Message Brokering - Should You Bet Your Business on It?

by Les Yeamans

Management Introduction

The challenges have continued to increase for Information Technology (IT) managers due to the implementation of a widening variety of hardware/operating system platforms, DBMSs, development tools and packaged application solutions. Applications developed separately, based on the individual needs of users, have become "islands of automation." The diversity has increased with the advent of component object brokers, the intranet/internet and Java and JAVABeans. This dizzying array of choices made up of both proven mission critical products and less mature, newer technologies leaves CIOs and senior MIS management with a daunting problem: How to integrate new applications with existing legacy and packaged applications across the enterprise without risking the farm?

Forward thinking IT executives in industries such as Finance and Health Care are solving this application integration problem by implementing a relatively new architecture based on a "Message Broker"(MB). An MB is a set of software components that allow applications to communicate with each other through the non-invasive bi-directional exchange of messages. While these early adopters have invested substantially in home-grown capabilities, a set of small companies (including NEON, TSI, HubLink, Active SW, MINT and Open Horizon) have come to market with a variety of first generation MB products.

This analysis will give an overview of application integration, describe what a Message Broker is and how it works, and assess the business benefits and risks of deploying MBs, including what's needed before IT executives should bet their business (and careers) on them.

Application Integration Overview

Applications get developed in order to accomplish a specific business objective in a measured time frame. In a typical large organization, hundreds of applications are developed by different teams of people using a wide assortment of operating systems, DBMS's and development tools. In each case, the specific requirements are satisfied without regard for integration with any other applications.

There are some powerful trends driving the market for application integration:

- Significant developments in peer-to-peer networking and distributed processing have made it possible for businesses to better integrate their own functional departments as well as integrate with their partners and suppliers;
- The intranet/internet explosion is fueling the demand for a new class of "human active" applications that require integration with back-end legacy applications;
- The tremendous growth in adoption of Enterprise Application Software packages, such as SAP R/3, also requires integration with back-end legacy applications;
- Message Oriented Middleware (MOM), products such as IBM's MQSeries message queuing product, are becoming increasingly popular. Once customers realize the benefits of simple one-to-one application **connectivity** with MQSeries, their interest in many-to-many application **integration** increases significantly.

As the need for businesses to integrate grows, the number of IT dollars spent on integrating applications is increasing rapidly. According to various industry analysts, the need for "mission critical" Application Integration will drive the combined market for Message Oriented Middleware (MOM) and Message Brokers(MB) to grow from $300 million in 1997 to over $700 million in 1999. According to an IBM survey of larger customers, "70% of all code written today consists of interfaces, protocols and other procedures to establish linkages among various systems." Savvy MIS executives can clearly see the dollar savings to be gained by acquiring off-the-shelf software to satisfy as much of this requirement as possible.

What is a Message Broker?

A message broker is a software hub that records and manages the contracts between publishers(senders) and subscribers(receivers) of messages. When a business event takes place, the application will publish the message(s) corresponding to that business event. The broker reviews its lists of subscriptions and activates delivery to each subscriber for that message type. Subscribers receive only the data to which they subscribe. A message published by one application can be subscribed to by multiple consumer applications. Similarly, a subscribing application can receive messages from multiple publishing applications. Exhibit 1 provides an overview of message broker interactions.

Here’s how it works:

Before applications can publish or subscribe to messages, they must register their interest with the broker. There are two different methods for applications to register interest in a message type. There is subject-based addressing and message-content filtering. In subject-based addressing, the broker uses the subject to identify and route the message to all parties expressing interest in that subject. The subject is a word used to describe the contents of the message. For example, a subject of the name "hr. emp. new," would serve to distribute information (name, address, employee number, etc.) on a newly hired employee.

In message content routing, subscriptions are made based on the contents of fields within the message. The subscriptions can be based upon the message type and/or specific selection criteria relative to a field within the message. For example, a loan approval application could subscribe to all purchase orders over $100,000.

One advantage to having two publish/subscribe paradigms is that the need to address messages to individual subscribing applications is avoided. Additionally, new subscribing applications can be added without any changes to the publishing application.

The publishing/subscribing broker uses a robust delivery vehicle for the actual distribution of messages between applications. As mission critical messages travel over a combination of external and internal networks, the systems software ensures that messages are never lost or duplicated in the event of network failures. Typically, an asynchronous message delivery capability is provided which uses store and forward message queuing. In this paradigm, the queue-to-queue transfer takes place in pseudo-real time when the subscribing application is available. If the subscribing application is unavailable, the message is stored in a persistent queue for later delivery.

To be effective, the message delivery vehicle must include a business transaction coordination function. A business transaction is typically made up of several units of work. Each and every unit of work must

complete in order for the transaction to occur. If even one unit of work fails, the whole transaction fails, and all completed units of work must then be reversed. These transactions are long-running and require message-based updates to multiple databases. The business transaction coordination function provides this managerial support.

Two other important components are the rules-based engine and the data-transformation component. The business rules engine allows you to process messages based upon the unique requirements of the business. Typically, business rules engines provide a visual front end to avoid the need for programming in a procedural language. With this flexible approach, changes in business processes can be easily reflected in a modified rules configuration.

The data transformation component is used to develop application-specific adapters. These adapters convert the data formats and applications semantics from the sending application to the receiving application. There are many conversion requirements. They range from basic data transformation to resolving the incompatibilities that exist between the structure (syntax), meaning (semantics) and timing of the information that must be shared.

There are two main strategies for application adapters. One strategy is to convert all of the data from the source and sync applications to a standard canonical form. Messages move from the source adapter to the sync adapter in this standard form. At the sync adapter, the messages are converted to the format of the sync application.

The second strategy for application adapters is to automatically convert the format and semantics from the sending application to the receiving application in one step, without any intermediate formats. In this approach, only one adapter is required for two applications to communicate with each other and it can be integrated with either of the applications.

The rules based engine and the data transformation component work together to reconcile the differences between applications. For example, before two applications can be integrated around an order, the business rules regarding the processing of orders must be defined within each system. Within Application "A," an order might be comprised of a collection of data from multiple files and databases; whereas within Application "B," an order might exist as an individual message nested within a larger file of business transactions. The difficult challenge is to resolve the incompatibilities between the structure of the data and the underlying content of an order as defined in each application.

Benefits of a Message Broker

There are a number of potential business benefits that message brokering provides:

Ease of application integration -- With message brokers, the integration of new applications with existing legacy or third party applications can be performed in a shorter period of time. The integration can take place without any need for understanding the internal structure and design of each application. By focusing on the interface as messages, existing applications can be integrated with minimal disruption.

Support for electronic commerce -- As businesses begin to automate the supply chain of vendors and

partners, there is a need for their independent applications to communicate in a loosely-coupled manner. This is precisely the essence and strength of message brokering. The message broker is completely congruent with the business need.

Support for continued heterogeneity -- As new technology has evolved, new architectures have been developed and heterogeneity is increasing over time. A methodology such as message brokering is designed to accommodate today's heterogeneous world and will be useful in the future. New, differing applications can be added over time as either publishers or subscribers, without any changes to the existing applications in the message broker.

In summary, message brokers have the potential to provide a least-common-denominator approach to integrating heterogeneous applications within an enterprise. Users can choose the best technology for each individual application whether it be JAVA, ACTIVE X, CORBA, et al., without concern for how that application will integrate with other applications in the enterprise. Message brokers thereby bridge the gap between applications of the future and the disparate and complex products and technologies that currently exist in today's application catalogues.

Risks of Deploying Message Brokers

While there are many benefits to adopting a message broker strategy, we must keep in mind that there are also potential pitfalls.

The very strengths of the message brokering in terms of its loose coupling flexibility, may also be its greatest weakness -- a double-edged sword. The nature of message broker software, as we've discussed, is very generalized. Because it is designed to handle so many different conditions, testing all possible end-to-end code paths is a Herculean task. When undetected bugs exist in the software, messages may be lost, delivered twice or delayed. The damage from such "accidents" would be most keenly felt in enterprises where message brokers are used to integrate mission critical transaction processing applications. For instance, in financial transactions, the delivery of one single message could be worth millions of dollars; concomitantly, its non-delivery or delayed delivery could result in the loss of millions.

A second risk to a message broker implementation is the possibility that foreign applications will introduce unauthorized messages to the broker. This may also result in loss. For example, in the banking industry counterfeit messages can be published causing the withdrawal or misappropriation of funds.

A third risk of message broker implementation is the classical, "single point of failure." Message brokers are typically implemented in a "hub and spoke" architecture. This means the majority of message traffic passes through a few central hubs. In the event of an outage or a physical disaster to one of the hubs, the mission critical operations of a business could come to a grinding halt.

Another problem with distributed hubs is the difficulty of managing the message broker complex. Because a message broker integrates so many different business applications into a few consolidated hubs, the talent and expertise required to manage and administer a message broker complex may be unattainable.

Management Conclusion

The Application Integration market will experience tremendous growth over the next several years. IT Management is searching for a sound business strategy for integrating the new with the legacy. Estimated investments in legacy environments are as high as 4 trillion dollars. Clearly, re-use is one of the hottest topics for CIO's.

The message brokering architecture shows great promise as a technology that can satisfy many of the requirements for application integration. Its loosely coupled architecture sounds ideal for today's ever increasing heterogeneity. With the use of innovative adapter technology, message brokering seems prepared to handle the problems of yesterday, today and tomorrow. If only it were that simple!

The potential risk exposure is large whenever technology is applied to the "nuts and bolts" mission critical transaction processing applications of an enterprise. The "problem" for message brokering is that it manipulates mission critical information. In relative terms, the message brokering technology is new. While it is true that some early adopter companies have had great success with the concept, much more is needed before mesage brokers can enter the mainstream. Some of the critical elements that must be added or improved are:

- comprehensive management, including centralized monitoring, operation and configuration;
- message tracking and manipulation;
- security covering authentication, authorization, privacy, non-repudiation and auditing;
- disaster recovery, fail safe rollover, message replay and dual-site logging.

As these capabilities are added or improved, message brokers will be ready for widespread adoption.

Until then, IT executives will no doubt endorse their use in selected tactical situations where the speed and ease of integration is compelling and comprehensive testing can be performed before production use. Just like MOM and other new paradigms, the transition will be gradual over the next several years. But I wouldn't bet my business or career on it just yet!

Back to the Message Q Homepage