Rational. software

SOA Development Using the IBM Rational Software Development Platform: A Practical Guide

Alan W. Brown
Simon K. Johnston
Grant Larsen
Jim Palistrant

Contents

SOA Development from a Business Perspective

The Need for Business Flexibility

As in years past, today's business leaders strive to drive revenue growth, contain costs, and improve the effectiveness of people and processes. What sets the current business climate apart from the past is rapid, continuous change. Increasingly, organizations that can respond to change with agility and flexibility distinguish themselves from those saddled with complex, brittle processes and systems.

Flexible businesses readily adapt to changes in any form, including mergers and acquisitions; compliance and regulatory mandates; competition; evolving technology; and shifting opportunities in outsourcing and insourcing. Business leaders must continuously rethink, redefine, and adapt their business models, operations, and organizational structures. Those who cannot, or will not, are quickly left behind.

IT Flexibility Enables Business Flexibility

In an On Demand world, success depends on the ability to rapidly respond to new challenges and opportunities. An On Demand Business can effectively transform business models and processes to adapt to change. An On Demand Business aligns business and technology for maximum flexibility and responsiveness, and establishes an integrated, modern business. Organizations that close the gap between the business view of activities and processes with the technology used to realize these activities can evolve business models in step with IT solutions. You must ensure that those goals and requirements drive IT development projects in order to close the business-IT gap and to achieve business goals. Moreover, with a flexible technology infrastructure, IT becomes an enabler of responsiveness and adaptability rather than an obstacle.

From Monolithic to Composite Applications

Traditional enterprise applications were large, monolithic solutions targeted at addressing a specific business function. Design techniques, tools, and processes were optimized around developing these kinds of systems. However, today's business systems increasingly involve *composite applications*. These n-tiered applications are collections of integrated capabilities and use information and logic from multiple sources. To pull all this together, a composite application spawns multiple transactions and sub-transactions across various runtime platforms and systems.

For example, as Figure 1 shows, a composite application may serve an end user who is interacting with a complex system that spans Web servers, J2EE application servers, integration middleware, and traditional systems. This kind of architecture is typical of many enterprise systems today.

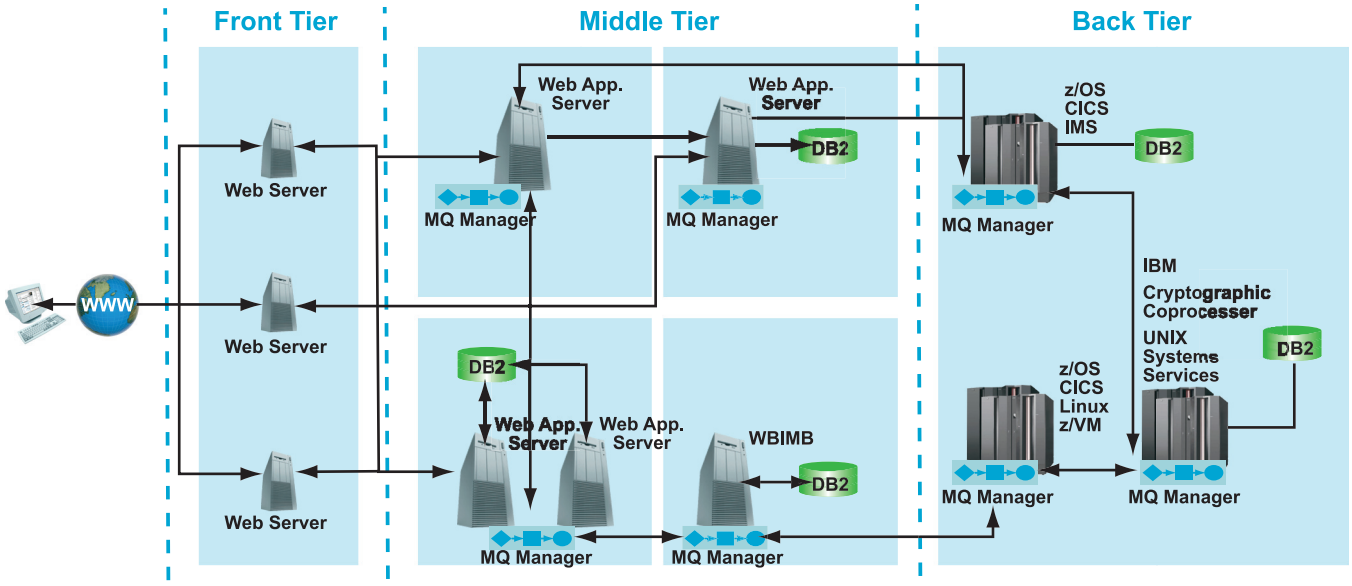

Figure 1. A composite application can span Web servers, J2EE application servers, integration middleware, and legacy systems.

In such a complex environment, it becomes increasingly difficult to build, run, and manage applications. The current data from industry analysts suggest that software development tools and techniques are not meeting the many challenges faced in such an environment and that organizations clearly need new approaches to developing and managing composite applications.

A New View of Design and Construction

The drive for flexibility, alignment, and efficiency is changing the way organizations view next-generation IT solutions. Three drivers characterize this new perspective:

- *Viewing an IT infrastructure's capabilities as a set of services that are assembled to meet specific business needs. This provides better insight into how business processes can be realized in IT solutions.*
- *Assembling and reassembling components into composite applications as business and market conditions demand.*
- *Focusing on asset management and reuse to obtain greater business efficiency.*

Service-Oriented Architecture: A Blueprint for Change

The need to respond to changing business demands with flexible IT solutions has led many businesses to Service-Oriented Architectures (SOAs). SOA is an IT framework that combines individual business functions and processes, called *services*, to implement sophisticated business applications and processes. SOA is an approach to IT that considers business processes as reusable components or services that are independent of applications and the computing platforms on which they run. It allows you to design solutions as assemblies of services in which the assembly description is a managed, first-class aspect of the solution, and hence, amenable to analysis, change, and evolution. You can then view a solution as a choreographed set of service interactions. The idea of viewing enterprise solutions as federations of services connected via well-specified contracts that define service interfaces is gaining widespread support. The ultimate goal of adopting an SOA is to achieve flexibility for the business and within IT.

Several important technologies and standards have been defined to support an SOA approach, most notably when the services are distributed across multiple machines and connected over the Internet or an intranet. SOAs that incorporate Web services rely on intra-service communication protocols, such as the Simple Object Access Protocol (SOAP); express the Web service interfaces in Web Services Definition Language (WSDL); register, search for, and reuse services and related assets in Reusable Asset Specification (RAS) repositories, directories, and Universal Description, Discovery, and Integration (UDDI) repositories; and share information in documents defined in the eXtensible Markup Language (XML).

While you can create an SOA that does not use Web services technology and you can use Web services technology in a way that would not be considered service-oriented, today Web services are the most common way to implement an SOA.

Of course, SOA is more than a set of standards and service descriptions, in the same way that object-oriented architectures are more than a set of class hierarchies. SOAs provide the flexibility to treat business process elements and the underlying IT infrastructure as secure, standardized components—typically, Web services—that you can reuse and combine to address changing business priorities. SOAs enable businesses to address key challenges and pursue significant opportunities quickly and efficiently. They also:

- *Make it easier to link legacy systems to new applications. Each successive wave of technology—from mainframes to client/server databases, Enterprise Resource Planning (ERP), Web applications, Java applications, and now services—has contributed to a mosaic of applications that most enterprises will likely feature for years to come. SOAs simplify the integration of a new application with one or more existing "legacy" applications.*

- *Facilitate the improvement of internal business processes. Analyzing, improving, automating, or creating new business processes is fundamental to cost-cutting, enhancing customer satisfaction, and virtually every business initiative. The ability to rapidly assemble or reassemble solutions using SOAs streamlines these efforts.*
- *Enhance shared business processes. SOAs are based on open standards and are platform-independent, making it easier for businesses to automate common B2B processes and improve the efficiency of supply chains.*

Transitioning to an SOA

Transitioning to an SOA is a journey that frequently requires business leaders and IT managers to shift their perspective. In particular, they need to place added emphasis on not only a new, service-oriented approach to developing applications, but also business-driven development and asset lifecycle management.

Service-Oriented Design and Development

Businesses that create solutions for SOA must rethink the kinds of systems they are building, reassess the skills in their organization, and redefine the ways in which team members collaborate. More importantly, adopting a services-oriented approach to solutions development necessitates a broader review of its impact on how you design solutions; what it means to assemble them from disparate services; and how you manage and evolve deployed services-oriented applications. IBM® refers to this broader context of service-orientation as Service-oriented design and development.

Service-oriented design and development incorporates a broad range of capabilities, technologies, tools, and skill sets, including:

- *Managing the services lifecycle—including finding, applying, evolving, and maintaining services.*
- *Establishing a platform and programming model, which includes connecting, deploying, and managing services within a specific runtime platform.*
- *Adopting practices and tools that enable teams to effectively create and assemble services in the context of solving changing business needs. This includes mining existing applications to discover potential services, wrapping existing functionality to make those capabilities accessible as services, creating new services, and "wiring" together services by connecting behavior exposed through their interfaces. Fundamental to these capabilities is the availability of clear guidance and best practices for architecting services-oriented solutions in repeatable, predictable ways.*

Business-Driven Development

In what IBM calls *Business-Driven Development for SOA*, business goals and requirements drive downstream design, development, and testing to transform business processes into composite applications that automate and integrate the business. You trace requirements across the entire lifecycle from business goals, through software designs and code assets, to composite applications.

You design solutions for integration to ensure that they are flexible and can be adapted as business needs change. You maximize asset reuse and minimize redundancy. Finally, you build in quality from the beginning.

Business-driven development closes the gap between business and IT and ensures that you build solutions based on business needs. At the same time, the delivered solutions are much more flexible, enabling your business to rapidly understand the impact of changing business needs and to respond to them appropriately.

Asset Lifecycle Management

When organizations think of services as key assets in system design, the value of reusing these services becomes more apparent. As a result, technologies and techniques for asset management and governance, and repeatable ways to capture patterns for combining assets, become much more important. In an asset-based development approach, these assets hold critical value to the organization and must be carefully managed and administrated. The team infrastructure for managing assets consistently across projects and across the enterprise takes on a key role in this approach.

Like any asset, the lifecycle of a service includes multiple phases in which you identify and discover the service; harvest and create it; certify and publish it; reuse and measure it; and ultimately retire it. A set of workflows, referred to as *Asset Production*, *Asset Identification*, *Asset Management*, *and Asset Consumption* (see Figure 2), describe this lifecycle.

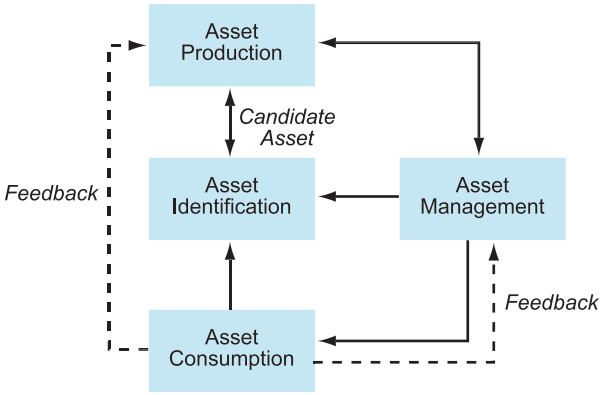

Figure 2. Workflows in service lifecycle management

You may consume or reuse services at various points in the lifecycle, including during development and at run time. To effectively reuse a service during development, you must package the services consistently with artifacts and meta data describing the service. The Reusable Asset Specification (RAS), an Object Management Group (OMG) standard, defines the meta-data format used to package the service. RAS is expressed in XML, and each asset has a RAS XML manifest that contains the meta data.

This service meta data includes classification information and versioning information, as well as information that describes the service artifacts and the relationship among those artifacts. For developers, RAS meta data describes how to properly use the service and the context within which it should be used.

IBM and SOA Development

SOA enables businesses to improve flexibility within IT and better align IT with business objectives. Of course, *building* SOA solutions is only part of the story. IBM provides the IBM SOA Foundation (see Figure 3) for SOA support across the lifecycle, including modeling, assembling, deploying, and managing SOA solutions. IBM also offers education, consulting services, and years of SOA expertise to help its customers build SOA-based solutions.

IBM SOA Foundation

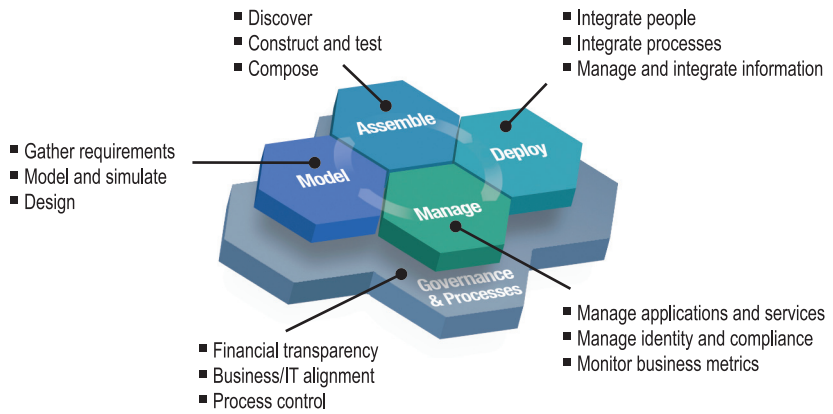

Figure 3. The IBM SOA Foundation is a modular, integrated, and open set of software, best practices, and patterns that support the complete SOA lifecycle.

The IBM SOA Foundation focuses on aligning IT with business priorities and enabling business flexibility: you can model business processes; design and construct services and assemble them into composite applications, which you then deploy, manage and adapt as needed to satisfy changing business requirements.

The remainder of this paper will focus on the *build* aspect of the IBM SOA Foundation in which you model, construct, test, and assemble solutions.

Business-Driven Development for SOA

When you use business requirements to drive all downstream development activities, you close the gap between IT and business. In the context of service-oriented design and development, the result is exceptionally flexible solutions that you can rapidly adapt to meet changing business needs and priorities. Business-Driven Development for SOA (Figure 4) addresses two of the key challenges facing businesses leaders—improving responsiveness and aligning IT with business goals.

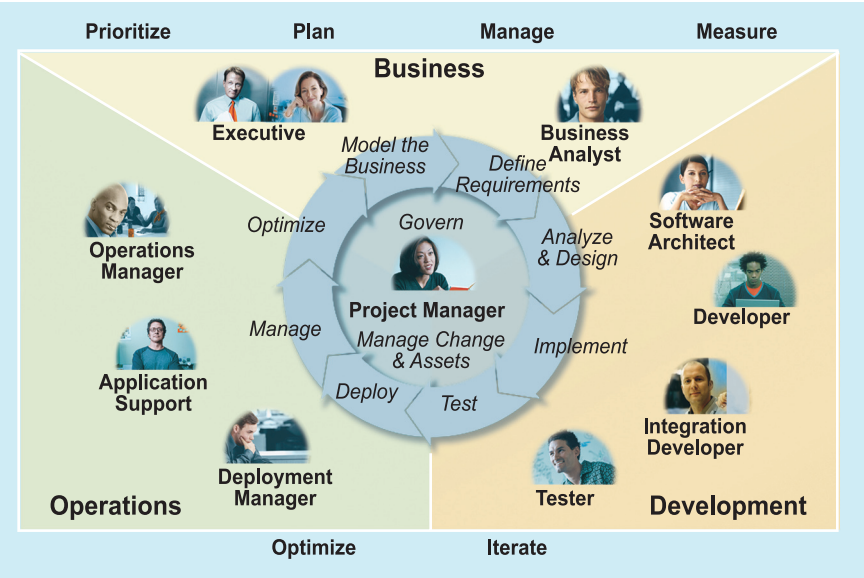

Figure 4. Business-Driven Development for SOA transforms business processes into composite applications that automate and integrate the business.

The IBM Rational Software Development Platform for SOA

The IBM Rational® Software Development Platform provides development teams with a complete, open, modular, and proven environment for business-driven development of flexible SOA solutions. It includes:

- *Standards-based and emerging programming models that automate many aspects of business process modeling, as well as services design, construction, and composition, to reduce project time and costs*
- *Industry-leading process and portfolio management and software configuration management solutions that guide practitioners through each phase of the SOA development lifecycle, and maximize asset reuse, project success, and predictability.*
- *Role-based tools that provide both business and technical users with the precise set of capabilities they each need, plus tight integration powered by Eclipse that closes the gap between business and IT by facilitating collaboration, ensuring that business requirements drive downstream development, and providing requirements traceability from inception to deployment.*

The IBM Rational Software Development Platform is a powerful foundation for Business-Driven Development for SOA. It also supports code-centric, visual programming, and model-driven styles of development.

In addition to providing a broad array of tools for every development lifecycle phase, the IBM Rational Software Development Platform delivers real value through the deep integration of tools and run-time capabilities across all aspects of the business in support of a services-oriented view of solutions.

Tight integration between tools in the IBM Rational Software Development Platform is based, in part, on the Eclipse platform. Eclipse technology enables

integration, flexibility, and extensibility in a single, coherent platform for software development. It serves as a foundation for the broader ecosystem that makes the IBM Rational Software Development Platform viable to many users.

As Figure 5 shows, the IBM Rational Software Development Platform provides role-specific tools for building SOA applications, including requirements and analysis, design, construction, software quality, process and portfolio management, and asset lifecycle management. The following overview provides a brief look at each of these capabilities, how they fit into an SOA approach, and the IBM Rational Software Development Platform tools that support them. This overview will set the stage for a more in-depth exploration of how these tools work together in the development lifecycle of an example SOA project.

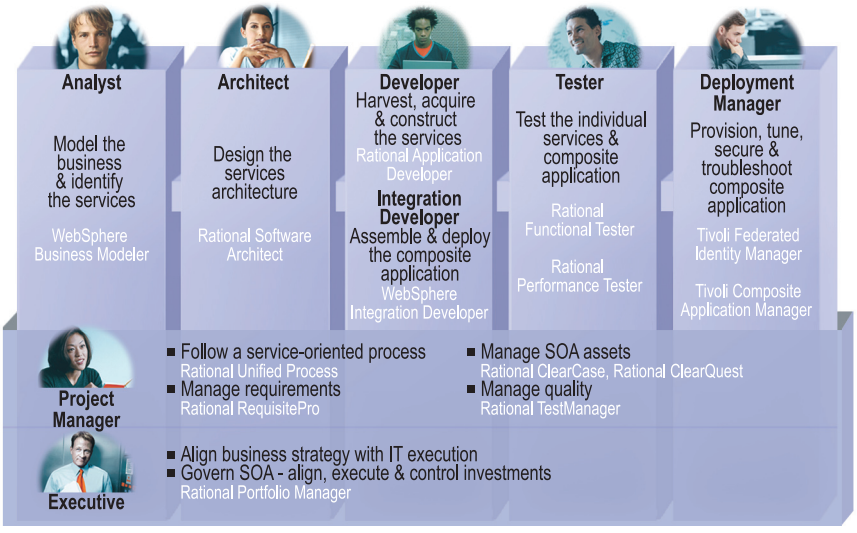

Figure 5. The IBM Rational Software Development Platform for SOA

The IBM SOA Programming Model

The development of services-oriented solutions requires teams to define, manage, and evolve a collection of different elements. These elements describe how the persistent data you're manipulating expresses the business algorithms you will execute and supports meaningful interactions with system users. Taken altogether, this collection of elements is typically referred to as the programming model.

A key aspect of the IBM Rational Software Development Platform is the use of a programming model influenced strongly by an SOA that is being implemented in the WebSphere® platform, the IBM middleware stack (DB2®, Tivoli®, and Lotus®), and in particular, within the IBM Rational Software Development Platform.

The IBM programming model for SOA separates development and deployment activities into separate phases that can occur at different times and that different individuals with different skills can perform. This yields a true separation of concerns, enabling you to repurpose software components.

Key elements of the IBM SOA programming model include:

- *Service Data Objects (SDOs) (now in the standardization path at the Java Community Process); they provide a simplified data access programming model for various resources and complement the core Web services standards XML, WSDL, and SOAP*
- *Business Process Execution Language for Web Services (BPEL*), a service orchestration and component scripting standard that supports workflow and business process integration*
- *JavaServer Faces (JSF), a Java framework that speeds Web application development for developers who are not expert J2EE developers*

* *BPEL is currently being standardized as WS-BPEL*

- *Customization of applications using external policies and rules, including a series of emerging standards that are in development for policy definition and enforcement, including Web Services Policy and OMG Business Semantics of Business Rules (BSBR)*

A simplified model for describing the overall service wiring and assembly aspects of an SOA solution is critical to creating an SOA solution. IBM has also introduced the Service Component Architecture (SCA), a simplified abstraction of services that encompasses various service implementation technologies. SCA complements SDO, BPEL, business state machines, and business rules support to simplify and accelerate service design, development, assembly, and deployment.

Business Modeling and Requirements

Effective business-driven development starts with analyzing business requirements, needs, and existing processes. Business analysts then use this analysis to drive requirements and all subsequent stages of development.

IBM WebSphere Business Modeler enables analysts to model, simulate, and analyze complex business processes quickly and effectively. Analysts use WebSphere Business Modeler to model "as-is" and "to-be" business processes, allocate resources, and perform "what-if" simulations to optimize and estimate business benefits. Later, they can transform these models into Unified Modeling Language (UML) and Business Process Execution Language (BPEL) to jump-start design and integration activities.

Once the analyst has identified a solution by selecting and optimizing the best "to-be" process, they define business requirements and refine them into software requirements and use cases in **IBM Rational RequisitePro**. Requirements and requirements management are central to any mature software development process, and business modeling helps ensure that

software requirements and use cases reflect real business needs. Well-written use cases provide the foundation for architecture, analysis, user interface (UI) design, and testing. Requirements in Rational RequisitePro can be linked to process models, as well as test cases and test plans, providing full traceability and ensuring that business requirements drive downstream development. The result is a system that meets business goals and requirements and delivers real business value. Within the IBM Rational Software Development Platform, requirements are integrated with defect tracking, design, development, and testing tools to jump-start activities, minimize rework, and provide each team member with a direct window into end user needs.

Service-Oriented Modeling and Design

An effective approach to designing a service-oriented application requires you to model and define well-documented interfaces for all major service components prior to constructing the services themselves.

IBM Rational Software Architect is an integrated design and construction tool for creating service-oriented applications. By leveraging model-driven development with the UML and unifying all aspects of software application architecture, Rational Software Architect enables software architects to design flexible services architectures and automatically apply design patterns for SOA from analysis and design to implementation.

Integration between IBM WebSphere Business Modeler and IBM Rational Software Architect enables teams to transform business models to software models that will automate business activities. The integration automatically maps business notation to software notation (UML), relieving designers from that conversion burden and ensuring that the software model does not inadvertently impose changes on the business process.

To assist with solution design in a services-oriented world, you can customize IBM Rational Software Architect with domain-specific notations and languages relevant to defining, assembling, and managing services. General mechanisms are available to create customized notations using built-in product extension mechanisms. Examples of domain-specific languages created using these mechanisms include:

- ***The UML Profile for Software Services*** *accelerates the transformation of business processes into Web services by providing a common language for describing services, and helping software architects to model, map, and partition services.*
- ***The UML Profile for Business Modeling*** *presents a language for capturing business models in UML and is supported by the Business Modeling discipline in IBM Rational Unified Process®.*

Service-Oriented Construction

Java, J2EE, and Web developers are often tasked with creating the core elements supporting SOA, typically Web services. Manually creating Web services—or even developing systems that consume existing Web services—requires a substantial amount of tedious and error-prone work.

IBM Rational Application Developer is a comprehensive integrated service-oriented development environment that automates many of the tasks commonly performed in the construction and consumption of Web services. Developers can focus on writing the business logic code while relying on Rational Application Developer to automate everything from the WSDL file and code generation to test-client generation and WS-I conformance verification. Developers use Rational Application Developer to create, build, consume, test, deploy, and publish Web services—either from scratch or by enabling existing applications for Web Service Interoperability (WS-I) compliance.

Rational Application Developer greatly simplifies the process of creating a Web service top-down based on requirements specifications and UML models. Developers save time and reduce errors by using Rational Application Developer to automatically generate WSDL files that contain an XML schema to describe Web services, as well as skeleton JavaBeans or Enterprise JavaBean (EJB) files. Rational Application Developer also includes an XML schema definition (XSD) editor for specifying message format.

When working bottom-up from an existing set of Java classes or EJBs, Rational Application Developer can also automate much of the process of turning those components into a Web service. An easy to use Wizard will generate WSDL interfaces to methods in the Java Class, a SOAP deployment descriptor, and a Web page to be used as a test client to test interactions with the Web service.

Rational Application Developer also has J2EE Connector Architecture tools that let the developer easily integrate existing Enterprise Information Systems (EIS) such as CICS or IMS into their SOA solution. Using these tools, developers can quickly create a Web service from an existing EIS transaction.

Support for Enterprise Generation Language (EGL) in Rational Application Developer enables business-oriented and procedural developers who may not know Java to develop, test, and debug data-driven Web applications, Web services, and business logic using procedural programming constructs instead of Java. EGL's comprehensive Web services support handles most of the hard work in developing or consuming Web services. Minimal coding is required to consume existing Web services or to enable existing assets; this leads to a shorter learning curve and high productivity for business-oriented developers.

To enable developers to build in quality early in the development cycle, Rational Application Developer includes:

- *Automated code review tools to validate coding best practices*
- *Component test tools to automate the creation of test stubs, harnesses, and input data based on WSDL file analysis*
- *Performance profiling tools information to detect and diagnose bottlenecks, deadlock, and race conditions*

Rational Application Developer also automates the process of discovering and consuming existing Web services. Developers use a built-in UDDI Explorer tool to browse UDDI registries, find Web services, and import them into a project.

For Web designers, Rational Application Developer will auto-generate everything needed to interact with the Web service from a Web page. It generates a JSP with JavaServer Faces UI controls for the Web service inputs and outputs and it generates the Java code needed to invoke the Web service.

Service-Oriented Composition and Assembly

Integration developers create and deploy composite applications by refining business process models—created in WebSphere Business Modeler—into an executable set of choreographed services. Integration developers use **IBM WebSphere Integration Developer** to assemble composite applications that deploy to the IBM WebSphere Process Server. Using the visual BPEL editor, integration developers can view the business processes that business analysts designed. They then use WebSphere Integration Developer to choreograph the services, wire them together into a composite application, optionally test them with a built-in test environment, and deploy directly to a run-time environment.

WebSphere Integration Developer provides integration developers with the visual software development tools they need to specify, test, and deploy

executable business processes that integrate Web services, enterprise applications, human tasks, and other service components into effective SOA-based business solutions.

Process and Portfolio Management

In addition to a comprehensive set of tools for services-oriented development, the IBM Rational Software Development Platform also incorporates practical process guidance. **IBM Rational Unified Process, or RUP®**, is a configurable software development process platform that delivers proven best practices and a configurable architecture.

RUP is well-suited to the needs of SOA development initiatives because it is founded upon software engineering best practices, offers a configurable process framework, and is scaleable to support enterprise initiatives.

Team members can apply RUP to support the development of a complete SOA composite application as well as each individual service within the solution. RUP also provides a systematic approach for bridging the gap between business and IT.

IBM also offers RUP plug-ins—downloadable assets containing various process components customized for specific tools, technologies, or domains—including plug-ins tailored to SOA development.

- ***The RUP Plug-In for SOA*** *integrates support for service-oriented architecture and service-oriented solutions into the RUP framework with SOA-specific concepts, guidelines, activities, artifacts, and tool mentors.*
- *The* ***RUP Plug-In for WebSphere Business Modeler*** *updates the Business Modeling discipline in RUP to leverage WebSphere Business Integration solutions and provide a unified approach for business modeling based on the essential capabilities of the IBM Rational Software Development Platform.*

From an SOA perspective, applications are dynamic entities—combinations of services to meet a particular set of business requirements. One effect of taking this view is a need for enterprises to explicitly design, implement, and manage an organization's set of services as a portfolio of available capabilities. Executives and project managers use **IBM Rational Portfolio Manager** to gain insight into the business benefits, costs, and risks of the SOA services portfolio. With Rational Portfolio Manager, they can prioritize proposed, existing, and under-construction services; track service-level financials; manage SOA project-team dependencies; forecast demand for service creation; and better understand the cost of SOA creation, operations, and maintenance.

Software Quality

In addition to the developer testing capabilities in Rational Application Developer and WebSphere Integration Developer, the IBM Rational Software Development Platform includes tools to help testers continuously assess the quality of Web services and ultimately composite applications. They can link test plans and individual test cases to project requirements in Rational RequisitePro to ensure complete requirements-based test coverage.

Effective quality assurance of SOA solutions depends on the management and execution of functional and performance tests.

Several commercial solutions exist to automate test scripts. The practicality of these solutions depends upon the coverage of user interface technologies that the tool supports. Business processes will likely involve several user interfaces, ranging from "green screen" applications and different types of Web applications (including ASP.NET, JSPs, JSF, and portlets) to native applications, such as mail clients and pervasive devices. Testing an end-to-end business process typically requires the automation engine to drive all these user interfaces.

IBM Rational Functional Tester provides testers with extensive technology coverage and an open test environment. Specifically, it offers scripting using non-proprietary languages: Java and VB.NET. In addition, they can easily record test scripts on a combination of different user interfaces, built with different technologies. This makes the orchestrated verification of an end-to-end business process virtually seamless. They can automatically trigger test execution from the command line, making it easy to integrate testing as part of an end-to-end build-deploy-test process.

While functional testing is central to any testing effort, you need to consider several testing dimensions beyond functionality. In the context of SOA applications, effective performance and security testing is vital. Chatty, over-communicative services and large message payloads can greatly impact an SOA application's performance profiles. Therefore, you need to deal with performance verification as early as design time, and QA teams must make it a prime consideration.

IBM Rational Performance Tester is a test creation, execution, and analysis tool that validates application scalability and reliability under multiple user loads. Rational Performance Tester enables teams to pinpoint system bottlenecks before application deployment. Because tests are created by recording a user's activity within a Web browser, teams need little to no programming knowledge to understand and modify these tests. Data pooling capability ensures unique data for each emulated user. Using an intuitive graphical test scheduler, teams can then organize their tests to accurately simulate the different types of users and user activities the application under test will support.

During test execution, while emulating the desired number of concurrent users, Rational Performance Tester generates reports that highlight poorly performing Web pages, URLs, and transactions. Teams can expose performance problems in even the most complex systems, increasing the opportunity for problem identification and repair before the system goes live.

Asset Lifecycle Management

Organizations are looking at their next-generation solutions platforms with a new focus on asset management and reuse. They're placing a new emphasis on creating, harvesting, applying, and managing assets across the lifecycle. For practical reasons, this focus is accompanied by strong governance practices for reusable assets tied to a flexible asset lifecycle management system.

As organizations create SOAs to leverage existing assets, the ability to accelerate and manage change becomes more critical. Using IBM Rational ClearCase® and IBM Rational ClearQuest® for software configuration management, teams can automate the software lifecycle and establish a consistent process across distributed environments.

IBM Rational software configuration management products help teams improve project collaboration and release coordination; increase development responsiveness and agility; and enhance operational efficiency.

In addition, you can package and consume reusable SOA-related development assets within the IBM Rational Software Development Platform through direct support for RAS (see Figure 6).

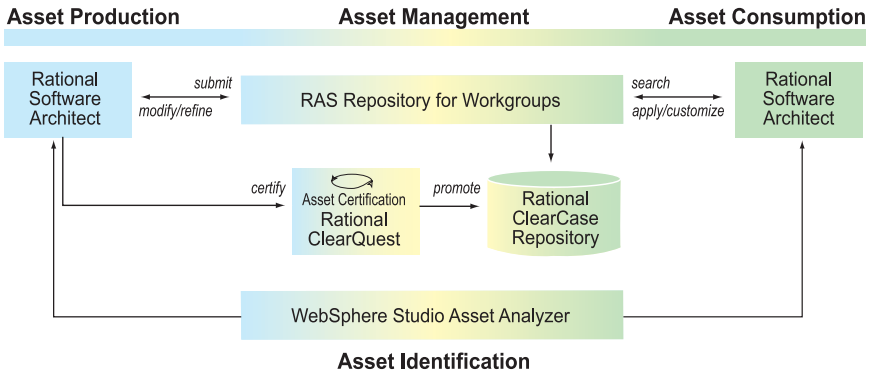

Figure 6. Rational ClearCase and Rational ClearQuest support RAS and asset lifecycle management.

The IBM Rational Software Development Platform includes several other tools that enable the team to more effectively manage asset workflow. WebSphere Studio Asset Analyzer provides support for the Asset Identification workflow. Rational Software Architect provides a RAS client for Asset Production. This tool also includes a RAS client, called the Asset Explorer, to facilitate Asset Consumption. Developers and architects use the Asset Explorer to connect to one or more RAS repositories and issue searches, which examine the RAS XML manifest in each RAS asset.

You can measure the assets and services in several ways, quantitative and qualitative. Teams can use the RAS Repository for Workgroups to track metrics, such as how many times users search an asset, how often they browse its documentation, and how many times they import and reuse it. Developers can also rate the asset or service and provide textual feedback. These kinds of measurements are critical for the health and livelihood of the asset repository.

An Example of Business-Driven Development for SOA

An effective way to understand the value of the IBM Rational Software Development Platform in Business-Driven Development for SOA is to explore how the tools help various roles work together throughout the SOA development lifecycle.

The following scenario traces one such effort. This paper leaves the specifics of the actual business needs open. In this scenario, a business may be automating particular aspects of a car rental reservation process, a work-order process, a supply chain, or any other business process—new or existing—that an organization may need to execute (see Figure 7).

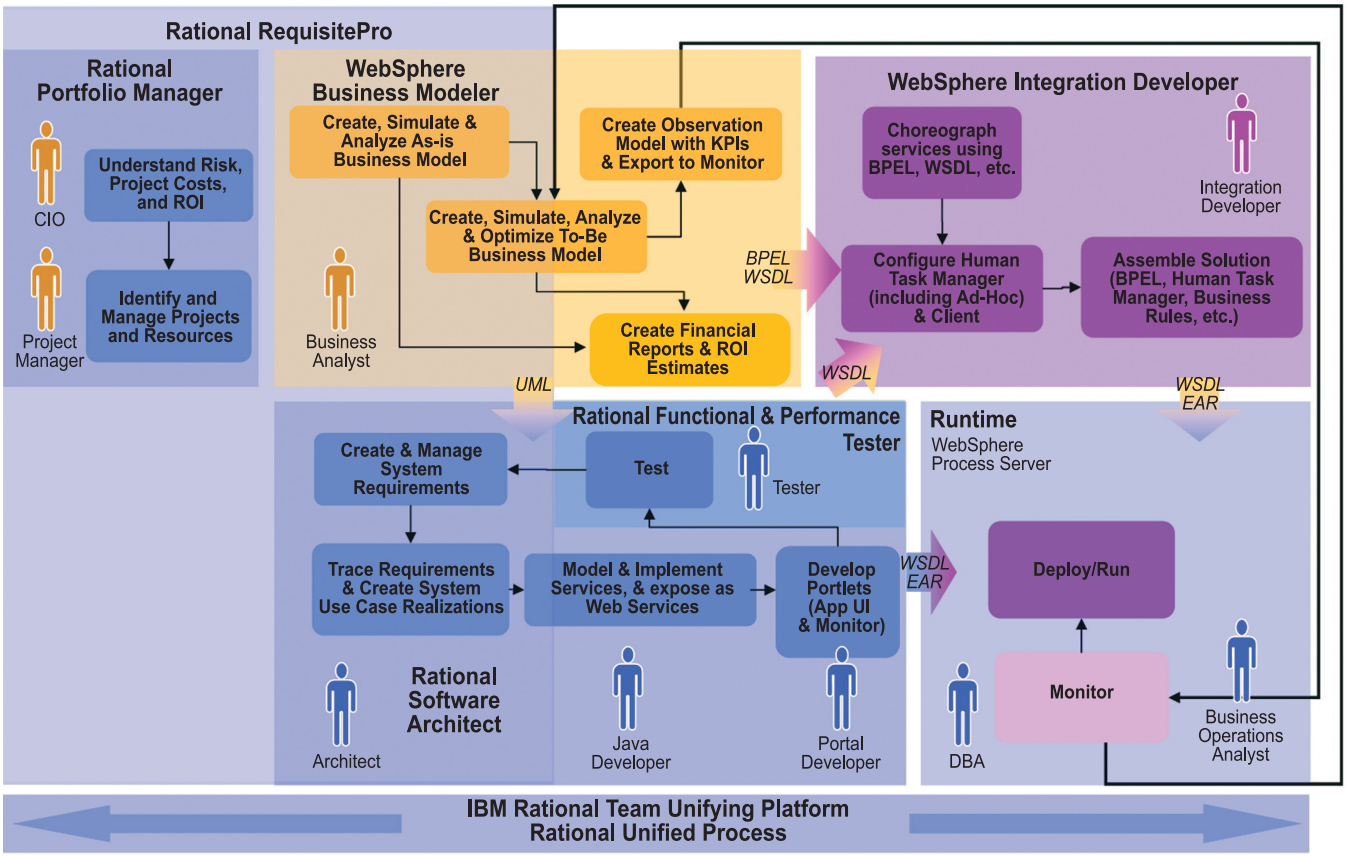

Figure 7. A suggested scenario for building a process integration solution using the Rational Software Development Platform

IT Executive and Project Manager: Aligning, Executing, and Controlling Investments

Governance underpins the success of SOA initiatives. IT executives and project managers use IBM Rational Portfolio Manager to assess business benefits, costs, and risks of the organization's SOA services portfolio.

With Rational Portfolio Manager, executives prioritize proposed, existing, and under-construction projects based on business priority, risk, and return, as Figure 8 shows. Additionally, the product allows them to track service-level financials, gain insight into SOA development, manage SOA project-team dependencies, and forecast demand for service creation and updates.

Once SOA initiatives are under way, Rational Portfolio Manager provides a complete project management, collaboration, and work management environment. Its central repository makes it easy for project managers to report status, teams to collaborate within and across projects, and decision makers to gain quick access to the progress and health of interconnected SOA projects and programs.

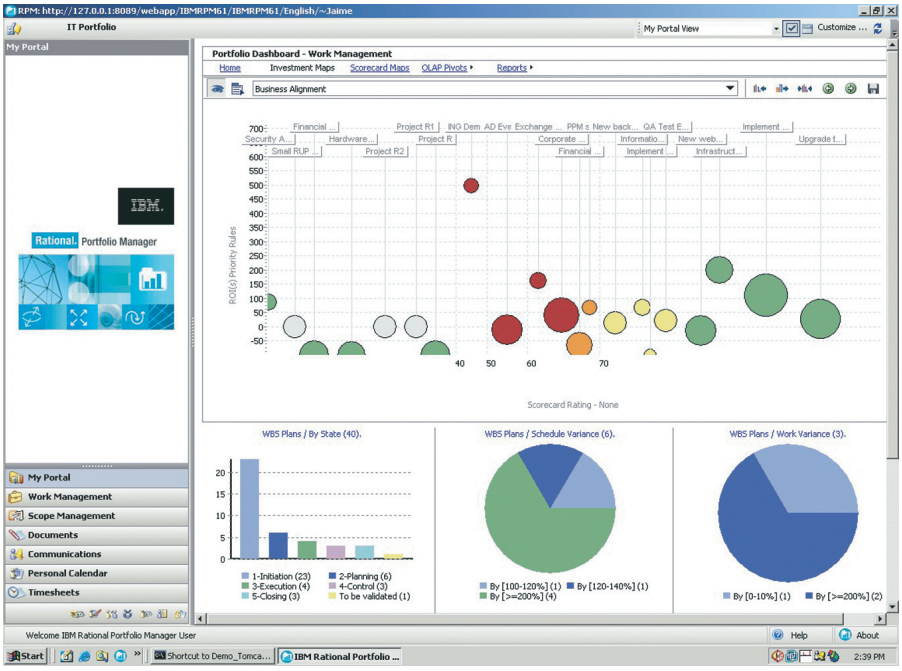

Figure 8. Rational Portfolio Manager can help assess the relative ROI, cost, and business alignment of proposals at a high level.

Business Analysts: Analyzing and Designing the Business Process

Business analysts begin by reviewing the business goals and objectives that drive the process enhancement request. Using IBM WebSphere Business Modeler, business analysts complete the "as-is" process model in an intuitive, easy-to-use notation. This allows you to understand the current system of automated and manual steps at a high level. The organization designs, simulates, and analyzes potential changes to the system for potential return-on-investment before it commits to any changes to the business process. This analysis results in the development of the "to-be" process model that an IT solution is intended, in part, to automate.

The business analyst then exports the business modeling project and artifacts as BPEL—as well as Web Services Definition Language and XML Schema or XSD artifacts—for subsequent consumption by integration developers (see Figure 9).

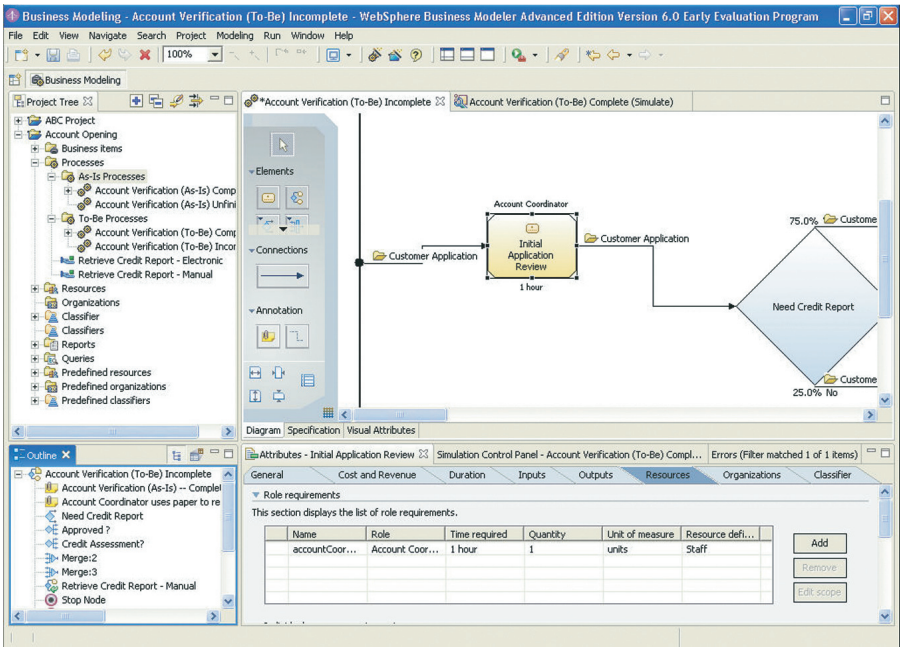

Figure 9. Business analysts develop business process models using WebSphere Business Modeler.

Working together with IT managers, business leaders, architects, and other stakeholders, the business analyst uses **IBM Rational RequisitePro** to identify solution requirements that will drive the development lifecycle.

Business analysts and architects define business requirements and refine them into software requirements and use cases by using Rational RequisitePro to author and share requirements; trace requirements and perform impact analysis; and receive early notification whenever requirements change.

Throughout development, Rational RequisitePro provides business analysts and the entire team with fast, intuitive access to a project's requirements from a centralized location, making it easy to stay informed and on track.

Software Architect: Designing the Business Service

Using IBM Rational Software Architect, a software architect imports the WebSphere Business Modeler project and, applying a model-driven development approach, automatically transforms the process model into a UML model. The software architect then uses Rational Software Architect to create detailed UML diagrams that identify the service elements that will be implemented. Using the Asset Explorer in Rational Software Architect, the architect may also search for and reuse existing RAS assets, including services, from across the enterprise. The RAS asset defines a rich set of data on the service, including the location of its UDDI registry.

The architect can develop a design model with traceability to the original business model. In addition, he or she can develop use case models in Rational Software Architect and link them to specific requirements in Rational RequisitePro; specifically, the architect may need to identify non-functional requirements associated with the process models imported from WebSphere

Business Modeler. Interfaces to any new services are modeled, showing input to and output expected from each service. (See Figure 10

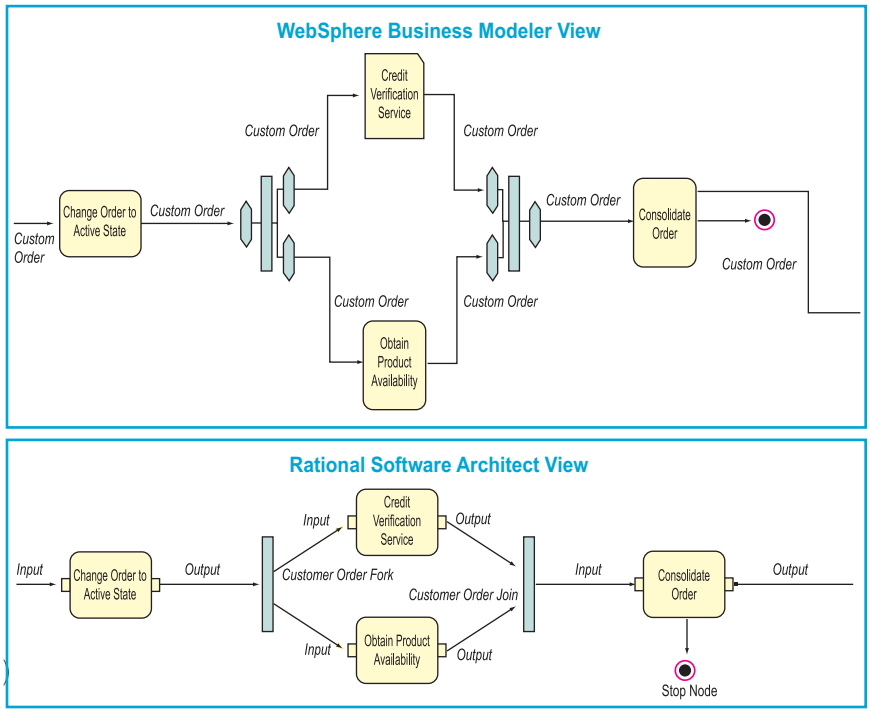

Figure 10. Business processes are represented in WebSphere Business Modeler using the business-preferred notation and in Rational Software Architect with the standard UML notation used by software

To jump-start development, an architect performs a UML-to-code transformation in Rational Software Architect in order to generate the skeleton of the service and the domain model, as Figure 11 shows. This transformation is based on the UML design model that implements the interface the architect provides. In this context, the interface represents a contract between the architect who designed the interface and the developer who will implement the interface.

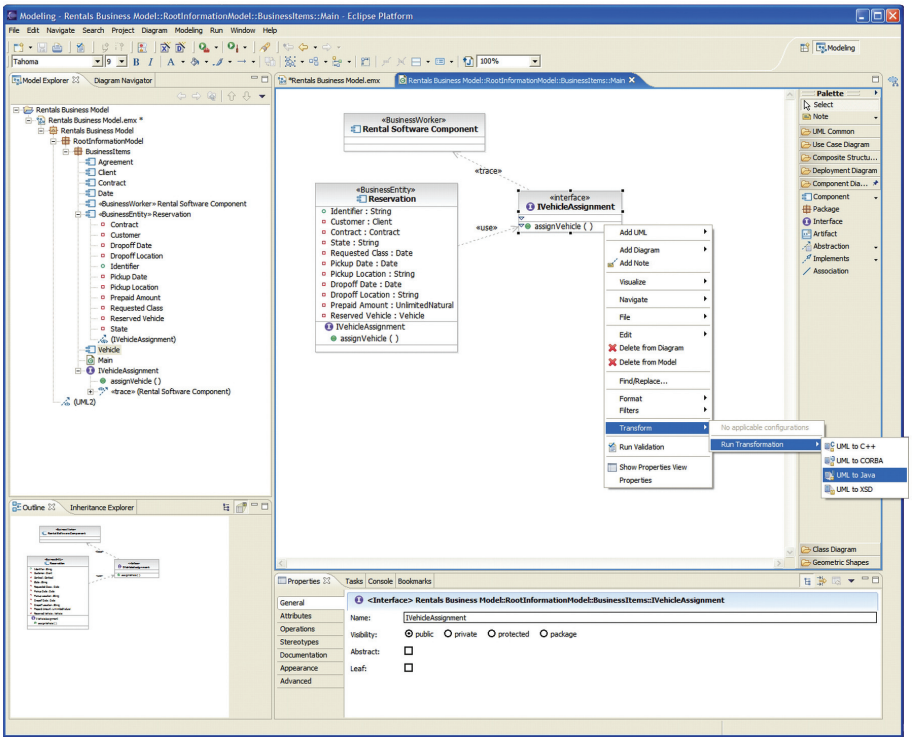

Figure 11. Transforming UML models to Java code using Rational Software Architect

Developers: Constructing the Service

A Java developer then uses IBM Rational Application Developer to implement the architect's specification and construct the service.

With many development tasks automated, the developer can focus on implementing the appropriate business logic needed to fulfill the service's functional requirements in Rational Application Developer. The developer can then deploy and test the code in the WebSphere Test Environment included with Rational Application Developer.

Alternatively, a developer may repurpose existing assets, as Figure 12 shows, and use Rational Application Developer to automate the creation of a Web service from existing Java classes, EJBs or EIS assets. Both Rational Application Developer and Rational Software Architect can take advantage of the UDDI browser included in the tools to find and reuse existing services in solution development.

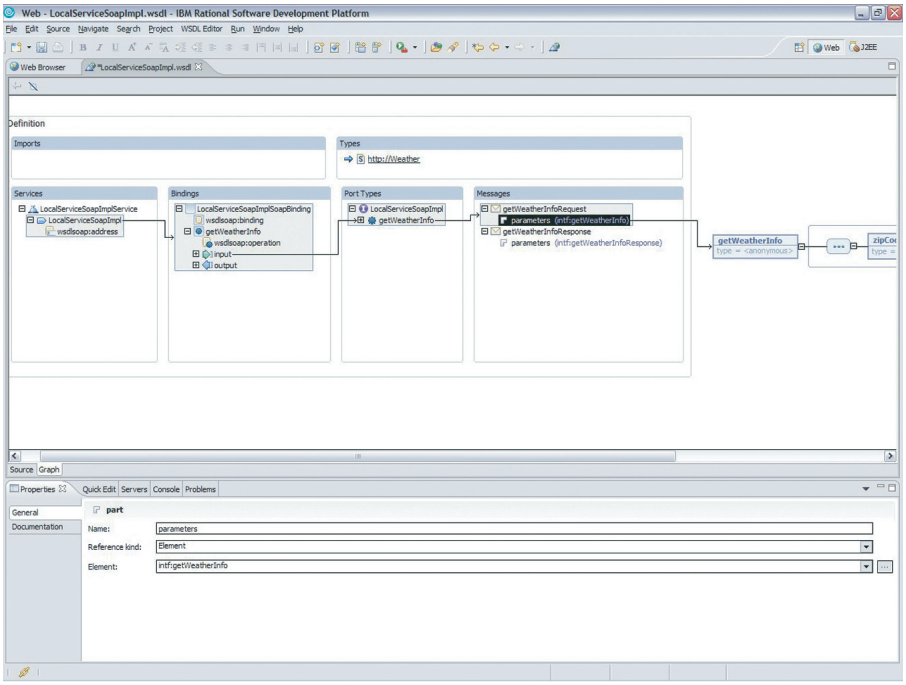

Figure 12. IBM Rational Application Developer makes it easy to create new Web services from existing assets, such as JavaBeans and EJB components, by automatically generating the WSDL files that describe the Web service, a SOAP deployment descriptor, and a test client that can be used to test the Web service.

Using Rational Application Developer, the developer can register the service's availability in a UDDI registry and create both a WSDL description of the service and an Enterprise Archive (EAR) file for the integration developer to use. Once he or she creates these assets, the developer can then package them and store them in a RAS-compliant repository, such as one based on Rational ClearCase.

Testers: Testing the Service

Long before developers have delivered executable code for test, the testing team can begin developing a test plan and test cases. Using the test management tool in the IBM Rational Software Development Platform, testers can leverage requirements in Rational RequisitePro as input assets for the test plan, and link those requirements to the test cases used to validate them.

Once code is available, testers use **IBM Rational Functional Tester** for functional and regression testing to ensure the service functions as designed. Testers can also use **IBM Rational Performance Tester** to assess and verify Web service performance and determine how the service will perform under various usage loads.

Integration Developer: Assembling and Deploying the Service

The role of the integration developer is to choreograph the constructed services as part of a business process workflow. Based on the overall business process model defined earlier in WebSphere Business Modeler and exported as a BPEL, the integration developer brings together the overall workflow by wiring together service implementations. Integration developers use **WebSphere Integration Developer** to import, create, enact, and manage business processes described in BPEL.

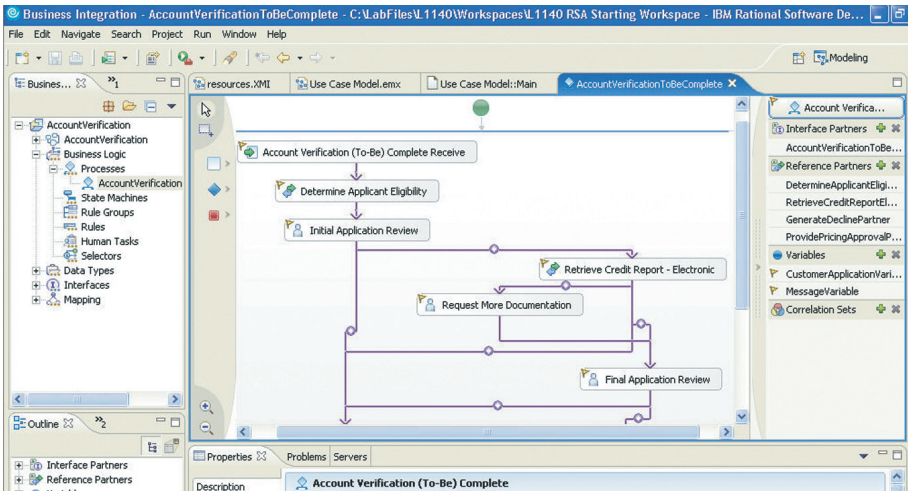

Figure 13. A BPEL realization of a business process in WebSphere Integration Developer

An integration developer can also develop the business processes by linking to a Web service-provided implementation. He or she can include a service found in a UDDI registry as a service provider in the business process. The integration developer may also include human tasks in the business process before testing it. At this point, the specialist will typically deploy the business process to a process execution engine, such as the WebSphere Process Server.

Project Manager and Team: Managing the Asset Lifecycle

Throughout the development lifecycle, the development team uses IBM Rational ClearQuest to automate and enforce the development workflow. Project managers use Rational ClearQuest to assign tasks and track project progress. Developers and testers use Rational ClearQuest as an integrated defect and change request management tool. In addition, Rational ClearQuest can be customized to support asset certification—the process of reviewing, testing, and verifying assets before making them available for a wider audience to consume.

The team also uses IBM Rational ClearCase for configuration management. Because Rational ClearCase is integrated with Eclipse, team members can check out and check in project assets directly from their IDE and greatly simplify parallel development.

To facilitate RAS asset management, teams can use Rational ClearCase together with the RAS Repository for Workgroups, found on alphaWorks.

Via its Web service interface, the RAS Repository for Workgroups supports the following capabilities:

- *Searching and browsing assets in the repository using the RAS 1.0 standard repository service interface, and a second, enhanced interface that supports more complex queries*

- *Retrieving asset information in multiple formats, including viewing the documentation or the feedback and importing the complete asset*
- *Publishing assets to the repository, creating and organizing the logical view of assets in the repository, and tracking metrics*

Project Manager and Team: Leveraging Best Practices and Guidance

By configuring IBM Rational Unified Process with the RUP Plug-In for SOA and the RUP Plug-In for WebSphere Business Modeler, the entire development team has access to SOA-specific process guidance at all times. These RUP plug-ins provide the project manager and the team with guidance and best practices for capturing and simulating business processes and for designing and implementing service-oriented applications.

To help ensure the success of projects and address the root causes of typical software development problems, the team applies RUP practices in the development of individual services and in assembling composite applications.

Getting Started with IBM and SOA

Transitioning to an SOA does not require your organization to perform a complete overhaul of your IT infrastructure and development processes all at once. Often, the transition takes place in incremental steps. You may decide to rework an existing business process using an SOA, or implement a new SOA application. Another organization may begin by adopting SOA-oriented development methodologies and tools for a particular phase of the software development lifecycle. The adoption model shown in Figure 14 illustrates four levels of SOA adoption.

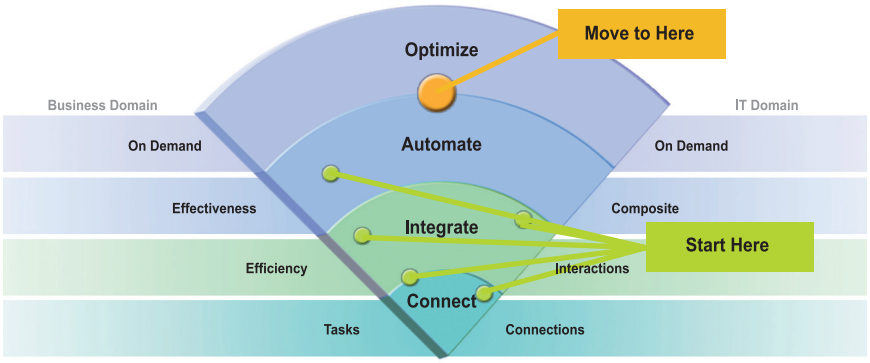

Figure 14. The IBM SOA adoption model

At the *Connect* level, an organization assures reliable and flexible information flow between diverse applications and systems. As the organization progresses, it will establish an integration framework that supports interoperability among heterogeneous environments—removing barriers to building an integrated architecture powered by Web services and non-Web services applications and integration approaches. Next, the organization can begin to automate the orchestration of business and IT processes to align IT with the business goals and grow revenue while containing costs. Ultimately, the business can optimize the process with a holistic approach to transform and manage the organization by aligning strategic and operational objectives with business activities and supporting IT services.

Regardless of your level of SOA adoption, IBM is positioned to help with a complete set of software and service offerings to help you begin or enhance your SOA initiatives. These offerings support all the stages of the SOA lifecycle, including the modeling, assembly, deployment and management of services to build effective SOA solutions.

Summary

Flexibility is a fundamental need for companies seeking to react to a rapidly changing landscape that includes emerging competitive threats, shifting compliance and regulatory requirements, mergers, acquisitions, and evolving technology. Just as important is the ability to align a core businesses process—software development—with business needs and priorities.

Achieving flexibility and better alignment of business and IT objectives requires executing IT projects with a high level of coordination, accuracy, and clarity. Business-driven development for services-oriented solutions helps businesses create solutions that truly meet an organization's needs today and are readily adapted when those needs change in the future.

The IBM Rational Software Development Platform plays an important role in enabling organizations to create service-oriented solutions that can realize these goals. It combines industry-leading tools, process and portfolio management and software configuration management solutions, with support for standards-based and emerging programming models to automate and accelerate service-oriented development.

IBM Corporation
Software Group
Route 100
Somers, NY 10589

Produced in the United States of America
09-05

G507-0956-00