Zend Technologies Library

Zend Whitepaper System i

Integrating PHP with System i using Web Services

Author: Sam Pinkhasov, Zend Technologies

As is the case with most development teams tasked with maintaining legacy applications, many System i shops continue to maintain valuable data that is trapped within their applications. In most cases, these applications are not easily integrated with external applications or services. Fortunately, System i developers now have a new tool in their toolbox to help them release their data, PHP. PHP provides the tools System i developers need to easily expose their data and services to external applications via Web Services. Using PHP along with Zend's developer products, a Web Service can easily be written to wrap almost any existing program.

Zend provides a complete set of products that support System i integration that span from development to production. Zend's solutions include native System i database support and System i objects access. (e.g., RPG/COBOL/CL programs, CL commands, spooled file entries, data area and many more). In this article we will show how quick and simple it is to create a Web Service wrapper for an existing RPG program.

Web Service

So exactly what is a web service? Wikipedia tells us this:

> *The W3C defines a Web service as a software system designed to support interoperable machine-to-machine interaction over a network. Web services are frequently just application programming interfaces (API) that can be accessed over a network, such as the Internet, and executed on a remote system hosting the requested services.*

A definition more germane to our discussion however, is this one.

> *A Web Service is a unit of application logic providing data and/or services to other applications.*

Stripping away the tech-speak, a Web Service enables an application to expose what it does or what it knows so that other applications can make use of it.

An example of this could be, a company using a CRM that exposes its core functionality as an API. The CRM now becomes a Web Service exposing what it does to other applications. The corporate website's process for collecting leads can now utilize that Web Service to insert new leads directly into the CRM as they become available. Since Web Services are wrapped around the core business logic of the CRM, new leads can be acted upon as if they were manually entered.

In short, Web Services delivers on the promise of process inter-operability made by so many technologies before it. Web Services puts Service Oriented Architecture (SOA) easily within our grasp. PHP makes Web Services on System i easy.

Currently there are many Web Services available on the Internet. Each of them can be used to extend existing applications and give new functionality to the users.
The web site www.programmableweb.com tracks publicly available APIs. It lists more than 350 APIs, ranging from mapping and address correction to data storage. Since some Web Services carry fees or access restrictions it is important to review the appropriate documentation before implementing to make sure there are no nasty surprises after you have deployed. The list of ways you can extend existing applications via web services grows on a daily basis and it limited only by the imagination. Web Services opens the door for software applications written in different languages to be able to exchange data and services.

Until recently, creating a new Web Service was not a trivial task. Building Web Services still requires knowledge of OOP, XML and SOAP. However Zend's products shield developers from most of the complex details. Now building Web Services on System i has been greatly simplified. This allows developers to spend less time on the details of Web Services so they can spend their energy solving the business problem at hand. This would be a good point to stop and take a look at what goes into creating a Web Service using Zend's tools that will wrap existing RPG program.

Zend Whitepaper System i

Creating a Web Service

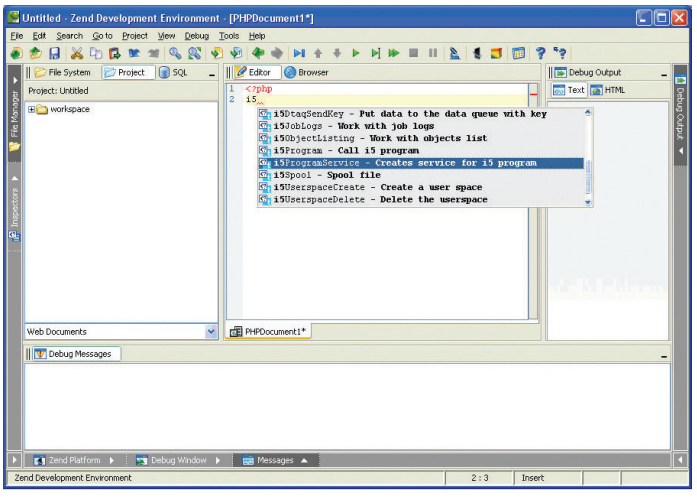
Let's see an example of how easy it is to create a simple Web Service that wraps an existing RPG program containing two parameters: a code number and a description name. The Web Service is to receive a code number as an input parameter and will return a description name as the output parameter.
Zend Studio for i5/OS includes templates to help users build PHP applications quickly. The templates are available via a Code Completion pop-up menu.

1. Start Zend Studio (in Microsoft Windows: click **Start | Programs | Zend Studio for i5/OS | Zend Development Environment).**
2. Enter the following lines into Studio's editor:

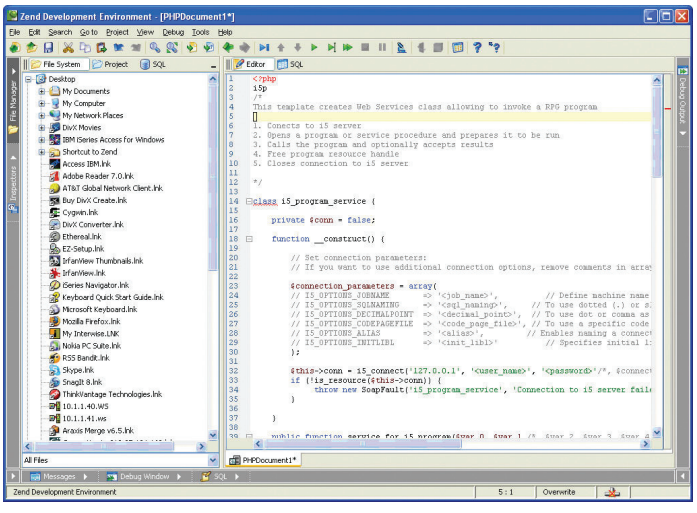
```
<?php
i5P
```

A pop-up containing a list of templates/functions will appear as soon as you begin to type (or click Ctrl + spacebar).

3. Select i5ProgramServices.



This template is one of many templates included with Zend Studio for i5/OS. It includes all the code required to create a Web Service wrapper for an RPG program.
Zend Studio's Editor uses its Code Completion pop-up to display template / function choices that begin (in this example - with the characters "i5"). After selecting the i5ProgramService Web Service template, the editor inserts the code that was pre-typed in the template. See the figure below.



4. Use the TAB key to scroll through the program variables and edit them (as required). For our example, we need to stop the TAB key on the variables *<user name>* and *<password>* so that we can change them to our i5/OS user profile and password.

```
15      private $conn = false;
16
17      function __construct() {
18
19          // Set connection parameters:
20          // If you want to use additional connection options, remove comments in array block
21
22          $connection_parameters = array(
23          // I5_OPTIONS_JOBNAME       => '<job_name>',          // Define machine name by defaul
24          // I5_OPTIONS_SQLNAMING     => '<sql_naming>',     // To use dotted (.) or slashed (/)
25          // I5_OPTIONS_DECIMALPOINT  => '<decimal_point>',  // To use dot or comma as decimal s
26          // I5_OPTIONS_CODEPAGEFILE  => '<code_page_file>', // To use a specific code page
27          // I5_OPTIONS_ALIAS         => '<alias>',          // Enables naming a connection
28          // I5_OPTIONS_INITLIBL      => '<init_libl>'          // Specifies initial library
29          );
30
31          $this->conn = i5_connect('127.0.0.1', '<user name>', '<password>'/*, $connection_param
32          if (!is_resource($this->conn)) {
33              throw new SoapFault('i5_program_service', 'Connection to i5 server failed, use i5
34          }
35
36      }
37
38      public function service_for_i5_program($var_0, $var_1 /*, $var_2, $var_3, $var_4, $var_5,
39
```

5. Next, specify the RPG program and library name that we want to call for this service. Move the TAB to *<library_name>* and *<program_name>* variables and specify/name them. Because the template already has two RPG program parameters defined, we are finished editing.

```
63          for STRUCT - array containing data definition of the structure
64          for INT, FLOAT - ignored
65
66          Count (optional) - repetition count if the field is an array
67          CountRef (optional) - reference to the repetition count if the field is an array */
68
69          $description = Array (
70          array ('Name' => '<name_0>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
71          array ('Name' => '<name_1>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
72        // array ('Name' => '<name_2>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
73        // array ('Name' => '<name_3>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
74        // array ('Name' => '<name_4>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
75        // array ('Name' => '<name_5>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
76        // array ('Name' => '<name_6>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
77        // array ('Name' => '<name_7>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
78        // array ('Name' => '<name_8>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
79        // array ('Name' => '<name_9>', 'IO' => I5_INOUT, 'Type' => I5_TYPE_CHAR, 'Length' => '<
80          );
81
82          $prog = i5_program_prepare('<library_name>/<prog_name>', $description, $this->conn);
83
84          if (is_resource($prog)) {
85
86           /* Execute Program */
87              $params = array (
```

In order to call an RPG program containing additional parameters, simply uncomment the appropriate number of commented program parameters (included in the template) or type additional parameters as required.

6. Next, save the code by clicking **File | Save**.

Now we need to create a WSDL file. **WSDL** files are XML format files used for describing networks services as a set of endpoints operating on messages containing either document-oriented or procedure oriented information.

Zend Studio for i5/OS provides a WSDL generator (Tools | WSDL Generator) that allows you to generate WSDL files by:

- Defining which files and classes are going to be extracted,
- Providing the URL of each class (mapped to a port) and
- Configuring the global setting for WSDL files (name convention, binding options and encoding style).

Enter the required information and click **Finish**. This will create the WSDL file and open it in Zend Studio's Editor. The figures below show the WSDL wizard dialogs:

Zend Whitepaper System i

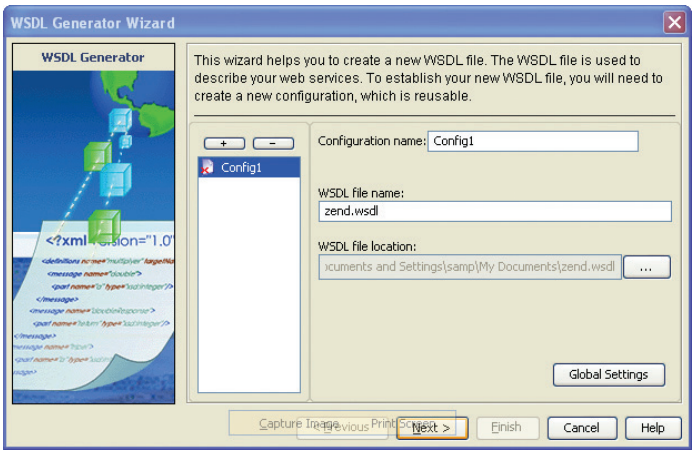


Figure 1 *Create WSDL*

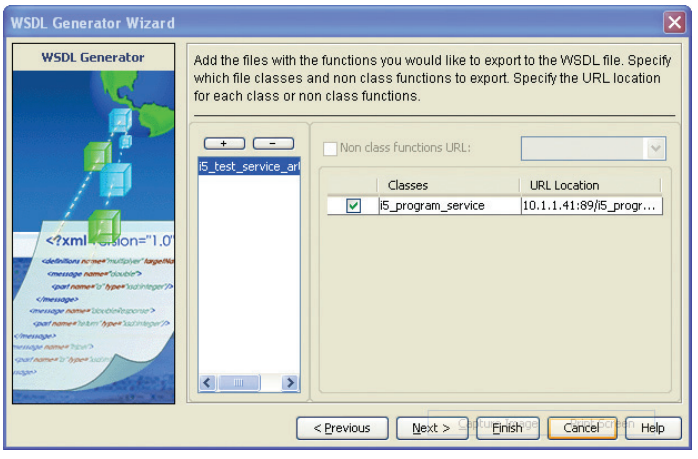


Figure 2 *Export Functions to WSDL*

Here is the WSDL file generated by the WSDL Generation Wizard.

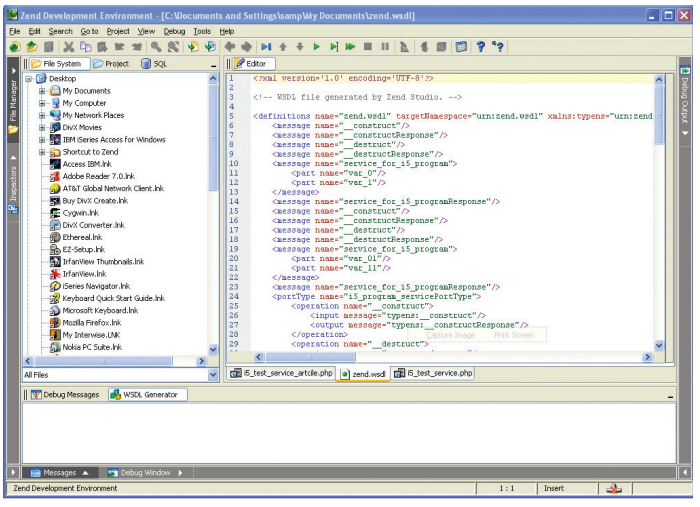


Figure 3 *WSDL File*

Finally, we need to create a client program to call the service. The following code defines the WSDL file for the service.

```
<?php
ini_set('soap.wsdl_cache_enabled', '0');
$my_client =  new SoapClient('zend.wsdl');
?>
```

That's it !

Zend Studio for i5/OS has made creating a Web Service a matter of a few simple steps. Now the developer can concentrate on the business logic necessary to expose his data or services to the world.

Deployment

In order to deploy, simply move the WSDL file and the client program to a place where the web service can access them. We are now able to make a call that will execute our RPG program from any place in the world.

Web Services Benefits

Now that we've examined the how of Web Services, let's take a look at the why, "Why use Web Services"?

The first reason is the easiest to measure. Using Web Services shortens the application development cycle and reduces costs. This is a direct and measurable metric. Since Web Services are language agnostic, you no longer need to re-implement concepts every time technology changes or you start working with a new language. Any language that can access the web can now access your Web Service. As we stated at the beginning of the article this means that existing business logic captured in legacy systems can be re-used instead of re-invented. Other technologies in the past have promised code re-usability but again, Web Services actually delivers on that promise in a real and measurable way.

Closely related to reusability is the idea that each Web Service encapsulates a single concept. Most Web Services are written to perform an atomic task. This means that properly designed Web Services will become the building blocks for your more complex applications. Many common business routines can now be replaced by a Web Services and reused throughout an enterprise's application space. This benefit translates into lower overall maintenance costs as only one code entity needs to be maintained (i.e. the Web Service itself).

Given that Web Services are re-usable and atomic, they now become the building blocks of Service Oriented Architecture (SOA). In SOA, each service is designed as a stand-alone block of business functionality which is loosely-coupled as well as being highly inter-operable. Examples of this could be services that:

- Verify an account number
- Validate customer payment
- Send eMails
- Synchronize/transfer data between systems
- etc.

Since any given Web Service is reusable, it can be applied to Composite Applications. New applications can now be created by using a set of Web Services to access legacy data and applications, adding new business rules to deliver a new solution.

Zend Whitepaper System i

Conclusion

Web Services mean different things to different developers. In many cases, they are simply a way to add new "whiz-bang" to an existing web page through AJAX. However, to **System i** developers, Web Services hold the promise of being able to re-use and extend legacy applications with a minimal effort. Beyond simple mash-ups displaying your customers on a world map, Web Services can help you reach and react to them in new and expanding way.

What else is possible with i5/OS and Zend?

- Improve PHP performance and enhance control over your development, testing and production environments with Zend Platform for i5/OS – the PHP Application Server that provides deep insight into your PHP applications. For more information visit: http://www.zend.com/products/zend_core/zend_for_i5_os#zpi5

- Make the software installation easier with Zend Core for i5/OS the fully tested and enhanced version of the open source PHP. Zend Core is the only Zend certified and supported version of PHP for i5/OS. http://www.zend.com/products/zend_core/zend_for_i5_os#zci5

- Speed development cycles and simplify complex projects with Zend Studio for i5/OS – the industries leading PHP Integrated Development Environment (IDE) designed for professional developers, which includes all the development components necessary for the full PHP application lifecycle. Free for System i users! http://www.zend.com/products/zend_core/zend_for_i5_os#zsi5

Zend
The *php* Company
www.zend.com

Corporate Headquarters:

Zend Technologies, Inc.
19200 Stevens Creek Blvd.
Cupertino, CA 95014
Tel: 1-888-PHP-ZEND
1-888-747-9363
Fax: 1-408-253-8801

Central Europe:

Zend Technologies GmbH
Bayerstrasse 83
80335 Munich, Germany
Tel: +49-89-516199-0
Fax: +49-89-516199-20
E-Mail: info-germany@zend.com

International:

Zend Technologies, Ltd.
12 Abba Hillel Street
Ramat Gan, Israel 52506
Tel: 972-3-753-9500
Fax: 972-3-613-9501

UK:

Zend Technologies
50 Basing Hill
London NW11 8TH, United Kingdom
Tel.: +44 20 8458 8550
Fax: +44 20 8458 8550

Italy:

Zend Technologies
Largo Richini 6
20122 Milano, Italy
Tel.: +39 02 5821 5832
Fax: +39 02 5821 5400

France :

Zend Technologies SARL
5, Rue de Rome, ZAC de Nanteuil
93110 Rosny sous Bois, France
Tel : +33 1 4855 0200
Fax : +33 1 4812 3132

0124-M-WP-0307-R2-EN