Scientific web interfaces using PHP and EPICS

A. Bertrand, R. Krempaska, PSI

Introduction

Most of the scientific instruments at the Swiss Light Source (SLS) at PSI are controlled using the EPICS control system, which offers many solutions to easy implement applications, but they are platform dependent. For developing users' services, the platform independence and remote access are starting to be mandatory. To fulfil those needs, we have developed an interface between the EPICS channel access protocol and the PHP Web scripting language. This PHP module is now used to develop data-collection applications at the Protein Crystallography beamline.

SLS Control System

The Swiss Light Source (SLS) is a high brightness 3rd generation synchrotron light source at the Paul Scherrer Institute. It consists of a Linac, a booster synchrotron, a storage ring and experimental beamlines. The SLS machine and beamline control system[1] based on EPICS has a two-tier structure with VME crates containing PPC processors running vxWorks and with PCs running Linux as consoles. The Experimental Physics and Industrial Control System (EPICS) has been widely adopted by large accelerator community. It is basically composed of servers called Input Output Controllers (IOCs), a protocol called Channel Access (CA) and of tools which allow to design graphical user interfaces, to archive values, plot graphs, handle alarms, etc. Each server or IOC controller generally controls variables, for example the position of a motor, state of the shutter, etc. These variables, in the EPICS world, are called Process Variables (PVs). Channel access is a protocol used to establish connections to PVs. Once a connection has been established, it is possible to read or change PVs values, or monitor their values.

Access to the control system via the web

One of the requirements for the control system expressed especially by beamline scientists and users is that as much information as possible should be presented via the web. This comprises not only live control system displays but also the possibility to monitor and control an experiment via the web with appropriate security. As channel access has been adopted as a standard communication protocol the solution to integrate this protocol directly from a web application was needed. For web applications development at SLS we adopted a PHP[2], an HTML-embedded scripting language PHP which allows web developers to write dynamically generated pages quickly. In order to integrate channel access we created a custom PHP extension module using C programming language.

Implementation of the module

We used C and EPICS Channel Access API[3] to access control system process variables or PVs, which are any piece of data related to the system. The functions which access data from the control system are callable from a PHP program. We implemented functions by using which a user can read a value of EPICS PV, write to it, or establish a monitor on it.

Access EPICS from PHP

The code example shows how to read or set an EPICS value. The ca_get and ca_put functions implemented in php_epics module are used from the script as they would

be part of PHP language. Before the first call of the EPICS function, a load of the module is needed.

```
<?php
dl(“php_epics.so”);

//read a value from EPICS
$value=ca_get(“PV_NAME”);
//set a value
$val=2.4;
ca_put(“PV_NAME”,$val);
?>
```

Img.1: Usage of ca_get and ca_put from a PHP code.

Another code example shows how to monitor an EPICS value from a PHP page.

```
<?php
dl(“php_epics.so”);
function my_monitor($a,$b) {
   flush();
}
ob_end_flush() ;

$res=ca_monitor("PV_NAME1,PV_NAME2",40,"my_monitor");
echo "$res... Done.";
?>
```

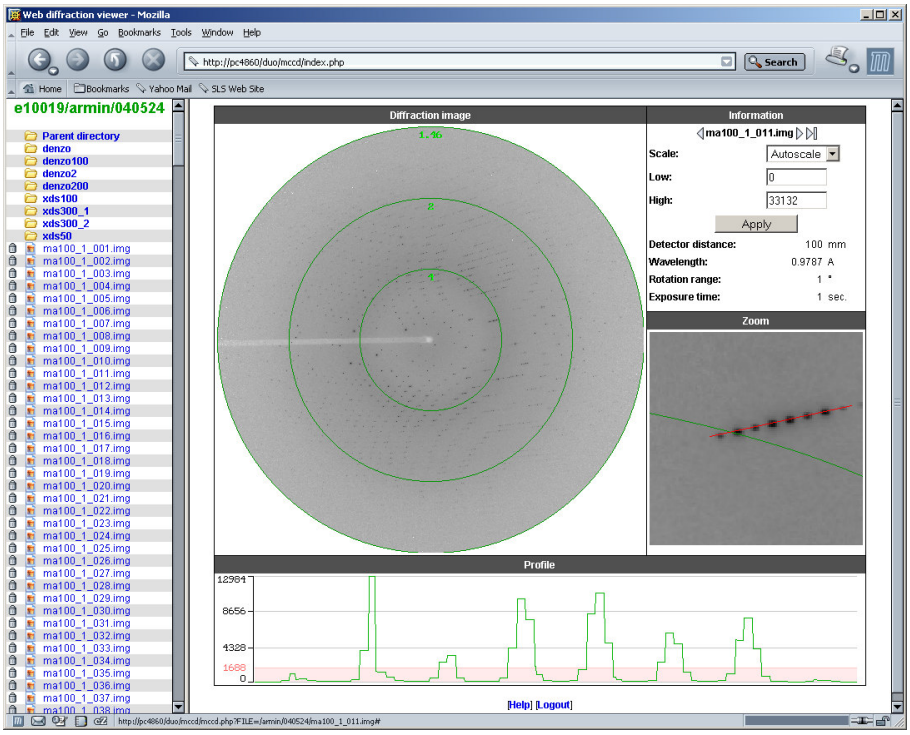
Img.2: Usage of ca_monitor from a PHP code.

Trees To Speed Up

The php-epics module is implemented in such a way that when a client does an access to an EPICS PV for the first time, a connection is established and a channel identifier or chid is created. We are storing the chid in a structure so that when the next access to this PV is done, the already created chid is used. First we used a chained linked list or chain. However, with increasing number of channels, the performance was slow, which resulted to long php page load time. That’s why we started to implement a tree structure which should speed up the search. First we used a binary tree, a structure where each node has at most two leaves - right and left. This is working for so called true binary trees. If, however the elements are in the alphabetical order, we can end up with a degenerated tree which has one branch very long. That’s why we implemented another solution - a quaternary tree having four branches at each node. The search through quaternary tree and through linked chain list is factor three faster.

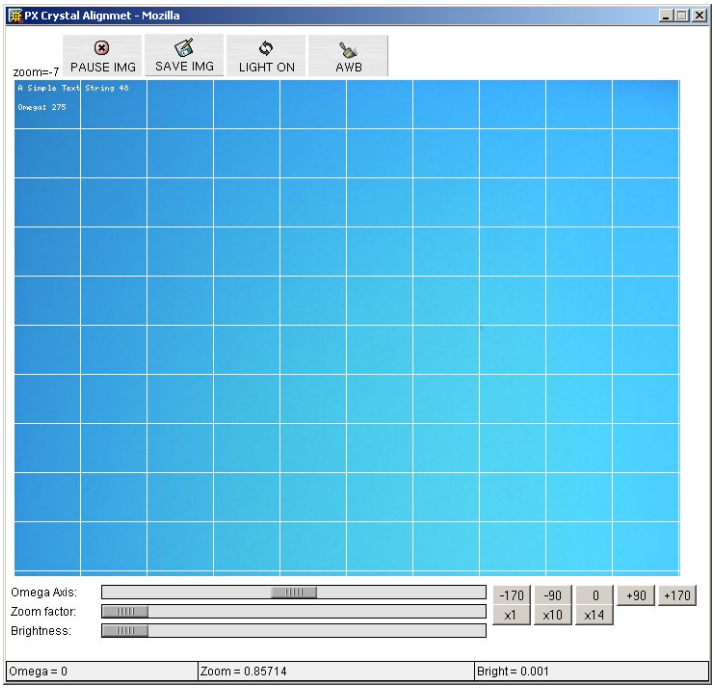
Using the web browser for science:

Actual web browsers allows, with some works, to create complex interfaces which can be used for experiment handling, or displaying data. By mixing C, PHP and JavaScript we are able to create web interface like a diffraction viewer, where the HTML and JavaScript is generated on the fly by a PHP script, and the images and plot are C functions which extends PHP:



Img.3: Diffraction viewer

We presented in the talk the crystal alignment tool which allows to view the position of the loop and a center it in the middle of the beam. Again this tool mix C for the epics channel access, PHP which generate the HML and call the PHP_EPICS module and JavaScript in order to create those sliders and buttons found on the bottom of the page.



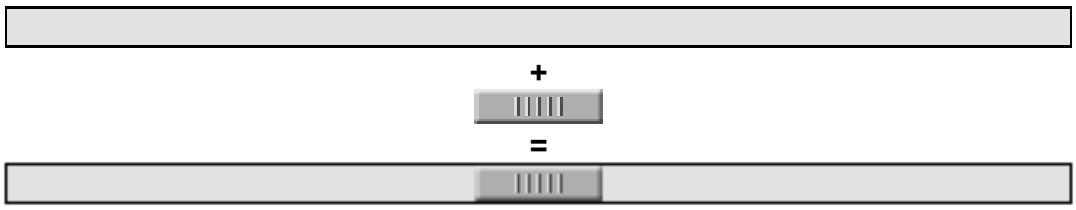
Img.4: Crystal alignment

As stated before, our control system is based on EPICS, there fore accessing or changing specific values to instruments is done using the channel access protocol.

The first prototype we created for the crystal alignment interface was using XML-RPC and then an XML-RPC/Channel access bridge. Even if this solution worked, we discovered in no time that it was not practicable for long-term operations as it was introducing just another component in the middle. So we started to investigate for the possibility of extending PHP in order to directly access channel access. This didn't took us too long to have a first version running. Now this web interface doesn't use anymore XML-RPC but instead directly connect via channel access, via the homemade PHP-EPICS module written in c.

PHP is used to receive the user requests, pass them to channel access, and generate the HTML. The image with a grid seen in the middle of the page is actually a JPEG stream created within PHP which on one side request the actual microscope image, and on the other side, using the server push technology, send the image which on the browser side is converted to a movie.

Finally, as HTML do not directly contain any tag which would allows to create sliders, a small JavaScript function control the mouse and two different images to create the appearance of the sliders:



References:

1. S. Hunt et al. Status of the SLS Control System, 8th International Conference on Accelerator & Large Experimental Physics Control Systems, 2001, San Jose, California
2. http://www.php.net
3. J.O.Hill, EPICS Channel Access Reference Manual