Web Services project roles

The team perspective

Olaf Zimmermann (ozimmer@de.ibm.com), Senior IT Architect, IBM Enterprise Integration
Frank Mueller (mueller.frank@de.ibm.com), IT Architect, IBM Global Services AMS

Summary: This article describes the many different job roles involved in Web services development projects, what their goals are, what their tasks are and how they work with each other. It does not go into detail over the actual tasks to be performed (such as creating a doc/literal service from WSDL); rather it tries to give IT staff of any background overall guidance on what they should be thinking about when approaching a Web services project. The intention is that this article can help an IT department figure out how to organize its projects better and plan for the full picture.

Date: 10 Jan 2004
Level: Intermediate
Also available in: Japanese

Activity: 7240 views
Comments: 0 (View | Add comment - Sign in)

★★★★☆ Average rating (30 votes)
Rate this article

Web services have passed the days when only a few enthusiasts played around with immature yet highly-praised technology, struggling to get anything -- even simplistic, unsecured exchanges of basic data structures up and running. In contrast, during the last two years, the technology has proven its maturity on numerous real-world projects. As a consequence, many technical leaders nowadays consider Web services to be another powerful part of the enterprise application and software integration toolbox, ready to be used in their next big project in this area. So, as Web services usage is spreading over to the "normal" enterprise application project in your organization, you might find yourself to be part of a Web services project team, even if you have never considered yourself to be one of the enthusiasts mentioned above. Now, what role will you have to play? Let's find out what's available!

Why bother?

There are several reasons why you should consider reading this article: if you are a project manager, chief architect, or another technical leader in your organization, you can can find advice on how to structure and staff your first Web services project. Our collection of roles and responsibilities can serve as input to your work breakdown structure. If you are a developer just getting started with Web services, you can learn which tasks and tools exist -- and what keywords you should add to your CV so that your name makes it into the project plan of the next Web services project near you.

Note than this is not an article on developing in teams in general, we focus on the Web services-specific aspects; for instance, roles you do not find in standard J2EE projects, as well as tools and information sources for the practitioners assigned to one or more of these roles.

You might be wondering why we decided to write an article that at first glance appears to be a bit dry. Indeed we agree that applying the technology to solve real business problems is the real fun on any project. However, a good dose of structure and methodology is key to success, and an unsucessful project is never enjoyable, even if it is circled around the hottest technology on the planet. So trust us, it is worth making the effort!

Recap: Web services in a nutshell

Web services solutions and service-oriented architectures (SOAs) include service requestor (client) and service provider (server) implementations, which communicate via SOAP (XML messaging). Web Services Description Language (WSDL) service descriptions provide the glue between requestors and providers. Optionally, a service broker, for example a Univesal Description, Discovery and Integration (UDDI) registry, might be involved. The service descriptions and interactions as well as XML schemas for shared data have to be modeled. Implementations have to be designed, developed, deployed, and tested. So far, so good; not too surprising, you might say, in case you have visited the developerWorks Web services zone before. Now, the question is: how does a project team get there?

Project phases and roles

Any development project runs through different phases, requiring different skills and collaborations throughout its lifecycle. Web services are no exception here. Depending on the methodology used in your environment, you probably have already come across generic terms such as the following:

- Requirements engineering

- Business domain analysis
- Solution architecture outline
- High- and low-level design
- Object-oriented analyis and design (OOAD)
- Various test phases (such as unit, integration, system, acceptance test)
- Going live
- Maintenance
- Management.

Aspects such as service modeling (for example, coarse- or fine-grained interface), choice of SOAP engine (IBM WebSphere SOAP, Apache Axis, or Apache SOAP 2.3), and organizing interoperability tests are first examples of Web services specific considerations. The nature of these issues varies, for example service modeling prerequisites a different skill and mindset than interoperability testing.

The *role* metaphor has proven to be very helpful in this context, bringing chaos to order. Roles are related to project phases and define an abstraction layer decoupling job descriptions and performing resources. All project team members take one or more roles. A *role model* is a common construct in project management and design methodologies. The role concept establishes a commonly understood vocabulary, which has proven to be a very powerful instrument at project initiation time.

So let's now look into such a role model for Web services development projects. For presentation purposes, we divide the roles in our model into three categories. As Web services projects are just another type of development project, it is not surprising that we find a lot of well-known roles here. We define a category of *existing roles* for them. However, some of the existing roles receive additional Web services-related responsibilities; we categorize those under *extended roles*. Finally, there are new roles with special Web services-related responsibilities, which are listed under *extra roles*.

Existing roles

Let us start with four roles you all have seen (or taken) on projects:

Project Manager
Has the overall management and leadership responsibility for the project team. Defines and tracks project plan and work breakdown structure.

Business Analyst
Harvests the business users' functional requirements and provides domain knowledge to the team. Must understand the business language and have industry and domain skills.

Architect
Technical leader of the project. Develops the logical and physical layout (structure) of the overall solution and its components.

Developer
A.k.a. code warrior. No need to introduce this role here!

Security Specialist
Responsible for the definition of security guidelines (policies) and the implementation of security means adhering to these guidelines.

System and Database Administrator
Performs installation and ongoing maintenance work on hardware, operating and database systems, and middleware.

Note that this list certainly is non-exclusive. We could have mentioned any role without Web services-specific aspects here, as they all fit into this category. However, we have limited the list to the most prevalent roles to occur in Web services projects -- this article is not a general project methodology tutorial.

Extended roles

Five standard roles receive additional duties on Web services projects. These roles and their new responsibilities are:

So many people?

Note that any real person can wear several hats. In other words, a single performing resource can take several of the listed roles. For example, your IT architect might also be reponsible for the service modeling. To map roles to persons is a main task for the project manager once the work breakdown structure has been defined in the abstract.

Obviously, letting individuals wear several hats works especially well for the top performers in your team. We sincerely hope you have access to at least one or two of such luminaries!

Product Vendor
Supplies a WS-I-compliant Web services run-time container, and optional service registry and SOAP gateway services.

Deployer

Takes the development artefacts and installs them in the target runtime environment. Generates stubs and skeletons for the target environment from WSDL and installs them together with the service implementations. Provides JAX-RPC mapping and handler configuration through Web services-specific deployment descriptors.

Tester

In charge of the various standard test stages such as integration, load, and acceptance test. Also defines test cases for Web services interoperability and conformance tests.

Toolsmith

Designs and implements project-specific scripts, generators, and other utilities. The degree of standardization in the Web services world makes it possble to, for example, develop custom WSDL-, JAX-RPC- or JSR-109-aware tools.

Knowledge Transfer Facilitator

Provides access to subject matter experts and technical instructors who bring in extended knowledge regarding Web services concepts and implementation assets.

Extra roles

Finally, the time has come to define which additional roles you can see on Web services projects:

SOA Architect

Responsible for the end-to-end service requestor and provider design. Takes care of inquiring on and stating the non-functional service requirements.

Service Modeler

Applies data and function modeling techniques to define the service interface contracts including the shemas of exchanged messages.

Process Flow Designer

Investigates explicit, declarative service orchestration (aggregation, composition) possibilities. An optional role.

Service Developer

J2EE developer familiar with Web services concepts and XML. Develops service interface and implementation (provider side) and service invocation code (requestor side).

Interoperability Tester

Verifies that the developed requestor and provider implementations interoperate seamlessly and ensures Web Services Interoperability (WS-I) conformance.

UDDI Administrator

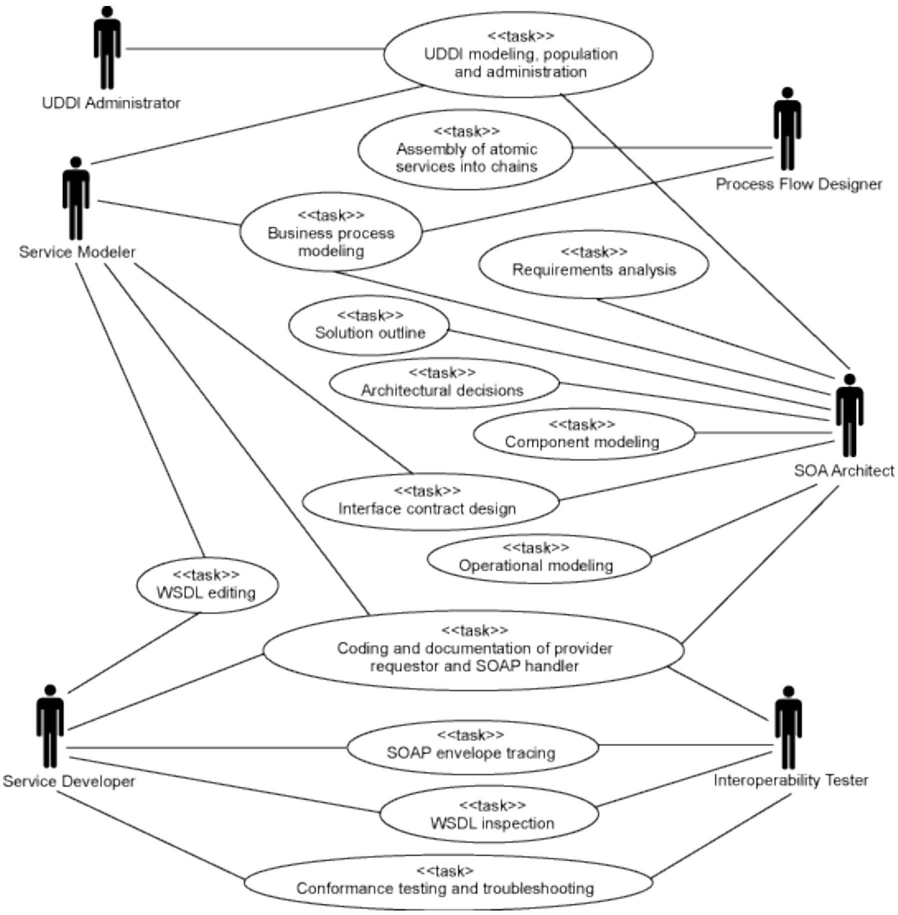
Defines how the generic UDDI data model is customized and populated. An optional role.

Note that our split between extended and extra roles is somehow arbitrary. Both extended and extra roles originate from existing ones (for example, SOA architect and service developer). However, we believe that for the extra roles, the introduction of new names is justified. From now on, we will soleley focus on the extra roles.

Web services-specific roles

Let us now investigate the Web services-specific roles a little further. The following figure shows these roles together with the tasks they perform:

Figure 1. Extra roles in a Web services project and their tasks


UDDI Administrator
<<task>> UDDI modeling, population and administration
<<task>> Assembly of atomic services into chains
Process Flow Designer
Service Modeler
<<task>> Business process modeling
<<task>> Requirements analysis
<<task>> Solution outline
<<task>> Architectural decisions
<<task>> Component modeling
SOA Architect
<<task>> Interface contract design
<<task>> Operational modeling
<<task>> WSDL editing
<<task>> Coding and documentation of provider requestor and SOAP handler
Service Developer
<<task>> SOAP envelope tracing
Interoperability Tester
<<task>> WSDL inspection
<<task> Conformance testing and troubleshooting

The following table shows how these roles interface with each other, which skills are required to perform in these roles, and which tools are available to the performing practitioners:

Project Role	**Performed Tasks**	**Collaborates With**	**Prerequisite Skills**	**Supporting Tools**
SOA Architect	Solution outline Requirements analysis Architectural decisions Component modeling Operational modeling	Any other team member	General IT architectures J2EE technology XML, XML schema Web services concepts and platforms, best practices	UML editors, office suites
Service Modeler	Interface contract design WSDL editing (top-down, bottom-up, meet-in-the-middle)	Business Business Analyst SOA Architect Service Developer	WSDL XML schema and namespaces J2EE technology	WSDL editors, Java to WSDL generators
Process Flow Designer	Business process modeling Assembly of atomic services into chains (processes)	Service Modeler Business Analyst SOA Architect	BPEL4WS, WSDL	Graphical flow modeling tools, BPEL4WS generators Corresponding runtime support
Service Developer	Service provider coding Service requestor coding Provide SOAP header handlers if needed Code documentation	SOA Architect Service Modeler Interoperability Tester	J2EE, XML, SOAP, WSDL	Web services wizards in IDEs WSDL to Java generators
Interoperability Tester	WSDL inspection SOAP envelope tracing Conformance testing Troubleshooting	Service Developers (requestor and provider side)	SOAP, WSDL, WSI Basic Profile Version 1.0	TCP/IP tunnels and monitors WSI test tools
UDDI Administrator	UDDI modeling UDDI population UDDI administration	SOA Architect, Service Modeler	UDDI data model and API knowledge Database design and administration	UDDI browsers UDDI4J

Roles and skills

Identifying roles helps, but finding the right people with appropriate skills is crucial. Don't underestimate the effort. Technologically, Web services usually are rather lightweight and simple; this is part of their power. With Web services, heterogeneous system islands can be opended up for collaboration easier than ever before. However, along with these new possibilities come new sources of errors. Web services projects may be a *new* kind of project for your organization, but chances are they will not be a *simpler* kind of project.

Your project team setup should reflect this with appropriate skill levels. Recommendable is a mix of practitioners who have gained experience on different platforms. This is especially important for your SOA Architect. If such a person is not available to your team, it can be feasible to have additional (part-time) co-architects filling the gaps.

Regarding the tools discussion, we have assumed that J2EE is the service implementation platform of choice. In case other platforms such as Microsoft .NET are involved, additional skills, tools etc. have to be added to the picture. Furthermore, we have deliberately stayed away from product names until now; as you can imagine, a complete stack of Web services tooling and runtime support is available from IBM as well as the open source community. Start your journey at the Eclipse and the Apache Web services open source projects, and do not forget to investigate the IBM WebSphere Studio Application Developer product and the technology previews available at alphaWorks (see Resources).

Assignment of People to Roles

Every role addresses a different aspect of the project as a whole. Earlier on we said that one person typically wears several hats, in other words, acts in more than one role. However, projects risk decreases if different people with broad and diverse skills are on board. There are situations in which only such a purposeful cooperation of different people can unveil the crucial issues of the project and lead to a sound solution. On the other hand, the communication overhead increases with each new team member. There is no single and no simple answer to the roles-to-people mapping challenge. There are many different opinions and controversies on how it should be aproached (even the two authors of this article do not always agree!).

Rather than continuing this debate, let us now take a look at a small example. Consider the following scenario: a fictitious company, say from the insurance industry, has decided to build a new set of mid-office business applications for risk and policy management that has to interface with two different back-end systems. Both back-end systems have been built as J2EE applications -- one uses EJBs, the other one only servlets, JSPs, and JDBC to connect to its customer and contract databases.

During the initial stages of the initiated development project, the roles defined above are assigned to team members. In addition to the Web services-specific activities, the standard project tasks and roles are also identified and assigned as well. The following table shows the results of this work breakdown exercise:

Team Member	Assigned Roles	Collaborates With	Actual Tasks	Selected Tools
1	Project Manager	(all team members)	Project planning Ongoing project control	Spreadsheets, project management software
2	Business Analyst	3	Problem domain analysis	Process modeling tools, office suite
3	SOA Architect Service Modeler Knowledge Transfer Facilitator for Web Services	(all team members)	Software and system architecture WSDL model for service invocations	Rational XDE, WebSphere Studio Application Developer Harvested experience from other projects
4a	Service Developer (Unit) Tester	3, 5, 6	Development of risk and policy management service requestors (clients)	WebSphere Studio Application Developer
4b	Service Developer (Unit) Tester	3, 5, 6	Development two service provider implementations (EJB, non-EJB)	WebSphere Studio Application Developer
5	Tester Interoperability Tester	3, 6	Integrate and connect components developed by 4a and 4b	JUnit, WSI conformance test tools TCP/IP tunnel monitor
6	Service Deployer System Administrator Database Administrator UDDI Administrator	3, 4, 5	Take care of the entire runtime infrastructure	J2EE deployment tools, Ant Product specific administration GUIs

You can see that except for the Project Manager and the Business Analyst, all other team members wear more than one hat. Furthermore, the extra role Process Flow Designer has not been assigned at all, as it is not required in this scenario.

Also note that the example is a rather simplistic one; in real-world project, expect larger teams. We have had good success with team sizes between seven and ten in the core team. It all depends on the scenario at hand; be prepared to split the project into stages in case your work breakdown structure gets too complex to handle. In other words, make sure to plan the project to operate in an iterative and incremental fashion. This gives the team a chance to learn on the job and minizes project risk. In the world of agile development, this principle is called *continuous integration*.

We do not elaborate any further on the fictitious insurance example here. As a matter of fact, it originates from the book *Perspectives on Web Services* (see Resources), where it serves as an end-to-end case study and scenario for the featured reference implementation. This article can be viewed as a small companion to the book -- an extension to its Engagement Perspective if you will.

Conclusion

In any real-world application development project, not just technology can make or break it. Soft facts such as a reasonable work breakdown structure, the right skill mix, and good teamwork are also key to success, many times even more so than the technical solution elements. As Web services are a relatively new technology, a lack of related documented experience in this field can be perceived. There are several Web service-specific roles and other roles well-known from standard development projects receive additonal responsibilities.

Certain additional skills are required, and many tools are available to assist you. The assignment of roles to resources has to be balanced; there is a trade-off between advantages from a high degree of specialization and the related communication overhead. In any case, the "lone wolf" approach certainly does not work for nontrivial projects, and general project management techniques apply to Web services projects just like to any other project.

As members of the IBM services organization, we have had the chance to participate in a number of full-scope Web services projects over the last couple of years. We hoped that with this little article, we could share some of our experience from these projects with you.

Resources

- Start your journey at the Eclipse and the Apache Web services open source projects.
- Investigate the IBM WebSphere Studio Application Developer product and the technology previews available at alphaWorks.
- Check out the Architecture and the Engagement Perspective of *Perspectives on Web Services*. This is the Web site supporting this book.
- An article called Establishing the Services Lifecyclefrom the CBDI forumdiscusses role and other lifecycle aspects beyond development.
- The Java Community Process specification JSR-109 defines its own set of roles and responsibilities.
- Also take a look at methodologies such as eXtreme Programming (XP) and the Rational Unified Process.
- The Web Services Interoperability Organization (WSI) makes its Basic Profile available here.
- In *Death March*, Prentice Hall 1997, ISBN 0-13-014659-5, Edward Yourdon teaches you how to detect and avoid projects that are bound to fail.
- If you prefer to look at positive examples, you can find hundreds of them at the IBM jStart Web site.
- Web services best practices talks such as those by Kyle Brown and Rachel Reinitz as well as Mark Colan, also touch upon roles and responsibilities.

About the authors

Olaf Zimmermann is a Senior IT Architect in the IBM worldwide Enterprise Integration team. His areas of expertise include distributed computing and service-oriented architectures in general, as well as Web services/XML and Java 2 Enterprise Edition (J2EE) in particular. Over the last few years, Olaf has conducted numerous Web services-related engagements, including several production references. He is an author of the Springer text book *Perspectives on Web Services* , ISBN 3-540-00914-0. He also contributed to several IBM ITSO Redbooks such as *Web Services Wizardry with WebSphere Studio Application Developer* , SG24-6292-00. Olaf has been working with IBM since 1994, in areas as diverse as product development, technical presales consulting, teaching, and systems integration. Olaf holds an honours degree in Computer Science from the Technical University in Braunschweig, Germany.

Frank Mueller is an IT Architect in the IBM Global Services AMS Solution Implementation organization. He helps IBM key account customers to select and design software solutions in support of core business processes and covers the full application development lifecycle. On the basis of his technical backgound in J2EE technology and XML, Frank got in touch with emerging Web services technology rather early in 2001 and started to work toward service-oriented architectures (SOAs) even before this became a well-established term. Frank has an honours degree in Computer Science from the University in Karlsruhe, Germany.

Close [x]

developerWorks: Sign in

If you do not have an IBM ID and password, register here.

IBM ID:

Forgot your IBM ID?

Password:

Forgot your password?
Change your password?

After sign in: Stay on the current page

By clicking **Submit**, you agree to the developerWorks terms of use.

Submit Cancel

The first time you sign into developerWorks, a My developerWorks profile is created for you. This profile includes the first name, last name, and display name contained in the profile you created when you registered with My developerWorks. Selected information in your My developerWorks profile is displayed to the public, but you may edit the information at any time. Your first name, last name (unless you choose to hide them), and display name will accompany the content that you post.

All information submitted is secure.

Close [x]

Choose your display name

The first time you sign in to developerWorks a profile is created for you, so you need to choose a display name. Your display name accompanies the content you post on developerWorks.

Display name: (Must be between 3 – 31 characters.)

Note: Please choose a display name between 3-31 characters. Your display name must be unique in the developerWorks community and should not be your email for privacy reasons.

By clicking Submit, you agree to the developerWorks terms of use.

Submit Cancel

All information submitted is secure.

★★★★☆ Average rating (30 votes)

○ 1 star ★☆☆☆☆ 1 star
○ 2 stars ★★☆☆☆ 2 stars
○ 3 stars ★★★☆☆ 3 stars
○ 4 stars ★★★★☆ 4 stars
○ 5 stars ★★★★★ 5 stars

Submit

Add comment:

Sign in or register to leave a comment.

Note: HTML elements are not supported within comments.

1000 characters left

Post

Be the first to add a comment

Trademarks | My developerWorks terms and conditions