Understanding Integration Strategies

By Hisham Alam

A smorgasbord of application technologies has accumulated in the relatively short history of computing. Enterprise Application Integration (EAI) products attempt to make sense of the technology tangle. They're the glue that makes different applications talk to each other. So what's the best way to take advantage of them?

The set of technologies to deal with include:

- Batch run applications
- Character-based User Interface (UI) applications
- Hierarchical databases
- Graphical User Interface (GUI)-driven client/server applications
- Relational databases
- Browser-driven applications
- Java applets
- Mobile device applications.

These technologies range in user interface look and feel, application standards, and database technologies. The EAI glue has to contend with all of them.

Internet commerce poses another set of EAI challenges. For example, in the business-to-consumer (B2C) world, the UI experience — its presentation, functionality, ease of navigation, and speed — must leave nothing to chance. Otherwise, the site gets a bad rap and traffic drops. In the business-to-business (B2B) world, there are several forms of automation. Open B2B exchanges and markets require much of the same UI experience of the B2C environment. In closed B2B environments, which facilitate access to a limited number of business partners, training can compensate for the difficulties of a complex application user interface. At the other extreme, consider B2B applications that rely on the exchange of data across organizations, such as eXtensible Markup Language (XML) and Electronic Data Interchange (EDI). For these, the UI experience is unimportant, but reliable, flexible, and integrated data transmission is crucial.

One of several approaches can meet the EAI challenges for these differing requirements. The forward integration model emphasizes the user experience and places the EAI server close to the user interface. The back-end integration model builds the EAI server as the corporate information backbone. A third approach, the hybrid integration model, combines the other two.

The EAI environment extends beyond integration. The Operational Data Store (ODS) plays a significant role in leveraging EAI benefits. The EAI environment

also integrates Customer Relationship Management (CRM). Finally, security is a key component of the EAI environment.

The Forward Integration Model

The forward integration model is the approach of choice when a business process initiated by an end user will generate multiple transactions across multiple applications and near real-time response is needed. Consider, for example, that when a policyholder or insurance agent files a claim, EAI allows updates to both the policy administration and claims system in near real-time. The user initiates both transactions behind the scene from a common user interface, accessible over the Internet.

Figure 1 provides an overview of the forward integration model. As shown, the user accesses a Web browser interface. The Web application server receives the user's requests and hands them off to the EAI server. The Web application server also processes the response to the user. The EAI server uses business rules to apply transactions across multiple applications. These applications may represent legacy applications or a set of new applications that represent best-of-breed functionality in their respective areas (e.g., policy administration, claims, billing, agent commission compensation, etc.). The EAI server may also interface with an external service such as credit verification.

The forward integration model can funnel transactions through a single user interface and connect to multiple applications, so it's efficacious in a B2C environment. It's also used to integrate applications without a Web browser interface. This is achieved by front-ending the EAI server with a Web application server.

The forward integration model lets the EAI server route transactions in near real-time. This requires application-to-application (A2A) integration as opposed to integration at the data layer. This may require development of customized application adapters.

The forward integration model permits considerable control over the types of permitted transactions, but this comes at a cost. The EAI server must mimic the transaction logic of the applications it serves. Complex business rules must be developed and data must be mapped across multiple applications.The forward integration model lets you seamlessly augment application features that may be missing from your core applications. For example, if your applications don't handle online credit verification and authorization, this facility may be integrated with the EAI server.

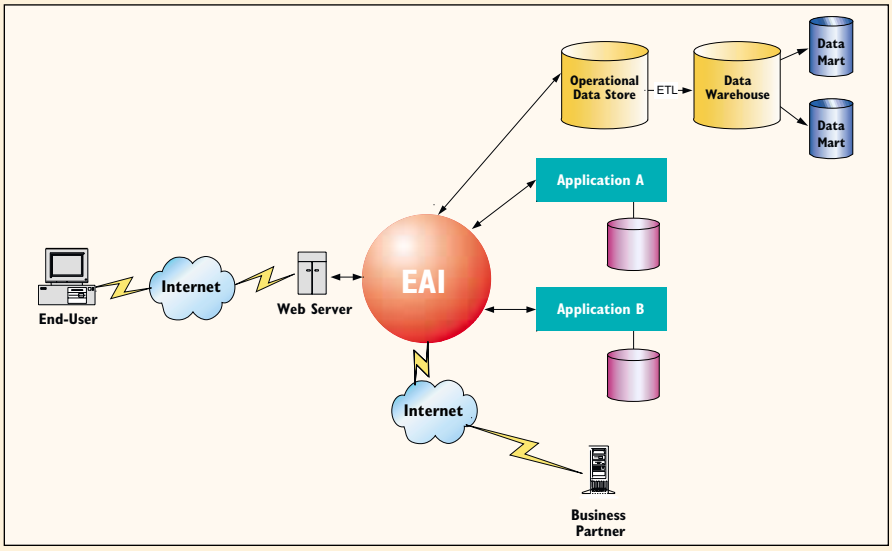

Figure 1 — The Forward Integration Model

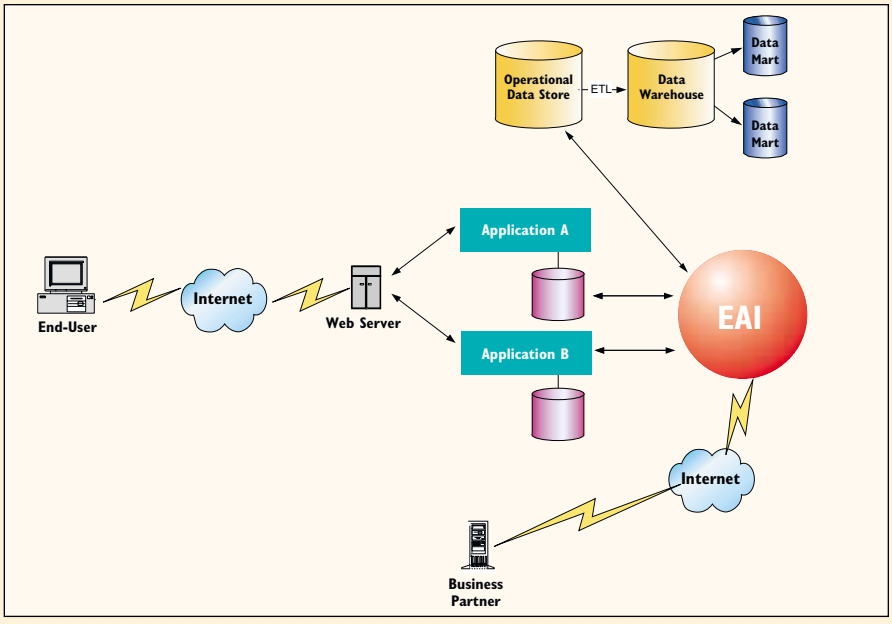

Figure 2 — The Back-End Integration Model

Back-End Integration Model

The back-end integration model facilitates automated exchange of data across multiple applications. Think of it as an information backbone. Figure 2 is a rendition of the model. The user accesses the applications, which must be Web-enabled, directly through the Web application server. The user interface may integrate the presentation of the applications, but the applications handle each transaction separately. Application integration occurs in the back-end. For example, when a policyholder or insurance agent files a claim, the claims application processes the claim transaction. Subsequently, when an adjuster works the claim, he or she can trigger the EAI server to retrieve the associated policy information from the policy administration application.

The back-end integration model assumes a passive role. EAI functions as a conduit for information across applications and data sources. This model services both A2A and application-to-data integration. Integration with databases tends to be simpler. That's because the EAI server doesn't require application-specific Application Program Interface (API) calls.

The back-end integration model is less complex to implement since the EAI need not mimic application transactions

and maintains a relatively simple set of business rules. The approach efficiently serves B2B applications requiring exchange of data across business partners using a store-and-forward messaging model. For example, the response to a request for a motor vehicle report from an information ordering service may lag several days. The auto insurance policy can be generated and sent to the customer pending the final outcome of the report. The policy administration application can batch the requests and send them through the EAI server during off-hours. Another use of this approach is uploading or downloading data between the different applications of business partners.

Hybrid Integration Model

The hybrid integration model is a combination of the forward and back-end integration models. Figure 3 depicts the approach. The user accesses an application as well as the EAI server through a browser-based user interface. The front-end application performs some functions while the EAI server routes others to back-end applications. In our insurance example, an agent commission reporting application (accessible to authorized agents via valid identification and password) could be accessed directly by an agent, while the policy and claims administration applications are separately updated by the EAI server. The server also updates the commission reporting database when a policy transaction occurs.

The hybrid model maintains most of the features and complexity of the forward integration model. It's used when most of the requirements call for the forward integration model and some additional back-end integration is necessary. It's superior to have separate EAI servers to provide forward integration and back-end integration when the load requirements are high for each. In this scenario, the forward integration server will process high-volume, near real-time requests while the back-end server will process a combination of ad hoc, near real-time, and batch requests.

Role of the Operational Data Store

The ODS provides different functions in an EAI environment and may be used with all three integration models. The ODS may be used as a repository that integrates operational data from different applications and facilitates reporting. The EAI server updates transactions across applications and the ODS, which maintains data structures that enable integrated reporting across applications. This obviates the need to access separate applications to create a report. Application performance is also not burdened by reporting functions carried out during daily operations.

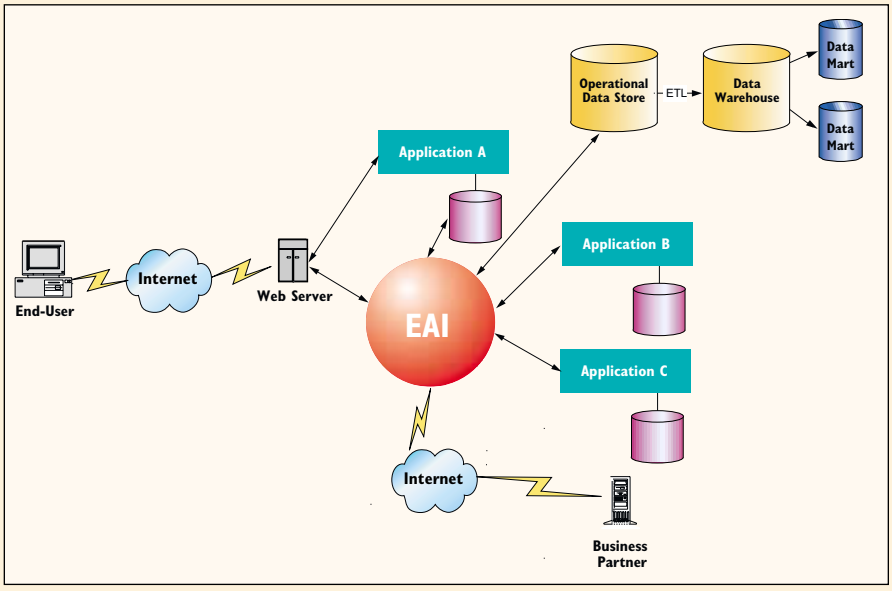

Figure 3 — Hybrid Integration Model

The ODS maintains a limited history of operational data ranging from one month to a year. Older historical data is extracted and loaded into the data warehouse on a regular basis. An Extraction, Transformation, and Loading (ETL) tool may be used for this purpose. The data warehouse performs analytical queries and reporting. Subject-area data marts may be created to serve particular business requirements. For example, insurance agents may use the ODS to view reports about a policyholder. The report will provide cross-application summary information about the policy held, claims made against policies, billing, and commissions.

The ODS may also be used as repository for reference data. It can be used to hold customer data that crosses multiple application boundaries. When a transaction involves a new customer, the EAI server references the ODS to see if the customer exists and retrieves the current information for confirmation. Customer updates are applied to the ODS and to applications processing the transaction. The ODS thus provides one final source for customer and other key reference data. For example, a financial services company may offer personal banking or insurance products. A centralized customer database will allow for common customer lookup for transactions involving both sets of applications.

The ODS is also used as an interim repository of operational data from one set of applications to subsequently feed other downstream applications. The ODS functions like an "information router," feeding data to downstream applications. This approach doesn't support a near real-time environment. In reality, the ODS provides little value-add other than as a staging area for downstream applications. The approach can be improved upon by properly integrating the downstream applications via an EAI server.

Integrating CRM

Including ODS in the EAI environment provides a foundation to build an integrated CRM solution. There are different types of CRM solutions.

Customer call center management provides the ability to manage inbound calls through call routing, Computer Telephony Integration (CTI), and Interactive Voice Response (IVR) technologies. The goals of call center management are to improve customer retention by providing high levels of service, increase revenues through cross-selling, and reduce costs through automation. In this model, the ODS functions as the reference hub for customer information. The ODS can maintain customer data across multiple

applications. For example, if a customer calls to purchase a homeowner policy, a simple lookup of the ODS will provide information about an automobile policy the customer may have without having to prompt the customer.

Sales force automation can be integrated with various applications, including the ODS. For example, the policy quoting process can be integrated with sales force automation to keep track of when and where to call and follow-up with the customer.

Marketing campaign management can be integrated with the ODS to use customer information to launch and track marketing promotions and campaigns.

Real-time personalization, used in Web and mobile applications, allows the user to be personally greeted and provided with information relevant to his or her interests. Personalization improves the chance for cross-selling products. At the heart of personalization is the ability to create simple relationships. When buying a book on the Web, for example, you may be prompted with other titles that may be of interest to you. Suggested titles may be based on sales to other people who also bought the book. Personalization relies on customer analytics and the ODS for customer data.

Customer analytics is a means to correlate customer behavior patterns and use that information to cross-sell a product or offer a sales promotion. Results of customer analytics may be used to launch a marketing campaign or improve personalization. The ODS is a ready data source to assist.

The synergy between the ODS and CRM is greatest when the definition of customer is broad. A broad definition provides greater opportunity for customer cross analysis. For example, in the insurance domain, a customer can be thought of as a party, and a party can include the insured, beneficiaries, agents, co-insured, dependents, policyholder if not the insured, active and inactive accounts.

Ensuring Security

Security serves multiple functions:

- Authentication ensures that the concerned party is really who they say they are.
- Authorization lets the concerned party access the application they're requesting.
- Confidentiality protects sensitive data in transmission and storage.
- Integrity ensures that data hasn't been altered during transmission or storage.
- Non-repudiation is required to prove a transaction occurred.
- Single sign-on permits only one point of entry.
- Provisioning permits creation and management of identities from a central point.

There are several security control points in an EAI environment. In the forward integration model, the first security control point occurs before the EAI server is invoked. When a user accesses a Web application, the Hypertext Transfer Protocol (HTTP) request is intercepted by a privilege management system and the user is prompted with a logon identifier and password. The privilege management system authenticates the user and determines whether he or she is authorized to access the application. After valid authentication and authorization, the user can issue application requests that the EAI server routes and handles. The privilege management system also passes the user identifier to the EAI server. The EAI server uses the identifier to access the applications it services in one of two ways:

- The EAI server can establish a common session with an application and multiplex multiple requests over that session. The burden of non-repudiation falls on the EAI server to log an audit trail of transactions performed. With this approach, intrusion detection is compromised. If an intruder uses the common session to break into the application, it's difficult to trace his or her activity.
- The EAI server can connect to applications on a per-user basis. This ensures better security control, but at the cost of system performance, which is degraded as users access the system and require connections. Additionally, user identities must be created in each of the applications being integrated.

An alternative approach provides more granular, centralized security control. Application authorization is embedded within the EAI server. In this scenario, the privilege management system authenticates the user but doesn't perform application authorization upfront. When a user initiates an operation that accesses an application, the EAI server queries the privilege management system to validate the user's authorization. If denied access, the user gets an error message. This approach centralizes application access security and makes it easier to administer security policy and provide user identities across multiple applications.

Using the Secure Socket Layer (SSL) protocol ensures transaction confidentiality and integrity. SSL provides for the initial exchange of public keys between the browser and server and the transmission of data using encrypted Internet Protocol (IP) packets.

Summary

Developing an enterprise integration strategy involves the selection of an appropriate integration model and beyond that, assessing the role of the ODS, CRM, and security in the overall EAI environment. eAI

About the Author

Hisham Alam is a principal consultant with PricewaterhouseCoopers. He has extensive experience in implementing strategies and solutions in Web-centric applications, data warehousing, and technical architecture. e-Mail: hisham@hisham.net.