JOURNAL OF OBJECT TECHNOLOGY

Online at www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2003**

Vol. 2, No. 2, March-April 2003

Pattern Driven Solution Engineering

Mahesh H. Dodani, IBM Software Group, U.S.A.

1 PATTERNS FOR SOFTWARE ENGINEERING

Over the last decade, application development has depended largely on the use-case driven OO software engineering methodology (http://www.aw.com/catalog/academic/product/1,4096,0201544350,00.html) along with the many patterns for designing reusable software (http://www.awprofessional.com/catalog/product.asp?product_id={E1CD5BE7-E008-481B-8D0C-8E80CE9978F9}.) Most of the innovations during this time have been targeted primarily at variations of the methodology and additions to the patterns catalog. However, over the same time period, the applications themselves have gotten more complex, moving from single applications addressing a specific functionality or set of requirements to an integrated set of applications representing an enterprise e-business.

What is an e-business? It is a catch-all term that identifies an enterprise which has transformed its business by leveraging web technologies to reengineer business processes, enhancing communications and lowering organizational boundaries with its customers and shareholders (across the internet), employees and stakeholders (across the corporate intranet), and its vendors, suppliers and partners (across the extranet.)

This e-business transformation is evolutionary and complex. The typical evolution for an enterprise is to move from providing access to enterprise applications and information, through integration of systems and applications in the enterprise, to an on-demand adaptive enterprise which can optimize operations and dynamically adapt to the needs of all its constituents.

Furthermore, in going through this e-business transformation, an enterprise must address the needs of all its constituents, resulting in a very complex application model. For example, in addressing the transition of governments into an e-business, Figure 1 shows the complexity of the model. Note the different constituents on the left of the figure accessing the enterprise through a portal using various devices, the complex enterprise services and the delivery of these services to the various constituents, the integration of the enterprise via adaptors to various applications in both public and private sectors, and the services needed to support the integration and connectivity. This comprehensive model sits on top of a complex infrastructure that provides the supporting "plumbing" and common services.

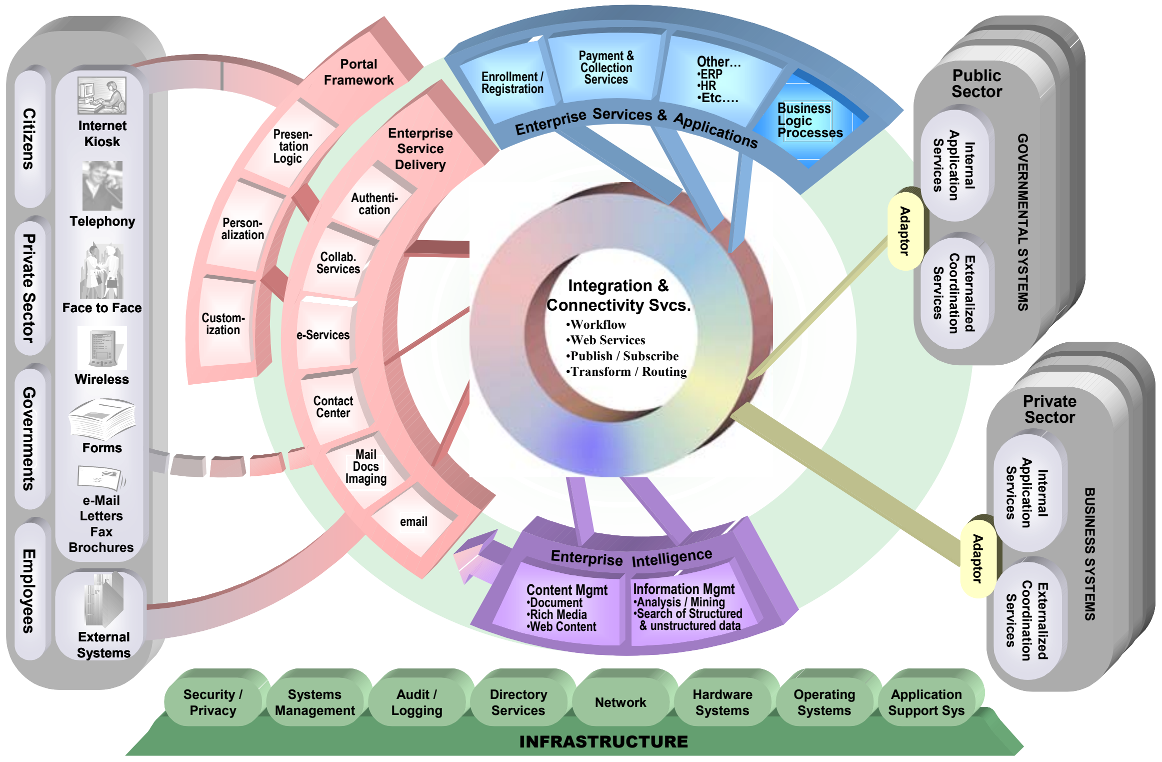


Figure 1: A Complex e-business Transformation Model

Such complex e-business solutions impose their own unique challenges on top of the difficult well-understood challenges of developing OO applications, and need to answer the following questions:

- How to integrate with legacy systems both within the enterprise and with systems outside the enterprise?
- How to satisfy the needs of all the users of the enterprise fast?
- How to adapt to rapidly changing technologies, products and architecture/design frameworks?
- How to handle the acute shortage of key skills and experience in building these complex solutions?

The approach to engineering complex solutions adopted by IBM and described in the rest of this paper is to capture the experiences and approaches that have worked to solve problems in a series of patterns targeted at different levels (ranging from the business level, through the architecture and design levels, to the runtime level), and then integrating these patterns into a cohesive methodology that drives a business requirement into a technical solution.

2 PATTERNS FOR E-BUSINESS SOLUTIONS

"Each pattern describes a problem that occurs over and over again in our environment and then describes the core of the solution to that problem in such a way that you can use this solution a million times over without ever doing it the same way twice" – Christopher Alexander, A Pattern Language, 1977 (http://www.patternlanguage.com/index.htm.)

The primary objective of the patterns for e-business is to provide a consistent set of patterns that can help drive business priorities and requirements into proven technical solutions. Experience in this area has shown that only 20 percent of an application development project is unique to the business; 80 percent of a project can be approached with proven best practices and established techniques.

IBM Patterns for e-business (http://www-106.ibm.com/developerworks/patterns/) define a set of proven, reusable architectures that can drive the design, development, implementation and extension of e-business applications. These patterns have been culled from the documentation and analysis of thousands of successful IBM e-business projects. They match business challenges with Business and Integration patterns, use proven Application and Runtime patterns, populate the Runtime patterns with pre-tested Runtime Product Mappings, and establish best practice guidelines for application design, development and management.

Figure 2 below summarizes the pattern driven solution engineering approach. Business requirements are used to identify the applicable business pattern based on the primary business actors, and the common interactions between the users, business and data. There are four business patterns which form the basic building blocks for the e-business solution:

1. Self Service: which facilitates users to access business transactions at any time.
2. Collaboration: which facilitates users to work with one another and share data and information.
3. Information Aggregation: which facilitates data from multiple sources to be aggregated and presented across multiple channels
4. Extended Enterprise: which facilitates the integration of data and processes across enterprise boundaries.

For the development of complex systems, composite patterns allow any combination of these business patterns, plus one or more common integration patterns to address the needs of front-end and back-end integration:

- Access Integration: which facilitates a consistent user interface by enabling access from multiple channels (devices) and integrating common services.
- Application Integration: which facilitates integration to core business applications, processes and data.

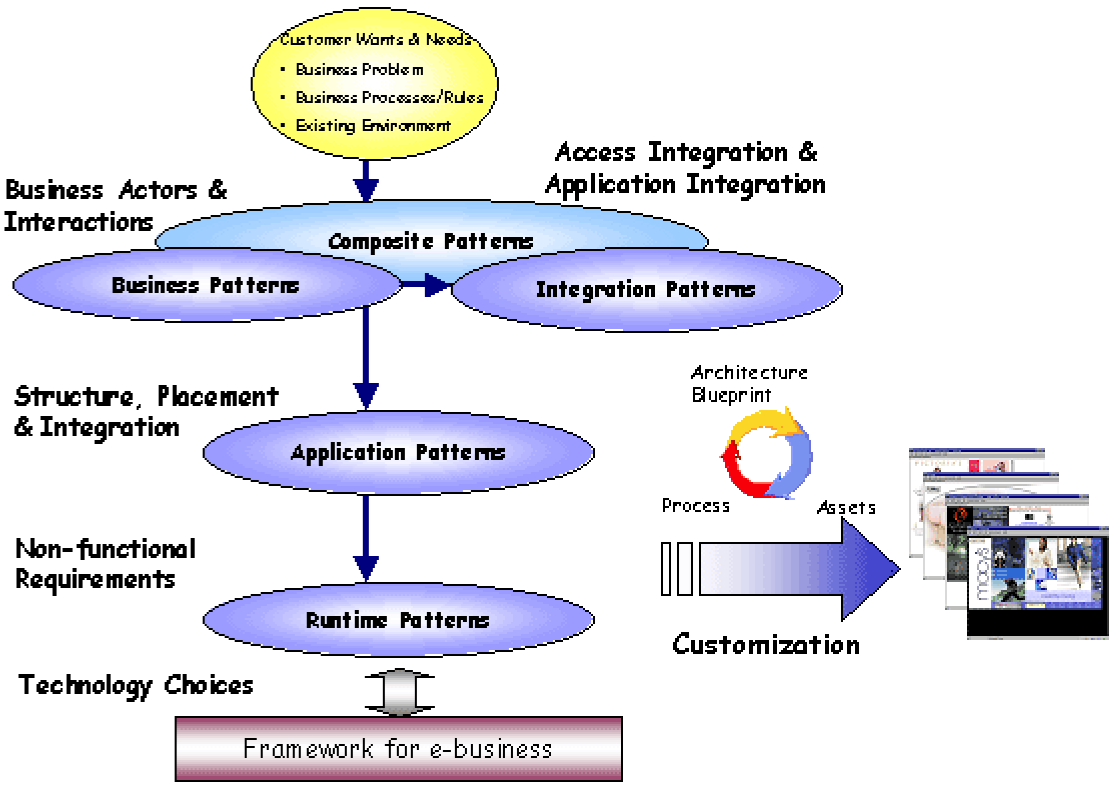


Figure 2: Pattern-driven Solution Engineering

Each of the business and integration patterns can be implemented by one or more application patterns. These application patterns describe the structure of the application (in terms of components and their interactions), the placement of these components to establish how processing and data are split between the tiers of the solution, and the integration (loosely or tightly coupled) between the application and other back-end systems.

Each of the application patterns, in turn, can be implemented by runtime patterns. These runtime patterns define how non-functional requirements, especially availability, performance, scalability and security are demonstrated in the application. These runtime patterns are represented by a logical node diagram which defines the topology of the architecture and the placement of nodes within the architecture. Note that these runtime patterns are product/tool independent, and can be mapped to any software/hardware products that can provide the needed service/functionality. Depending on the established drivers for the business and IT, the runtime patterns will be mapped onto appropriate products and tools that are known to support these requirements.

The best practices for each of the application patterns are documented in terms of technology options available for each component, guidelines for application design, application development and system management, and the considerations for performance. The technology choices are based on IBM's Software Strategy for e-business which is based on open standards and technologies, and comprises proven practices and a portfolio of products for developing, deploying and managing e-business applications. These best practices are documented in Redbooks (http://www.redbooks.ibm.com/.)

To further aid the use of these patterns, a Pattern Development Kit (PDK) is available for implementing various Self-Service solution designs (http://www-106.ibm.com/developerworks/patterns/install/index-lite.html.) The PDK is a complete, self-configuring, end-to-end skeleton Web application, which can be used as the basis for learning about the pattern, or as the skeleton on which to build the intended solution.

Finally, note that the entire pattern-driven solution engineering approach can be integrated into any software driven methodology. Starting from business objectives, the patterns for e-business can drive the solution choices based on proven best practices, and these can further be customized to meet the unique requirements of the enterprise.

3 USING PATTERNS TO ENGINEER SOLUTIONS

There are two major ways in which the patterns can be applied to help enterprises engineer solutions:

- Engineer a solution based on a single business pattern.
- Engineer a solution through a custom design that uses multiple business and integration patterns.

The first approach follows the pattern-driven approach in a top down fashion. Figure 3 shows a sample of using this approach applied to the self-service business pattern. The business requirements were translated into the business actors and interactions that follow the self service business pattern. The Self-Service business pattern, also known as the User-to-Business or U2B pattern, captures the essence of direct interactions between interested parties and a business. Interested parties include customers, business partners, stakeholders, employees, and all other individuals with whom the business intends to interact. As an example for the e-government transformation, self service would include submitting tax returns and renewing licenses. From this business pattern, the business and IT drivers indicated the choice of the router application pattern. The Router application pattern provides a structure for applications that require the intelligent routing of requests from multiple delivery channels to one of multiple back end applications. This application pattern in turn has an associated router implementation pattern. In the Router application pattern, the router tier serves as an integration point for delivery channels in the presentation tier, allowing access to individual back-end applications. In this Runtime pattern, the functions of the router tier are performed by an integration server. Associated with this router runtime pattern are several proven product mappings, the one shown is a coarse-grained, message-broker based design on the IBM AIX and Windows NT platforms. Along with the product mappings, there are established guidelines and related links to support designing, building, deploying and managing the solution (http://www-106.ibm.com/developerworks/patterns/u2b/at5-library-nt-mb.html.)

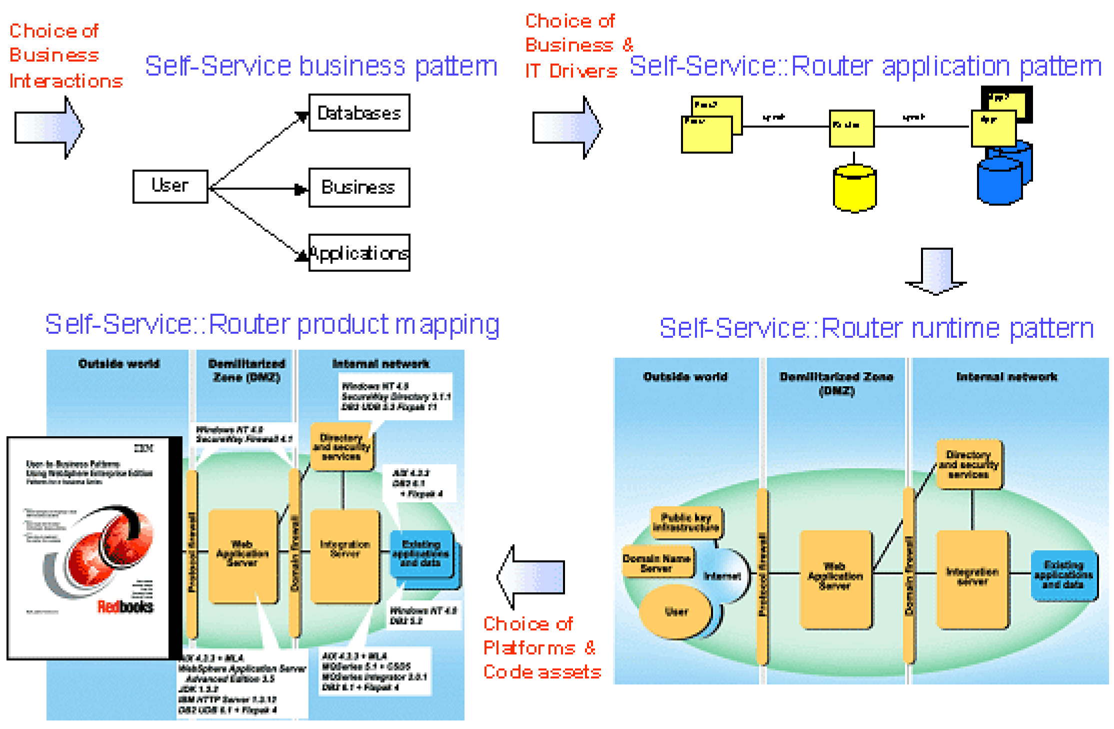

Choice of Business Interactions
Self-Service business pattern
User
Databases
Business
Applications
Choice of Business & IT Drivers
Self-Service::Router application pattern
Self-Service::Router product mapping
Self-Service::Router runtime pattern
Outside world
Demilitarized Zone (DMZ)
Internal network
Directory and security services
Web Application Server
Integration Server
Existing applications and data
Protocol firewall
Domain firewall
Public key infrastructure
Domain Name Server
Internet
User
Redbooks
Choice of Platforms & Code assets

Figure 3: Solution Engineering Based on Single Business Pattern

The second approach is more complex and described in the following scenario taken from the paper http://www-106.ibm.com/developerworks/patterns/guidelines/lord.pdf. “A team is building an online sales system. Thus, they begin with the Self-Service pattern to set up the initial interface that enables potential customers to take orders. Now, the team needs the system to integrate with existing applications to process orders. They then, using the Collaboration pattern, add capabilities to provide direct customer service support and communication within the enterprise. If the team wanted to allow customers to access data that would allow them to check on the status of an order or make inquiries directly to other data stores and applications, they would use the Information Aggregation pattern, quite possibly coupled with the Application Integration pattern. The team also wants to support supply chain management, so they use the Extended Enterprise pattern. Lastly, because they want to allow access from a variety of devices, the team turns to the Access Integration pattern to develop an end-to-end solution.” Figure 4 shows the composition of many patterns to engineer the solution.

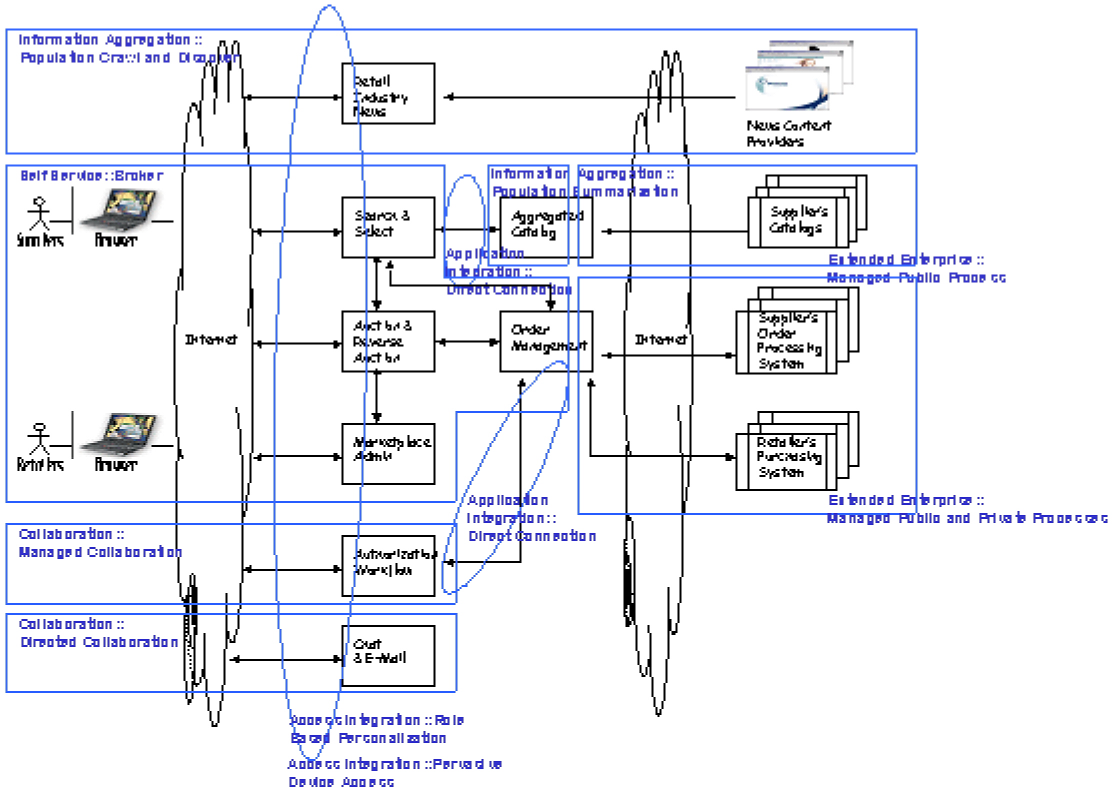
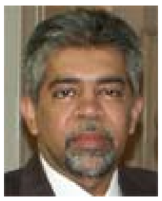

Information Aggregation :: Population Crawl and Discovery
Retail Industry News
News Content Providers
Self Service :: Broker
Suppliers
Browser
Information Aggregation :: Population Summarization
Search & Select
Aggregated Catalog
Suppliers Catalogs
Application Integration :: Direct Connection
Extended Enterprise :: Managed Public Process
Internet
Auction & Reverse Auction
Order Management
Internet
Suppliers Order Processing System
Retailers
Browser
Marketplace Admin
Retailers Purchasing System
Application Integration :: Direct Connection
Extended Enterprise :: Managed Public and Private Processes
Collaboration :: Managed Collaboration
Authorization Workflow
Collaboration :: Directed Collaboration
Chat & E-Mail
Access Integration :: Role Based Personalization
Access Integration :: Pervasive Device Access

Figure 4: Custom Design of a Pattern-Driven Solution

In summary, complex e-business solutions require appropriate proven industrial strength methodologies that can handle the complexity, and are based on reusable architectures, flexible designs, and best practice implementations. IBM Patterns for e-business provide a proven, pattern-driven approach for engineering such complex solutions.

About the author



Mahesh Dodani is an e-business architect with IBM Software Group. His primary interests are in enabling individuals and organizations to tackle complex e-business industry solutions. He can be reached at dodani@us.ibm.com.