**Scopus CitedBy**

**EDIT:**

These scripts have been updated to point to the new webserver v10 as the v7 is due to retire summer 2012.

Joe Schulkins 11/01/12

**Evaluation:**

When I first began looking at writing the command line script I didn’t anticipate that it would be such, I’d hoped to create a screen plugin which would load and render the information as a record was accessed. However, I’d grossly underestimated the amount of time that it would require and complexity that would be involved, not least because my knowledge surrounding eprints, perl and the web service involved was fairly limited so I probably wasn’t best placed to assess the possible pitfalls. Although I had been using eprints for approximately thirteen months, my skill level and familiarity with the repository was fairly limited due to my ad hoc involvement with it. I’d also had no previous perl experience (other than making changes as suggested by others) and had no experience with web services (regardless of language). Writing this script has helped me to get a better understanding of how the repository fits together and has helped in my appreciation of how various elements of the repository interact, I’ve also been able to get a better handle on perl and what elements of the perl scripts are/do. With increased activity planned for the repository, both of these are going to be extremely helpful in the future.

Whilst it is a great benefit that my understanding of both perl and eprints has increased during this time, it wasn’t completely unexpected that it would do so. However, a further benefit came on the back of a few operational wrinkles. As previously whenever we modified an existing file there was no way of knowing what part of the file was changed, or when, or by whom and as such a colleague has recommended a new way of working which is now in place whereby changes/additions to existing files are wrapped in comments to show that they are a local change/addition, the date of the change/addition, the author and the reason for the change/addition. This will help both in troubleshooting and during periods of upgrades when we look at merging back local changes/additions to the new files.

Even though this script does what it is designed to do, it is not what I’d had in mind when I began planning it and as such there are a few areas I’d like to look at in the future. Most notably this would include revisiting the functionality of the script so that it behaves like the screen plugin I’d envisaged it to be (more than likely this would involve the use of AJAX).

**Script Instructions:**

Before proceeding with the development of your own Scopus command line script, you will need to contact Scopus to arrange the proper user credentials and be given access to the web service, they can be reached at [integrationsupport@elsevier.com](mailto:integrationsupport@elsevier.com). Among other things you will need to provide them with the ip of the server you will be working from, but depending on your local setup it might also be a good idea to include the ip or subnet of your own machine. Elsevier will then supply you with the server url for the service, a client ID for the requests, a partnerID for url generation and a salt key for md5 generation.

My script is written using the SOAP::Lite module, primarily because I found this far easier to work with than SOAP::WSDL. Elsevier do provide you with a wsdl and the associated bindings and as such you could use the SOAP::WSDL module, but as a beginner I found this perl module more complicated to use (updating the script so it uses SOAP::WSDL is definitely something I have added for the list for if/when I re-visit what I have done).

In addition to the SOAP::Lite module you will also need to have Digest::MD5 as this is referenced within eprint\_render.pl for the generation of an MD5 which is required for off campus access.

The script below uses one doi for illustrative purposes and the request and response message that follow are for this doi. Note: you can get this detailed output of the requests and responses by entering the debug mode of SOAP. To do this add the following to the ‘use SOAP::Lite’ statement: +trace =>'debug'

soap\_live.pl.

use lib '/eprints/eprints3/';

use EPrints;

use SOAP::Lite;

use strict;

my $session = new EPrints::Session( 1, "[*archiveid*]" );

exit( 0 ) unless( defined $session );

my $ds = $session->get\_repository->get\_dataset( "archive" );

my $search = new EPrints::Search( session=>$session, dataset=>$ds );

$search->add\_field( $ds->get\_field( "type" ), "article" );

my $list = $search->perform\_search;

$list->map( \&process\_eprint );

sub process\_eprint

{

my( $session, $ds, $eprint ) = @\_;

return unless $eprint->is\_set( "doi" );

# below is the doi code for the live version of the script

# my $doi = $eprint->get\_value( "doi" );

# for the purposes of a test the doi below is used

my $doi = "10.1371/journal.pmed.0020336";

# my $client = $eprint->get\_value( "doi");

# for the purposes of a test the doi below is used

my $client = "10.1371/journal.pmed.0020336";

# the regex below strips all characters bar numbers from the doi

$client=~s/\D//g;

# the stripped doi from above is randomised to provide a unique(ish) number

my $crf = int(rand($client));

print "Querying scopus for $doi ...";

my $body = SOAP::Data->name(getCitedByCountReqPayload => \SOAP::Data->value(

SOAP::Data->name(dataResponseStyle => "MESSAGE")->type(''),

SOAP::Data->name(absMetSource => "all")->type(''),

SOAP::Data->name(responseStyle => "wellDefined")->type(''),

SOAP::Data->name(inputKey => \SOAP::Data->value(

SOAP::Data->name(doi => "$doi")->uri('')->prefix('')->type(''),

SOAP::Data->name(clientCRF => "$crf")->uri('')->prefix('')->type(''),

))));

my $header = SOAP::Header->name(EASIReq => \SOAP::Header->value(

SOAP::Header->name(TransId => " ")->uri('')->type(''),

SOAP::Header->name(ReqId => " ")->uri('')->type(''),

SOAP::Header->name(Ver => " ")->uri('')->type(''),

SOAP::Header->name(Consumer => [Consumer ID])->uri('')->type(''),

SOAP::Header->name(ConsumerClient => " ")->uri('')->type(''),

SOAP::Header->name(OpaqueInfo => " ")->uri('')->type(''),

SOAP::Header->name(LogLevel => "Default")))

->uri('http://webservices.elsevier.com/schemas/easi/headers/types/v1')->prefix('');

my $soap = SOAP::Lite->proxy(http://services.elsevier.com/EWSXAbstractsMetadataWebSvc/XAbstractsMetadataServiceV10?wsdl')

->uri(' http://webservices.elsevier.com/schemas/metadata/abstracts/types/v10');

my $som = $soap->getCitedByCount($header,$body);

my $n = $som->match('//citedByCountList/citedByCount/linkData/citedByCount')->valueof;

$eprint->set\_value( "scopus\_citation\_count", $n );

$eprint->commit;

my $id = $som->match('//citedByCountList/citedByCount/linkData/scopusID')->valueof;

$eprint->set\_value( "scopus\_id", $id );

$eprint->commit;

}

$list->dispose();

$session->terminate();

Once the script has been written you will need to create two fields in your database to hold the values of “scopus\_citation\_count” and “scopus\_id”. This is achieved by altering eprint\_fields.pl to incorporate these two new fields:

{ 'name' => 'scopus\_citation\_count', 'type' => 'int', 'volatile' => 1, },

{ 'name' => 'scopus\_id', 'type' => 'int', 'volatile' => 1, },

Followed by:

bin/epadmin update\_database\_structure [*archiveID*]

This instruction will commit these fields to your database.

The next step is to add a few lines to eprint\_render.pl to control the display and what you put here will depend upon your own display preference. Within the display information a call to Digest::MD5 is made to generate the MD5 value which is needed for off campus access, below is the entry for our eprint\_render.pl:

### LIVERPOOL (js) - 05 May 2009 - scopus rendering

if( $eprint->is\_set( "scopus\_citation\_count" ) )

{

my $count = $eprint->get\_value( "scopus\_citation\_count" );

my $scopus\_id = $eprint->get\_value( "scopus\_id" );

my $citedby\_url = "http://www.scopus.com/scopus/inward/citedby.url";

my $args = "scp=$scopus\_id&partnerID=VE8K82pP&rel=6.0";

my $salt = "m5.QVzxS12ahKK+0+0pFKNjNfgq!mU6i";

my $md5 = new Digest::MD5;

$md5->add( "$args", "$salt" );

my $digest = $md5->hexdigest;

my $oncampus\_url = $citedby\_url."?".$args;

my $offcampus\_url = $oncampus\_url."&md5=".$digest;

my $div = $session->make\_element( "div", style=>"text-align: right" );

$page->appendChild( $div );

$p = $session->make\_element( "p", style=>"margin-bottom: 5px" );

$div->appendChild( $p );

my $cite = $session->make\_text( "Cited $count times in ");

$p->appendChild( $cite );

my $img = $session->render\_link( "$offcampus\_url" ); $img->appendChild( $session->make\_element

( "img", src=>"/images/liv/scopus.gif", height=>"10px", width=>"80px", alt=>"Scopus Logo", border=>"0" ) );

$p->appendChild( $img );

#$p->appendChild( $session->make\_element( "br" ) );

}

### LIVERPOOL (js) - 05 May 2009

The above rendering uses the scopus logo (stored as a gif in the images directory) as the link to the citation information page in Scopus.

Running the script using our example doi and with the debug option turned on (see above for how to implement this) brings back the following:

#This is the packaged request to the server

SOAP::Transport::HTTP::Client::send\_receive: POST [*server url which ends with ?wsdl*]HTTP/1.1

Accept: text/xml

Accept: multipart/\*

Accept: application/soap

Content-Length: 1121

Content-Type: text/xml; charset=utf-8

SOAPAction: "[*the abstracts namespace*]#getCitedByCount"

<?xml version="1.0" encoding="UTF-8"?><soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"><soap:Header><EASIReq xmlns="[*headers namespace*]"><TransId xmlns=""> </TransId><ReqId xmlns=""> </ReqId><Ver xmlns=""> </Ver><Consumer xmlns="">ULRA</Consumer><ConsumerClient xmlns=""> </ConsumerClient><OpaqueInfo xmlns=""> </OpaqueInfo><LogLevel xsi:type="xsd:string">Default</LogLevel></EASIReq></soap:Header><soap:Body><getCitedByCount xmlns="[*abstracts namespace*]"><getCitedByCountReqPayload><dataResponseStyle>MESSAGE</dataResponseStyle><absMetSource>all</absMetSource><responseStyle>wellDefined</responseStyle><inputKey><doi xmlns="">10.1371/journal.pmed.0020336</doi><clientCRF xmlns="">1.37044353218754e+15</clientCRF></inputKey></getCitedByCountReqPayload></getCitedByCount></soap:Body></soap:Envelope>

#this is the server response

SOAP::Transport::HTTP::Client::send\_receive: HTTP/1.1 200 OK

Date: Fri, 20 Feb 2009 15:38:29 GMT

Server: cdc.elsevier.com 315.10

Content-Language: en-US

Content-Length: 1369

Content-Type: multipart/related; boundary=MIMEBoundaryurn\_uuid\_AAF4B79A5E22BE1BFF1235144368212; type="text/xml"; start="<0.urn:uuid:AAF4B79A5E22BE1BFF1235144368213@apache.org>"

Client-Date: Fri, 20 Feb 2009 15:55:30 GMT

Client-Peer: 207.25.181.224:80

Client-Response-Num: 1

P3P: CP="IDC DSP LAW ADM DEV TAI PSA PSD IVA IVD CON HIS TEL OUR DEL SAM OTR IND OTC"

X-Cnection: close

X-RE-Ref: 1 1909901168

--MIMEBoundaryurn\_uuid\_AAF4B79A5E22BE1BFF1235144368212

content-type: text/xml; charset=utf-8

content-transfer-encoding: 8bit

content-id: <0.urn:uuid:AAF4B79A5E22BE1BFF1235144368213@apache.org>

<?xml version="1.0" encoding="utf-8"?><soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"><soapenv:Header><Q1:EASIResp xmlns:Q1="http://webservices.elsevier.com/schemas/easi/headers/types/v1"><RespId>8a2ce0fd-54809363-11f5c2fecf9--3079 </RespId><ServerId>[*Server ID*]</ServerId></Q1:EASIResp></soapenv:Header><soapenv:Body><ns2:getCitedByCountResponse xmlns:ns2="http://webservices.elsevier.com/schemas/metadata/abstracts/types/v7" xmlns:ns3="http://webservices.elsevier.com/schemas/easi/headers/types/v1"><ns2:status><statusCode>OK</statusCode></ns2:status><ns2:getCitedByCountRspPayload><ns2:citedByCountList><ns2:citedByCount><ns2:inputKey><doi>10.1371/journal.pmed.0020336</doi><clientCRF>1.37044353218754e+15</clientCRF></ns2:inputKey><ns2:linkData><ns2:eid>2-s2.0-33847339353</ns2:eid><ns2:scopusID>33847339353</ns2:scopusID><ns2:citedByCount>51</ns2:citedByCount></ns2:linkData></ns2:citedByCount></ns2:citedByCountList><ns2:dataResponseStyle>MESSAGE</ns2:dataResponseStyle></ns2:getCitedByCountRspPayload></ns2:getCitedByCountResponse></soapenv:Body></soapenv:Envelope>

--MIMEBoundaryurn\_uuid\_AAF4B79A5E22BE1BFF1235144368212--

After the script has retrieved the results for all Eprints, either a restart of Apache or reload of the repository configuration are needed before the abstracts reflect the additional fields. Once this has been done all Eprints should then display the Scopus citation count as illustrated with the Scopus logo acting as a link for the constructed url to the citation page for the document:



**Appendix**

A quick command line debug script which takes a doi as a user input and performs an individual query on scopus outputting the sent crf, the returned crf, the eid, scopus ID and citation count. Adding “ +trace=> 'debug' “ to ‘use SOAP::Lite’ will give the full request and response messages. Useful for debugging or if you want to see the results for one particular item.

soap\_debug.pl command line testing script for individual doi’s

#!/usr/bin/perl -w -I/eprints/eprints3/perl\_lib

use lib '/eprints/eprints3/';

use EPrints;

use SOAP::Lite;

use strict;

print "Enter the doi to search for: ";

#Enter DOI(s). For multiple searches separate them with a space

my @doi = split(/\s+/, <>);

foreach (@doi ){

my $client = $doi;

$client=~s/\D//g;

my $crf = int(rand($client));

my $body = SOAP::Data->name(getCitedByCountReqPayload => \SOAP::Data->value (

SOAP::Data->name(dataResponseStyle => "MESSAGE")->type(''),

SOAP::Data->name(absMetSource => "all")->type(''),

SOAP::Data->name(responseStyle => "wellDefined")->type(''),

SOAP::Data->name(inputKey => \SOAP::Data->value(

SOAP::Data->name(doi => "$doi")->uri('')->prefix('')->type(''),

SOAP::Data->name(clientCRF => "$crf")->uri('')->prefix('')->type (''),

))));

my $header = SOAP::Header->name(EASIReq => \SOAP::Header->value(

SOAP::Header->name(TransId => " ")->uri('')->type(''),

SOAP::Header->name(ReqId => " ")->uri('')->type(''),

SOAP::Header->name(Ver => " ")->uri('')->type(''),

SOAP::Header->name(Consumer => "[Consumer ID]")->uri('')->type(''),

SOAP::Header->name(ConsumerClient => " ")->uri('')->type(''),

SOAP::Header->name(OpaqueInfo => " ")->uri('')->type(''),

SOAP::Header->name(LogLevel => "Default")))

->uri('http://webservices.elsevier.com/schemas/easi/headers/types/v1')->prefix('');

#Query the test environment

#my $soap = SOAP::Lite->proxy('http://cdc315-services.elsevier.com/EWSXAbstractsMetadataWebSvc/XAbstractsMetadataServiceV10?wsdl')

#->uri('http://webservices.elsevier.com/schemas/metadata/abstracts/types/v10');

#Query the Production environment

my $soap = SOAP::Lite->proxy('http://services.elsevier.com/EWSXAbstractsMetadataWebSvc/XAbstractsMetadataServiceV10?wsdl')

->uri('http://webservices.elsevier.com/schemas/metadata/abstracts/types/v10');

my $som = $soap->getCitedByCount($header,$body);

my $n = $som->match ('//citedByCountList/citedByCount/linkData/citedByCount')->valueof;

my $id = $som->match('//citedByCountList/citedByCount/linkData/scopusID') ->valueof;

my $ret\_crf = $som->match ('//citedByCountList/citedByCount/inputkey/clientCRF')->valueof;

my $eid = $som->match('//citedByCountList/citedByCount/linkData/eid')-> valueof;

print "Submitted CRF = $crf\n Returned CRF = $ret\_crf\n EID = $eid\n Scopus ID = $id\n Citation Count = $n\n";

}