SOAP vs. REST API Implementation

Having spent copious amounts of time implementing and working with various APIs in the past, particularly using Simple Object Access Protocol (SOAP) and Representational State Transfer (REST) I thought it was about time I gave back some of the information all to do with the intricacies of creating a successful API implementation using either of these technologies.

It should be noted that, as I am predominantly a PHP developer and most of the APIs I have implemented have been written in PHP, this article has been written primarily with PHP development in mind. This does not mean that all of the content is useful solely to PHP developers however the code examples will only be useful for PHP.

So, on with the article.

Which to choose: SOAP or REST

From my experience there's a number of things that need to be considered when determining whether to go with SOAP or REST for your API.

API requirements

The requirements of your API should play a big part in helping you decide whether to go with SOAP or REST or any other technology for that matter. The things you need to consider include:

1. The requirements of the actual API methods and their arguments/data. How simple or complex are the methods?
2. The limitations of your environment/host. Are there any PHP or web server (apache) limitations that might restrict the effectiveness of SOAP or REST?
3. The volume of traffic that will be going through the API. SOAP will have a greater overhead than REST.

Generally speaking my recommendation would be that if your API requirements mean that you will have complex methods with multiple levels of arguments or nested argument structures then you should go with SOAP. If you will have relatively simple requirements, particularly if your argument/data structure is flat then REST may well be the best option for you.

Who's using it

Unfortunately despite the main goal of SOAP being to remove any language dependencies several of the implementations in various languages have small undocumented quirks that may cause problems when connecting with SOAP clients using different languages. For example the way the PHP SoapServer class handles arrays of data is (or was last time I checked) incompatible with the way an ASP.NET SOAP client handles them. There are ways around problems such as these although they will often involve lengthy investigations (unless some kind person has already come across the problem and offers a solution).

REST on the other hand, does not have any language dependencies at all as it is simply the HTTP protocol in it's purest form. As long as you are able to handle HTTP request headers and content you will not have an issue.

So, if your API will be simple enough and you don't want the hassle of worrying about compatibility REST should do the job.

Implementation time

To use SOAP in PHP you either need to use the SoapServer class (in PHP 5) or NuSOAP (in PHP 4 or 5). You will need to write some kind of layer to talk to your SOAP server (Converting PHP data into SOAP XML and visa versa). If you are using the SoapServer class you will need to produce a Web Services Description Language (WSDL) document somehow (either by using one of the generator libraries available or by writing your own). You will also no doubt want set up a SOAP client of some form to consume your web service to ensure it is actually working the way you expected.

REST on the other hand is a lot more flexible in this regard. Your data is simpley added in the HTTP request, which is handled natively by PHP. Using REST all you really need to do is decide how you want to pass your data (e.g. As a query string, plain text, XML, JSON etc) and make sure it can be encoded/decoded on both ends.

Another option

Well, if all of this is too much and you've got a bit of time on your hands you could get the best of both worlds by implementing a version of your API for both SOAP and REST. If done correctly you can quite easily have the SOAP and REST layers using the same API core functionality without duplicating any code at all. This will give you the most flexibility and allow your API users to choose their preferred option.

SOAP nasties

As mentioned previously there is an issue with the compatibility of PHP arrays and .NET SOAP clients. PHP does not support the minOccurs or maxOccurs attribute for arrays. One possible, although not very nice solution is to define your arrays as complex types in your WSDL as follows:

<xs:complexType name=”mytypeArray”>
<xs:complexContent>
<xs:restriction base=”soapenc:Array”>
<xs:attribute ref=”soapenc:arrayType” wsdl:arrayType=”tns:mytype[]“/
</xs:restriction>
</xs:complexContent>
</xs:complexType>

You array element would then be defined as follows:

<xs:element name=”mytype” type=”tns:mytypeArray” minOccurs=”0″ maxOccurs=”1″/>

REST nasties

Assuming you are using Apache as your web server and depending on your configuration the PUT request method may be disabled by default. As the PUT method will be used heavily if you are trying to make your REST implementation truly “RESTful” this will pose a bit of a problem. If you don’t have access to edit the Apache configuration file but you may be able to solve this problem using a htaccess file and a mod_rewrite rule. Create a file named .htaccess in your document root and add the following text to it:

RewriteEngine on
RewriteCond %{REQUEST_METHOD} ^PUT$ [NC]
RewriteRule (.*)$ /api.php [L]

If you have come across any other issues with REST or the implementation of a REST API I would love to hear about them.

Conclusions

Both SOAP and REST have many good points. If you are struggling to decide which way to go hopefully this article has shed a little light on some of the

considerations that need to be made and may point you in the right direction to move forward.

This article is the first in a series on API development. In the not too distant future I will be publishing some more in depth articles on the actual implementation of a SOAP and a REST API.

If you have any comments on anything in this article or require more information relating to the use of SOAP or REST within PHP please feel free to contact me.

ShareThis

December 17th, 2008 | tags: Apache, API, PHP, REST, SOAP | Other posts by Michael Little