JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2006**

Vol. 5, No. 2, March-April 2006

A Picture is Worth a 1000 Words?

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

1 SOA PICTURES

"*Today, if N architects are asked to draw an SOA, they will produce N different pictures, all labeled 'SOA'.*"

Sean McGrath, http://www.itworld.com/nl/ebiz_ent/03012005/

In a previous article (http://www.jot.fm/issues/issue_2005_07/column6) I presented the effort on addressing depth in the practice of SOA through the details of reference architectures, roadmaps, and governance. I focused on SOA reference architectures which define the key enabling technologies (services, web services, open standards, etc.), the realization of key SOA principles (composability, loose coupling, reusability, etc.), the details of key architectural layers and components (service provider/consumer architectures, enterprise service bus, component architecture, etc.), and the realization of these components through hardware, software and services.

Within IBM we are focusing on two main perspectives to articulate SOA pictures:

- A layered service perspective that shows the composite services that align with business processes, as shown in Figure 1 below. The layers show how enterprise-scale components (large-grained enterprise or business line components) realize the needed services and are responsible for providing their functionality and maintaining their quality of service using the operational systems. These exposed services are choreographed into composite applications to support business processes. Consumers access these exposed business services through a variety of interaction channels. The vertical layers describe capabilites that are needed to support the services across all other horizontal layers. The integration capabilities supports the interaction between services. As a specific example of the integration capabilities, an Enterprise Service Bus (ESB) supports the routing, mediation, and translation of these services, components, and business processes. The services must be monitored and managed for quality of service and adherence to non-functional requirements. The data architecture supports the storing and manipulation of service meta-data and information. Finally, the service location,

Cite this column as follows: Mahesh Dodani: "A Picture is Worth a 1000 Words? ", in Journal of Object Technology, vol. 5, no. 2, March-April 2006, pp. 35-40 http://www.jot.fm/issues/issue_2006_02/column4

version, service-level agreements, and subscription level are handled by approprite governance models.

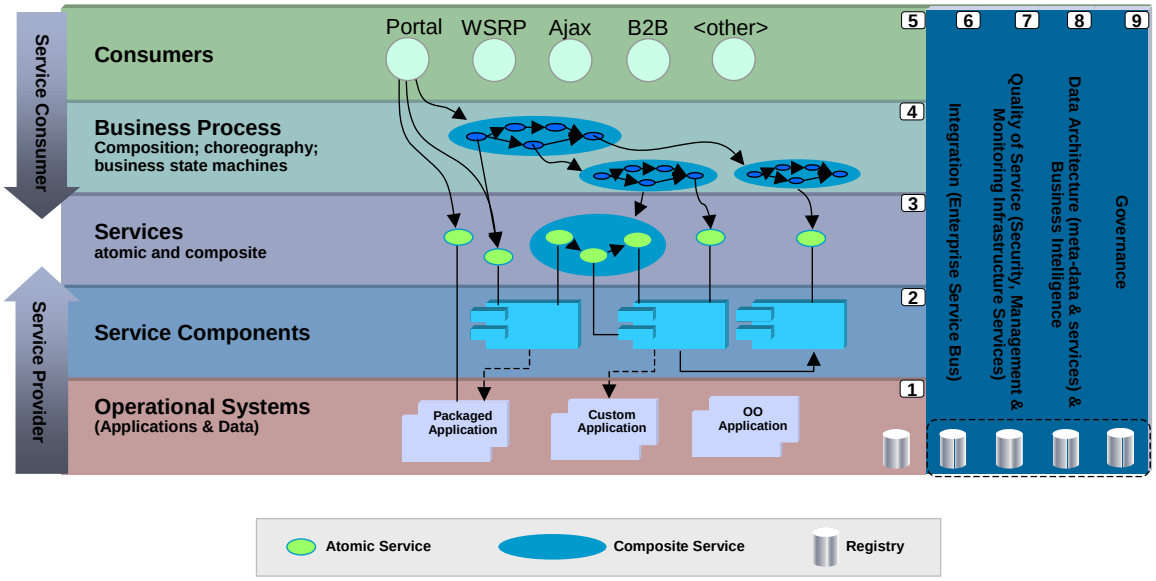

Figure 1: SOA Picture – Layered Services Perspective

- A capabilities perspective that shows the key functionalities that are required to implement comprehensive, enterprise wide SOA solutions, as shown in Figure 2. The Development Services are used to implement custom artifacts that leverage the infrastructure capabilities, and Business Innovation & Optimization Services are used to monitor and manage the runtime implementations at both the IT and business process levels. At the core of the SOA Reference Architecture is the Enterprise Service Bus which delivers all of the inter-connectivity capabilities required to leverage the services implemented across the entire architecture. Transport services, event services, and mediation services are all provided through the ESB. The SOA also contains a set of services that are oriented toward the integration of people, processes, and information through Interaction Services which provide the capabilities required to deliver IT functions and data to end users, meeting the end-user's specific usage preferences. Process Services provide the control services required to manage the flow and interactions of multiple services in ways that implement business processes. Information Services provide the capabilities required to federate, replicate, and transform data sources that may be implemented in a variety of ways. Many of the services in an SOA are provided through existing applications via the Access Services; others are provided in newly implemented components via Business Application Services; and others are provided through external connections to third party systems via the Partner Services. Underlying all these capabilities of the SOA is a set of

Infrastructure Services which are used to optimize throughput, availability and performance. IT Services Management Services include capabilities that relate to scale and performance, for example edge services, clustering services, and virtualization capabilities allow efficient use of computing resources based on load patterns.

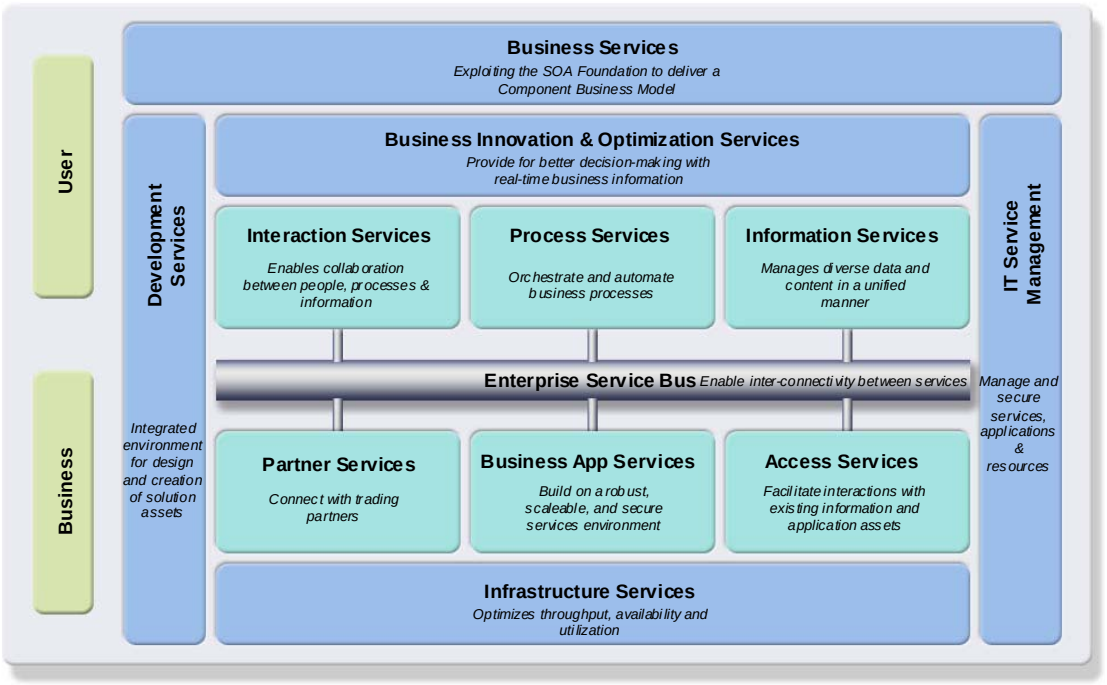

Figure 2: SOA Picture – Capabilities Perspective

The astute reader would realize that I used a around 1000 words to explain each of these pictures! Are these SOA pictures sufficient to explain how to architect, design and build appropriate solutions? Obviously not! In order to use these pictures, we need detailed descriptions, reference solutions, usage guidelines, mappings to technologies and products, implementation guidance, etc. So, what is missing from making these pictures meaningful and useful?

2 SOA MODELS – MAKING PICTURES MEANINGFUL

An emerging approach to making pictures meaningful is creating machine-readable models. A model is a description of a system from a particular perspective, omitting irrelevant detail so that the characteristics of interest are seen more clearly. Models help us to more easily understand complex technical environments such as a business domain or the architecture of a proposed solution. They enhance communication among stakeholders with different technical backgrounds and among disparate teams within an organization or across multiple organizations. Model-driven development is a style of

software development where the primary software artifacts are models from which code and other artifacts are generated. Models are either derived from other models through well defined description languages or are formally transformed between layers of abstractions (e.g. from analysis models to design models to code.) This transformational approach to models is formally defined through a model-driven architecture (http://www.omg.org/mda/) by the OMG (Object Management Group), and is shown in Figure 3.

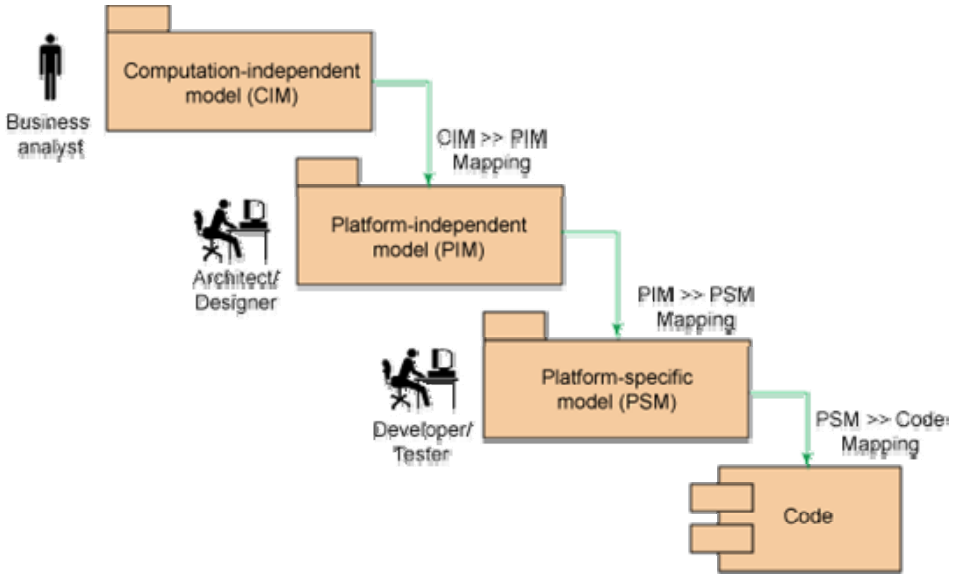

Figure 3: Model-Driven Development

As shown in the figure, the CIM does not show any system details and is often called the domain model because it is defined by business requirements. The CIM plays an important role in bridging the gap between requirements experts and design experts. The requirements experts focus on the business domain, whereas the design experts must connect new technology with artifacts to meet domain requirements that result in a PIM. Once the PIM is defined, the PSM (including the analysis and implementation models) can be generated using a transformation process based on the chosen implementation platform.

Applying this model-driven approach to SOA leads us to the concept of business-driven development, which describes a a methodology for developing IT solutions that directly satisfy business requirements and needs, as shown in Figure 4.

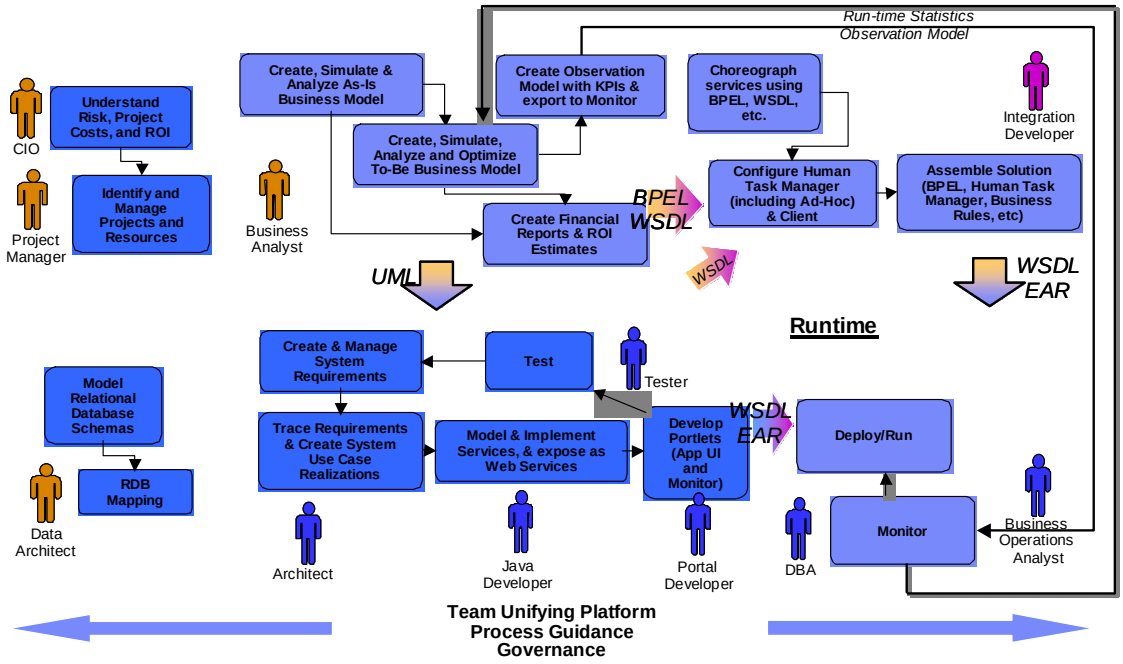

Figure 4: Business Driven Development

As shown in the Figure, a good architecture begins with a solid understanding of the business requirements; which in turn require a solid understanding of how the business works today, and how it should work in the future. Models (and the associated tools) document the business requirements (and goals and objectives) and create an "as-is" model of the business process. From these models, we can identify deficiencies and pitfalls and create a "to-be" model for how the business can be improved. These models can then be used to simulate the to-be process to validate cost savings, ROI, and other general improvement parameters. Once optimized and validated, the architect exports the to-be model via UML for further manipulation. The architect uses appropriate tools to transform the business requirements into software system requirements and models. This approach will ensure that the system implementation is driven by business requirements and is fully aligned with the business process model.

The architect imports business processes and refines application design, based on best practices and existing assets. An operational model will help the operations team plan the future deployment. The developer implements the application by leveraging best practices including highly productive J2EE capabilities. Built-in code analysis & unit testing capabilities enable developers to fix functional, performance, and security problems at the component level – early in the development cycle. Identifying and fixing problems at the component level greatly reduces the overall cost of delivering high quality applications. The application is validated to ensure that it functions as designed with acceptable performance. The functional and manual testing tools accelerate quality assurance activities as they build a valuable foundation of reusable test artifacts. The application models are assembled and deployed onto the runtime environment. The

Operations Manager monitors application performance and is automatically notified of problems, enabling fast triage to the right stakeholders (application, DB, network, etc.). Appropriate tools provide a centralized view into the network, systems, middleware, and application performance.

3 CONCLUSION

In conclusion, we need more than just pictures to facilitate consistent, effective and efficient use of architectures, designs, and best practices. Model-driven approaches allow us to give these technical pictures meaning and formally transition and transform between them. It facilitates communication between the various people involved with realizing a system – from business analysts, through architects and designers, to developers and testers, to implementers and operations managers, and to both business and technical monitors.

About the author

Mahesh Dodani is a software architect at IBM. His primary interests are in enabling communities of practitioners to design and build complex on demand business solutions. He can be reached at dodani@us.ibm.com.