portion of one enterprise application within another. In the same way that an OLE for desktop applications provides significant advantages, an "OLE for enterprise applications" would prove extremely attractive.

The following diagram depicts each of these solutions, using our model of the four types of application integration:

Application Integration in the Enterprise

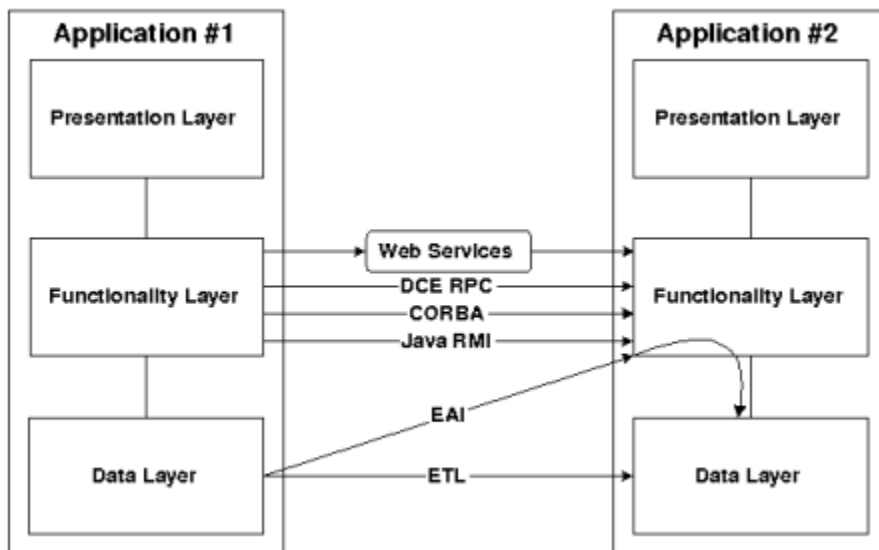


Enter Web Services

The recent development of Web services technology delivers significant promise for application integration. Web services technology provides a standardized method of publishing and subscribing to software services that are available over the Internet. A SOAP interface can be layered on top of a new or existing piece of code to allow it to be accessed by other applications over the Internet.

Applications can locate these services using UDDI (Universal Discovery, Description and Integration) and determine the interface definition of the service using WSDL (Web Services Description Language). In essence, Web services technology can be thought of as an evolution of function integration technology such as CORBA, as depicted in the following diagram:

Common Methods of Integrating Applications

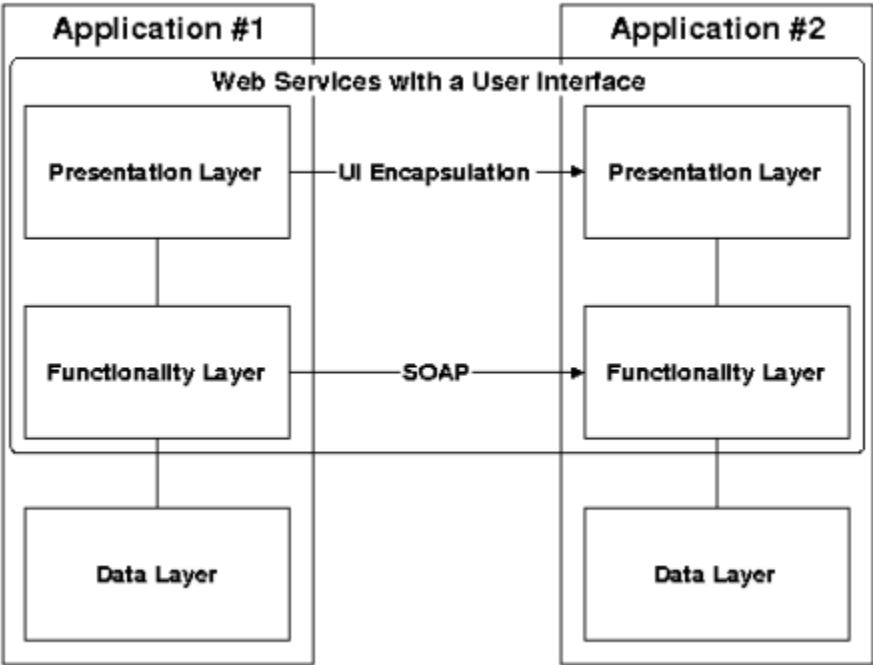


A major advantage of Web services over CORBA is that Web services do not require integration of an ORB (Object Request Broker), a task that is not for the faint of heart. The underlying transport protocol behind Web services is based on XML (eXtensible Markup Language) over HTTP (HyperText Transfer Protocol), so it is fully compatible with firewalls and the Internet.

Web services have also gained wide industry support in a very short amount of time. Hewlett-Packard, IBM, Microsoft, Oracle and Sun Microsystems have all announced major Web services initiatives.

However, Web services fall squarely in the function integration bucket. In order for Web services to be used for true application integration, several extensions must be made to the collection of Web services specifications. Web services currently lack a mechanism to encapsulate a user interface, which is key to being able to package an application and embed it into another application. If a user interface were packaged along with the method invocation, then the integration model would change to the diagram below:

Web Services that Span Presentation and Functionality



By extending Web services to encapsulate the user interface with the method invocation--spanning both the presentation and functionality layers--Web services then have the core ingredients to be classified in the fourth bucket of true application integration.

Several vendors have recently banded together to develop a standard for delivering Web services encapsulated with a user interface. The Web Services User Interface (WSUI) initiative, announced in June 2001, defines the concept of a view, which represents a single display screen of a Web service. Views use XSLT (eXtensible Stylesheet Language Transformation) to transform the results of a Web service invocation or a script into display code such as HTML (HyperText Markup Language) or WML (Wireless Markup Language).

WSUI requires that a "WSUI container" be implemented on the subscriber's site in order to construct the user interface. The WSUI specification is still in its infancy--the initial working draft has not yet incorporated any comments from its working members. Because of this, its value to large enterprises will not be realized for some time. However, WSUI holds promise and deserves significant attention because it addresses a gaping hole in Web services technology.

A key consideration that will affect the adoption rate of a user interface on top of Web services technology is its ease of use. With OLE on the desktop, the packaging and installation of the user interface and functionality is easily done without deep technical knowledge: a user merely cuts a portion of the Excel spreadsheet to the clipboard and pastes it as an object into the Word document. The creation and integration of a "package" containing the user interface and business logic of an enterprise application is another problem that is not currently being addressed by Web services. WSUI has a definition of this package but lacks tools to deal with the underlying complexity.

Web services also lack facilities for security, which is another major hole. Many pundits are looking to Web services to truly bring business-to-business integration (B2Bi) into the mainstream. However, the application of Web services for B2Bi will be limited if services for authentication, encryption, access control and data integrity are not available.

Currently, Web services technology cannot certify the identity of the publisher or consumer of a Web service. In addition, there are no facilities to restrict access to a Web service to a group of authorized users. Several initiatives have been launched to address some of these shortcomings.

The XML-Based Security Services Technical Committee from the Organization for the Advancement of Structured Information Standards (OASIS) is working on a specification for Security Assertion Markup Language (SAML). This specification allows organizations to exchange authentication, authorization and profile information securely with their partners. OASIS is also working on XACML (eXtensible Access Control Markup Language), a specification that enables organizations to limit access to services to authenticated, authorized users.

Other system standards are also being developed to ensure uniform methods for implementing required low-level security services. One of the most significant is the XML Key Management Specification (XKMS) proposed by Microsoft, VeriSign and webMethods. The vast majority of security services deployed on the Web today are built on top of public key infrastructure (PKI). Currently, creating products that support PKI is a difficult task that requires deep, specialized knowledge. XKMS aims to reduce this complexity by making trust utilities available on the open Internet. Developers would use XML to access these services, allowing them to bypass the complexities associated with PKI and focus on their applications.

Vendors have identified the key technical issues that inhibit Web services from supporting true application integration in B2Bi environments, and efforts are under way to address these issues. Key standards have been proposed, but they are still months away from being finalized. Without support for user interfaces, Web services will only facilitate function integration. Lacking strong B2B security, Web services will be limited in their value to B2Bi, even if the ability to encapsulate the user interface were available.

What You Should Do Today

Although Web services technology is still in the realm of the early adopters, it represents an attractive alternative in the application integration space. The technology provides a wealth of new opportunities because of its standardized, Internet-friendly method for integrating function calls.

With Web services, applications can be quickly and easily integrated into a portal or syndicated to your third-party business and channel partners. To apply the technology in the most appropriate manner, it is important to understand that the first generation of Web services technology is an option for function integration only.

The next generation of Web services technology will address the issues of user interface encapsulation and security. Vendors will develop products that support user interfaces on top of Web services and strong B2B security. When this happens, Web services will provide an extremely attractive option for true application integration between enterprises.

Visionary technologists can position themselves to reap the benefits of application integration within an enterprise and between enterprises by learning more about Web services now and starting pilots internally. Once your organization becomes familiar with Web services, you should analyze your current business processes within your company and between your partners to identify opportunities to move beyond basic data and function integration to capitalize on true application integration.