# SCIENTIFIC WEB INTERFACES USING PHP AND EPICS

*A. Bertrand, R. Krempaská*

*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 easily implement applications, but they are platform dependent. For developing users' services, the platform independence and remote access are starting to be mandatory. To fulfill 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.*

## 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[1], 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 the C programming language.

## TECHNOLOGY

We used C and the EPICS Channel Access API[2] 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 that allow a user to read a value of an EPICS PV, write to it, or to monitor it.

## FUNCTIONS IMPLEMENTED

Once the PHP-Epics module is loaded inside PHP, 5 new functions are available: ca_get, ca_put, ca_state, ca_info and finally ca_monitor. These cover all the needs for a full EPICS communication.

## 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 in long php page loading 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 works for so called true binary trees. If, however the elements are in the alphabetical order, we can end up with a degenerate tree which has one branch that is very long. That’s why we implemented another solution - a quaternary tree having four branches at each node. The search through a quaternary tree is a factor of three faster than the one through linked chain list.

## USING THE WEB BROWSER FOR SCIENCE

Actual web browsers allow, with some work, 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 interfaces like a diffraction viewer, where the HTML and JavaScript is generated on the fly by a PHP script, and the images and plots are C functions which extend PHP.

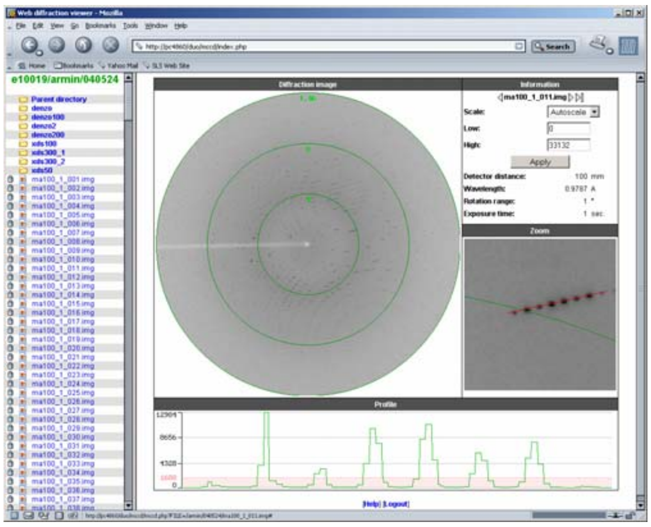

**Fig. 1:** Diffraction viewer.

## REFERENCES

[1] The PHP project website: http://www.php.net.

[2] J.O. Hill, EPICS Channel Access Reference Manual.