ARTICLE EXCERPTED FROM:

ISQ INFORMATION STANDARDS QUARTERLY

WINTER 2011 | VOL 23 | ISSUE 1 | ISSN 1041-0031

SPECIAL EDITION: YEAR IN REVIEW AND STATE OF THE STANDARDS

SUSHI IMPLEMENTATION: THE CLIENT AND SERVER EXPERIENCES

DEDICATED TO STANDARDS

EXTENDING AND PROMOTING THE USE OF OPENURL

ESTABLISHING SUGGESTED PRACTICES REGARDING SINGLE SIGN-ON

NISO How the information world CONNECTS

IP [IN PRACTICE]

SUSHI IMPLEMENTATION EXPERIENCES

CLIENT (PAGE 18)

VS.

SERVER (PAGE 20)

In 2007, NISO published the *Standardized Usage Statistics Harvesting Initiative (SUSHI) Protocol* (ANSI/NISO Z39.93), which defines an automated request and response model for the harvesting of electronic resource usage data utilizing a web services framework. It was developed to replace the time-consuming, user-mediated collection of usage data reports. COUNTER reports are the main type of usage data that is being harvested with SUSHI. Use of COUNTER with SUSHI requires that the reports be in XML format, which further enables the automation of importing this data into an electronic resource management (ERM) system.

In 2008, COUNTER issued Release 3 of the *Code of Practice for Journals and Databases*. New to the release was the requirement to support SUSHI in order to be considered compliant. This further added to the impetus of implementing SUSHI. Currently 34 publishers or aggregators are listed on the SUSHI Server Registry. The registry is one of many implementation aids that have been developed by the SUSHI Standing Committee and the SUSHI Developer community and posted to the SUSHI website.

Two implementers of SUSHI, one for the client side and one for the server side, have shared their experiences in implementing SUSHI in their organizations in the following articles. Neither developer had any previous experience in working with web services, but with the help of the SUSHI Developer community they both created successful implementations with only a few headaches along the way.

CONTINUED »

Omar
Villa

SUSHI Implementation: THE CLIENT SIDE EXPERIENCE

OMAR VILLA

I work in a Mexican company named Grupo Integra, which among other activities develops web systems for libraries. One of these systems called Kenvo Stats generates statistics on the usage of electronic resources and we added a module to this system to retrieve the COUNTER report statistics with a SUSHI client. I was in charge of developing that client. I had to learn about some new technology areas and experienced some trial and error, but ultimately developed the SUSHI client we needed.

I began to follow the SUSHI project two years earlier because we were very interested in the automation that SUSHI could provide. Then it suddenly became an urgent priority to implement a SUSHI client because some of our customers that already had our system wanted to have a SUSHI client to facilitate their work. We knew that it would also be important to attract new customers.

The first challenge I faced in my path to develop the client was that I had to learn about web services since I had never used them. So I started to read about them until I understood the basics. I then met another obstacle: the system in which the SUSHI client should be implemented is developed in PHP and in the SUSHI documentation, I only found examples of clients developed in ASP.Net and Java. I started doing tests with these clients to better understand how they work, and then I searched for PHP tools to help me make the client work in PHP. This required PHP SOAP requests and process responses, so I tried the PHP SOAP extension, a SOAP toolkit for PHP called NuSOAP, and the PEAR SOAP-Package.

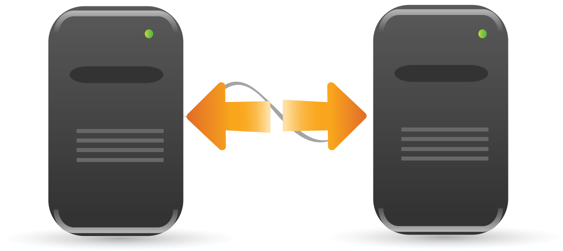
With a couple of tools I found, I succeeded in making requests to the test server of Project Euclid and got a correct response! I needed my SUSHI client to work with several suppliers, so I got information to connect to more SUSHI servers and I conducted the same testing using the same PHP tools. Unfortunately, I didn't get all positive results; there were some servers that I could not successfully communicate with using the PHP tools.

After my failure with PHP tools, I opted for developing the client in Java. I planned to then make this tool communicate to our statistical system created in PHP. Since my knowledge of Java was fairly basic, I knew this would be a challenge and probably take me a long time. Even so, I began to develop the client in Java, based on a toolkit made available by the University of Pennsylvania and posted on the SUSHI website. In working with this toolkit, I got more familiar with web services and that gave me a new idea about how to make the client work with PHP, but this time without using third party tools to make SOAP requests. I decided to create my own class in PHP and make the requests using Sockets. It didn't took me long time to figure out which headers I needed to correctly make a SUSHI request and thanks to the PHP functions, it was much easier for me to process the XML response.

Creating the client with PHP Sockets gave me greater flexibility to deal properly with the differences between SUSHI servers, as there are some that require authentication to send special headers. With the changes I implemented, my SUSHI client became fully interoperable.

When it came time to process the server responses, I realized that the XML responses had some variations, especially in the ReportItems node. Some servers send a ReportItems node for each ItemPerformance node; others put together multiple ItemPerformance nodes on a single ReportItems node. In some cases when the Count node value was zero, the ReportItems node was ignored, but in the same case with other servers, the node was included. Some servers shipped multiple Customer nodes in the same response; that is useful as it serves to collect statistics independently of each area of the institution to which statistics are retrieved. The variations I encountered complicated the processing of the responses and is something other implementers should note if getting data from different SUSHI servers.

Since I started tackling the SUSHI project I had many questions and sometimes I asked for help through the SUSHI Develepers list, and this enabled me to better understand several things. For example, during the development process I was uncertain how to make a request to the ProQuest server and I returned to seek help through the list, where I was told the key to making the request. The problem was that I needed to send some additional headers for authentication.



Currently our SUSHI client is successfully retrieving the COUNTER reports JR1, DB1, and DB3 from EBSCOhost, ProQuest, ACS Publications, and ISI Web of Knowledge servers. Having a module of a SUSHI Client has helped us to make our system more attractive for new customers

Currently our SUSHI client is successfully retrieving the COUNTER reports JR1, DB1, and DB3 from EBSCOhost, ProQuest, ACS Publications, and ISI Web of Knowledge servers. Having a module of a SUSHI Client has helped us to make our system more attractive for new customers and has saved much time and effort for them to obtain their statistics. Soon we will extend our support to add more providers and additional COUNTER reports.

I want to thank all those who have made the SUSHI project possible and especially to Oliver Pesch who helped me so much. | IP | doi: 10.3789/isqv23n1.2011.04

OMAR VILLA ACOSTA <ovilla@gpo-integra.com> is IT Development Manager at Grupo Integra <www.gpo-integra.com> in Mexico City, Mexico.

SERVER SIDE EXPERIENCE »