Written by Eilon Reshef

Building Interactive Web Services with WSIA & WSRP

WSIA and WSRP are new Web services standards that enable businesses to create user-facing, visual, and interactive Web services that organizations can easily plug-and-play into their applications and portals. This article will familiarize you with these technologies and illustrate how they can help your businesses.

One of the main promises of Web services is to enable the assembly of Web applications from functional components distributed across multiple locations. However, until now the assembly of visual, rich, interactive Web applications with a cohesive flow and look-and-feel has been a challenge. Custom programming is required to create a user interface tier for each new Web service, resulting in set-up and maintenance efforts that render business initiatives cost prohibitive as the number of components increases.

Web Services for Interactive Applications (WSIA) and Web Services for Remote Portals (WSRP) are standards for user-facing, presentation-oriented, interactive Web services intended to simplify the creation of distributed interactive applications.

With WSIA and WSRP, Web services include presentation and multi-page, multi-step user interaction. This lets service users plug them into sites and portals without custom development and to leverage new functionality available in future versions of the service without the need for additional custom development (see Figure 1).

AUTHOR BIO:

Eilon Reshef is the vice president of Products at WebCollage. WebCollage provides comprehensive software solutions for Interactive Web Services, allowing companies to package their existing customer-facing Web applications as Web services and rapidly share them with multiple business partners. EILON.RESHEF@WEBCOLLAGE.COM

History

WSIA and WSRP stem from parallel efforts initiated in the middle of 2001. Portal vendor Epicentric spearheaded the Web Services User Interface (WSUI) initiative to address the lack of a standard user-interface layer as part of the Web services stack. IBM initiated its own effort: Web Service eXperience Language (WSXL). WebCollage, a software vendor providing a platform for integrating interactive Web applications, wanted to standardize its customer implementations based on an initiative called Interactive Web Services. In parallel, many portal vendors recognized the need to address the same problem in the context of portal toolkits: how to quickly plug remote interactive services (called "portlets") into a portal without custom programming for each remote service.

The efforts were consolidated into two working groups under the umbrella of OASIS, the organization behind ebXML and other XML-related standards. WSIA focuses on the framework for creating interactive Web services, and WSRP defines the interfaces to include portal-specific features. Today the working groups include more than 30 industry-leading vendors from different industry segments: application server vendors (BEA, IBM, Oracle, Novell, Sun), pure-play portals (Epicentric, Plumtree), interactive application integration vendors (WebCollage, Kinzan, Citrix) and enterprise application providers (Peoplesoft, SAP).

Version 1.0 of the specifications includes the interfaces common to the two groups, and is currently in a review process.

WSIA/WSRP and the Web Services Stack

WSIA and WSRP define a set of APIs that allow applications to leverage remote interactive services. The APIs are built on top of the existing standard Web services technologies:

One step toward making

Web services what they were meant to be

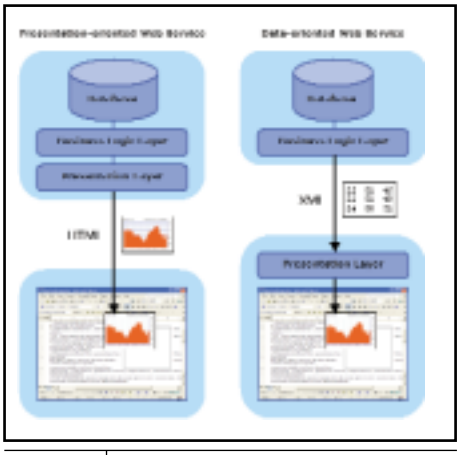

FIGURE 1 **Data vs. presentation-oriented Web services**

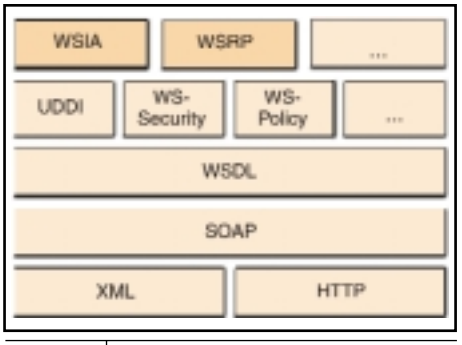

FIGURE 2 **WSIA/WSRP and related technologies**

- SOAP for the service invocations
- WSDL for formally describing the WSIA and WSRP service interfaces
- UDDI for publishing, finding, and binding the WSIA and WSRP services

WSIA and WSRP will leverage emerging Web services security and policy standards as they become available (see Figure 2).

Business Scenarios

WSIA and WSRP can be used in a variety of business scenarios. Described below are three scenarios considered in the working groups.

The "Indirect Customer" Gap

One of the common gaps in e-commerce today is between manufacturers and their indirect customers. For example, a consumer who researches electronic products at a retailer site online cannot find the information and tools needed for the process because the retailer sites usually lack the advisor and configuration tools typically used for this kind of task. The consumer is forced to go to the manufacturer site to find these online applications, and then go back to the retailer site to complete the purchase.

In the ideal scenario, online distributors and resellers would be able to offer manufacturer tools and content as part of their e-commerce offering. However, the technical challenge is that many of the tools developed by manufacturers – online advisors, product configurators, product capsules – are sophisticated in nature. Thus, they cannot be easily described using XML APIs. In addition, XML APIs don't provide the simplicity needed to scale integration to thousands of channel partners because they require each reseller to develop custom code for each of the manufacturers APIs.

WSIA allows companies who sell indirectly to package their online sales and service effectiveness tools as WSIA/WSRP services. This makes it possible for their channel partners to easily plug the tools into their own e-commerce sites and offer them to their customers. It allows manufacturers to make it easier to buy their products while maintaining their traditional role in the distribution chain.

The "Corporate Employee" Gap

One of the common goals for business-to-business service providers (e.g., providers of 401(K) programs) is to transition as many processes into self-service tasks over the Web as possible to reduce operating costs. The main challenge is that the target audience for these online services is mainly corporate employees who interact primarily with their employer's intranet. To locate and use an external service, the employees must leave their familiar environment and locate the remote service. This extra effort reduces

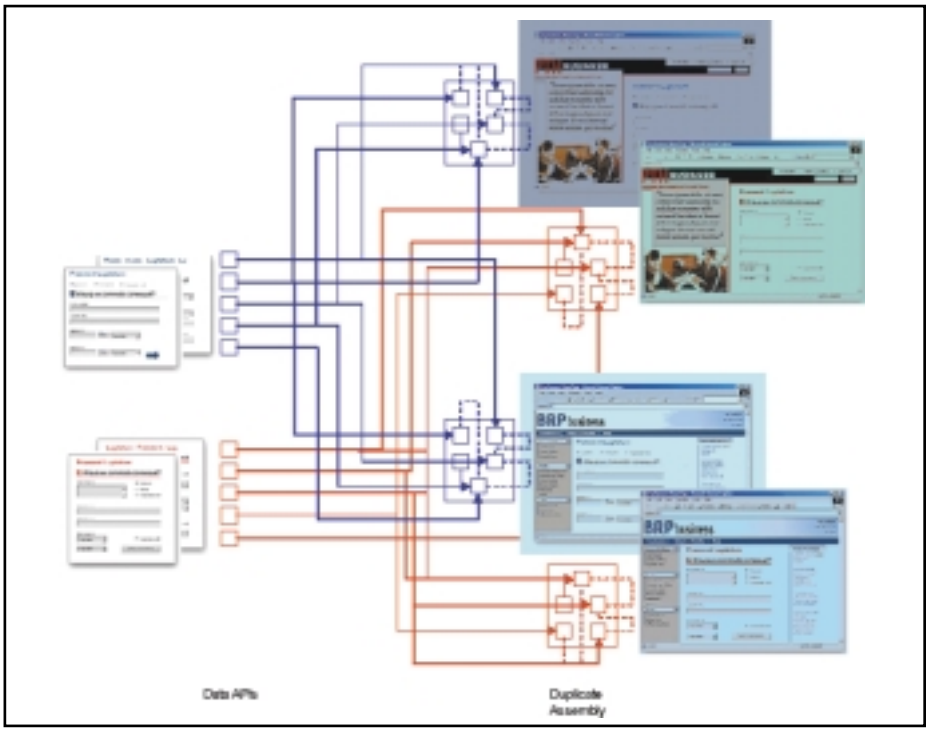

FIGURE 3 | **Assembly with XML APIs**

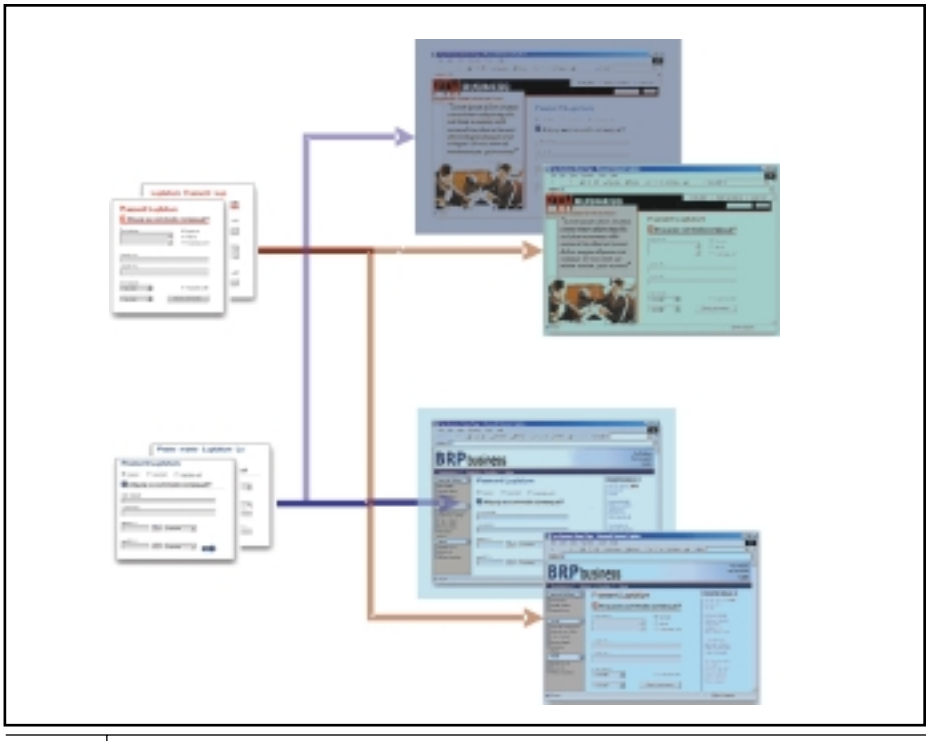

FIGURE 4 | **Plug and play with WSIA/WSRP**

the adoption rate for the online services offered.

In the ideal scenario, employers would be able to integrate such external services into their intranets and their corporate portals, and make them easily available to employees. However, the current cost of integration makes it cost-prohibitive to all but the largest organizations. With WSIA and WSRP, providers of service applications can package their financial services applications as WSIA/WSRP services, making it easy for corporate customers to integrate into their intranets. It allows business-to-business service providers to develop a closer relationship with their corporate customers and with its employees.

The "Build Once Deploy Anywhere" Problem

Fortune 500 companies provide a variety of online services to their employees, customers, and business partners. To do so, they often create multiple portals – one for each target audience.

One of the main challenges such companies face is the need to make a single application selectively available through each of those portals. With WSRP, companies can package their applications as "portlets" and make them available to portal administrations for integration. Portal administration can plug those portlets and make them available to the portals' end users without custom integration, while the application would still be centrally hosted and managed.

Technical Motivation

In many of the scenarios described above, WSIA and WSRP provide an alternative to data-centric Web services and simple XML APIs. When exposing only an application's business logic with XML APIs, API users incur the effort of integrating the application into their sites. The approach falls short for complex interactive applications whose flow spans several Web pages because (see Figures 3 and 4):

- Application providers need to decompose the interactive application into the underlying atomic functions, which requires a significant effort.
- API users need to recompose the underlying APIs with the workflow and the presentation into a coherent interactive application, resulting in high setup costs for the API user.
- As the APIs evolve, API users need to upgrade their application to comply with the latest interface to incorporate new functionality. This results in high maintenance costs for the API user.
- It's hard to ensure that the quality of the re-assembled application meets the application provider's quality standards. The quality of the implementation depends heavily on the technical

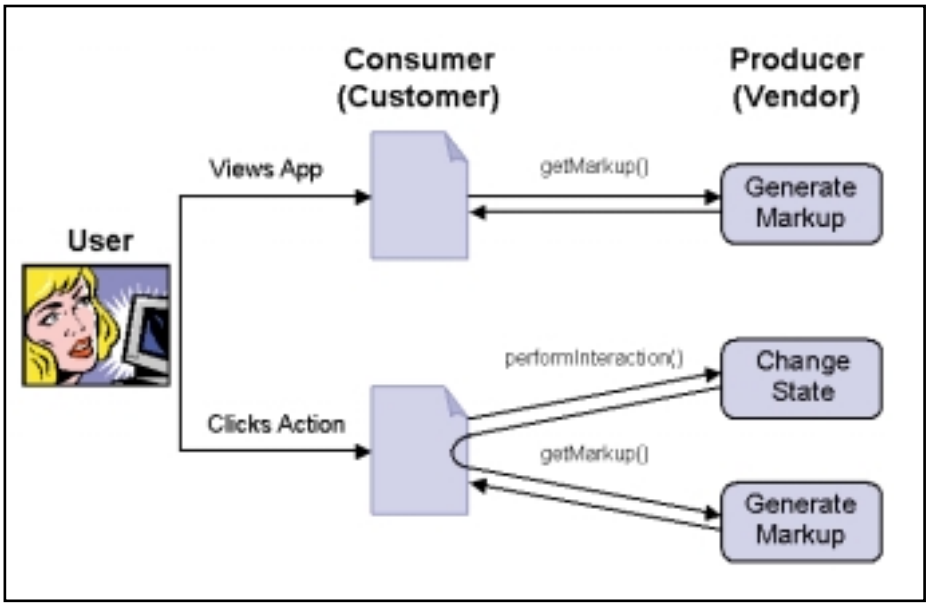

FIGURE 5 | **WSIA/WSRP Data Flow**

expertise of the API users, and the functions may not be assembled in the right way in each instance.

These problems stem from a single fact: instead of sending ready-to-use (customized) HTML markup fragments that the API user can embed without any further processing into their Web site, data-oriented Web services require further work, often quite complex, to transform the XML into HTML, and manage the multi-step interaction. Such an approach results in code duplication, because each API user has to re-implement the same composition logic of the application.

Initial vendor "hype" in the Web services space suggested that new assembly tools would solve this problem by providing "assembly platforms" that automate the creation of interactive applications. In reality, companies realized that using XML APIs and SOAP is no easier than using APIs available today in local code libraries. To create a composite application, developers must understand the application-specific semantics of the different functions offered, and invest time and resources to compose the different functions into a usable interactive application.

Many high-value business scenarios involve complex, interactive services that require a user interaction and a presentation model. By setting an industry standard to share this user interaction and presentation, WSIA and WSRP pave the way for providing richer and more compelling Web services for businesses and users alike.

WSIA/WSRP Technology Overview

WSIA and WSRP define a set of APIs that allow developers to produce and consume remote interactive Web services. They define three types of actors:

- Producer: Represents the service provider hosting the remote interactive Web service (for example, weather.com as a weather service provider).
- Consumer: Represents the entity integrating the remote service into its Web application, oftentimes using a portal toolkit (for example, Yahoo Weather or a corporate portal).
- End User: Represents the person that comes to the Consumer Web site to use the Producer's application in the Consumer's context.

In a nutshell, WSIA and WSRP fulfill the following roles:

- Define the notion of valid markup fragments based on the existing markup languages such as HTML, XHTML, VoiceXML, cHTML, etc.
- Provide a set of standard calls to enable a Consumer to request these markup fragments from a Producer based on the existing state.
- Supply a set of calls that support the concept of multi-step user interaction and preservation of state across those calls.

There are four central parts to the WSIA and WSRP APIs:

- Retrieving markup fragments (encapsulated in the getMarkup() call).
- Managing state
- Handling user interaction (encapsulated in the performInteraction() call).
- Supporting customization and configuration.

Generating Markup Fragments

The getMarkup() call requests a markup fragment based on the state of the Web service.

When the end user views a page that includes a remote interactive service, the Consumer invokes the getMarkup() call. The operation receives state information (see below) and returns a fragment of standard HTML code. The Consumer embeds this markup into the page, and returns the completed page to the user. In the case of WSIA, the container page can be part of any Web application. In the case of WSRP, the container page is generated by a portal toolkit.

There are certain guidelines governing the markup returned by the getMarkup() call. The most important one relates to user interaction. When the markup contains links and forms, they point back to the Consumer application, with the Web service's new state. This process, which is called URL rewriting, ensures that any further user interaction with the interactive service is routed through the Consumer application (see below). It also ensures that the next getMarkup() is invoked with the new state.

Managing State

State is critical because WSIA and WSRP services are typically interactive (for example, a three-step checkout process in an online store), requiring several calls from the Consumer to the Producer, each dependent on previous configurations, data entry, and actions.

Because WSIA and WSRP are connectionless protocols, the Producer must be able to return information to the Consumer, with the understanding that this information will be sent back by the Consumer. Two

types of state information exist:

- ***Navigational state:*** Allows the current page to be correctly generated, including on a page refresh or through a bookmark. This type of state is sent from the Producer back to the Consumer, which typically stores it in the URL. It is transferred in a parameter called navigationalState.
- ***Transient state:*** While navigational state defines the current "page" of the Web service, transient state is stored on the Producer and usually related to a sequence of operations. The Consumer is sent only a "handle" to this state, much like HTTP sessions. If the Producer decides to use a transient state, the Producer returns the handle to it. The Consumer is then responsible for attaching it to any subsequent calls.

Handling User Interaction

To support user interaction, all the URLs embedded in the markup fragment returned by the remote Producer service point back to the Consumer application. To do so, the Consumer sends a URL template as part of the invocation of the getMarkup() call. For example, the consumer may send the following URL template:

```
http://www.consumer.com/path?ns={navigationState}&si={sessionID}
```

The Producer is responsible for generating a markup fragment in which all the interaction URLs point back to the Consumer. For example, the Producer may generate a link that points to the following URL:

```
http://www.consumer.com/path?ns=page2&si=4ABB33A
```

WSIA and WSRP also provide an alternative mechanism that allows the Producer to create URLs that conform to a predetermined pattern. The specification allows Consumers to parse the markup and to rewrite such URLs to point back to their application.

Consequently, when the user clicks on a link or submits form data, the Consumer application gains control, and has access to the action carried out by the user. The Consumer application then invokes the performInteraction() call. Upon receiving the call, the Producer handles the action and returns an updated state. To redraw the complete page, the Consumer then invokes the getMarkup() call to receive a markup fragment. Because the state of the Producer has changed since the previous getMarkup() call, the markup fragment returned is typically different from the one previously returned. The end user can then perform another action, which starts a new interaction cycle (see Figure 5).

Supporting Customization and Configuration

To support a situation where a single centrally hosted service can be used across multiple Consumer applications and across multiple individual users, WSIA and WSRP support multiple configurations of a single service. For example, a remote interactive product catalog may be configured to display different prices depending on the Consumer application.

WSIA and WSRP provide a set of function calls allowing Producers to expose multiple pre-configured versions of the same service. It also allows Consumers to create and manage additional configurations of the same service, as well as allowing end users to create such configurations. In the context of WSRP, such configurations are static (that is: defined in advance), whereas WSIA plans to add support for dynamically configured remote services.

The Market and Future of Interactive Web Services

WSIA and WSRP are important technologies that help bring the promise of Web services to end users, by providing a standard to manage user interaction and application display. They enable business partners to integrate each other's online applications seamlessly, offering a more compelling experience to their customers.

These technologies are complex under the hood and can only thrive if vendors deliver on the promise of building tools to manage the complexity. Given the breadth of the specification, it is not expected that companies will develop WSIA and WSRP solutions in-house. It is likely that they will instead rely on tools from vendors, keeping their focus on creating the application functionality.

Although the WSIA and WSRP standards are still evolving, several vendors are already providing practical solutions in this space. Such solutions will likely migrate to the standards as they mature. You may expect advances to be made in the following directions:

- Epicentric (www.epicentric.com) and portal vendors are focusing on aggregating existing remote Web services into portals.
- IBM (www.ibm.com) and J2EE application server vendors are building tools to make it possible to create Web services from scratch that can harness the power of WSIA and WSRP.
- WebCollage (www.webcollage.com) and interactive application integration vendors are focusing on helping companies package their existing Web applications, so that they can be used remotely by customers and business partners alike.

As the Gartner Group pointed out in a recent analysis of the Web services space, "Successful software vendors and Web services providers will find innovations in usability and user interface to be a source of competitive advantage. Better-than-average usability is one reason why Yahoo, Amazon, AOL, Google, and Palm came to dominate their respective markets."

The vision of intertwined Web services with rich user interfaces, providing a complete package to the end-user, can be realized only if the presentation and user interaction problem is solved. WSIA and WSRP are a step in the right direction and go a long way towards making Web services what it was designed to be: pluggable application components that can be assembled into visually rich composite Web applications.