JOURNAL OF SYNCHROTRON RADIATION

ISSN 1600-5775

WIFIP: a web-based user interface for automated synchrotron beamlines

Yoann Sallaz-Damaz* and Jean-Luc Ferrer*

Institut de Biologie Structurale (IBS), Université Grenoble Alpes, CEA, CNRS, 38044 Grenoble, France.
*Correspondence e-mail: yoann.sallaz-damaz@ibs.fr, jean-luc.ferrer@ibs.fr

Received 10 December 2016
Accepted 18 June 2017

Edited by M. Yamamoto, RIKEN SPring-8 Center, Japan

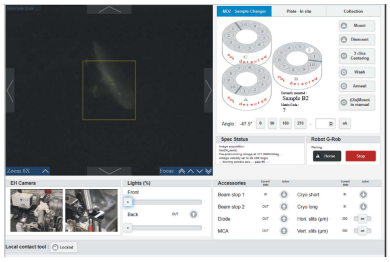
Keywords: graphical user interface; web-based; macromolecular crystallography; automated beamline.

The beamline control software, through the associated graphical user interface (GUI), is the user access point to the experiment, interacting with synchrotron beamline components and providing automated routines. FIP, the French beamline for the Investigation of Proteins, is a highly automatized macromolecular crystallography (MX) beamline at the European Synchrotron Radiation Facility. On such a beamline, a significant number of users choose to control their experiment remotely. This is often performed with a limited bandwidth and from a large choice of computers and operating systems. Furthermore, this has to be possible in a rapidly evolving experimental environment, where new developments have to be easily integrated. To face these challenges, a light, platform-independent, control software and associated GUI are required. Here, *WIFIP*, a web-based user interface developed at FIP, is described. Further than being the present FIP control interface, *WIFIP* is also a proof of concept for future MX control software.

1. Introduction

The beamline control software, through the associated graphical user interface (GUI), is the user access point to the experiment, interacting with synchrotron beamline components and providing automated routines. The design of the GUI is a key part of the beamline. Significant progress over the last decades has seen the evolution from command-line interface to graphical interface. In the field of macromolecular crystallography (MX), the most commonly used control interface in Europe is the Python/Qt-based interface *MXCuBE* (http://mxcube.github.io/mxcube/) (version 2) (Gabadinho *et al.*, 2010). In the last few years, the evolutionary trend for user interfaces is the replacement of the traditional GUI with web-based user interfaces (WUIs), as observed with many software (word processor, picture management, network attached storage). The WUI has the huge advantage that it can be used from anywhere without prior installation of specific software. In the case of beamline control, it offers the possibility to operate the experiment from any computer inside or outside of the synchrotron site.

FIP (the French beamline for the Investigation of Proteins) (Roth *et al.*, 2002) is a highly automatized MX beamline at the European Synchrotron Radiation Facility (ESRF). A significant proportion of FIP users choose to access the beamline remotely. Until recently, they were using a display sharing with *VNC* software through SSH (secure shell) or remote desktop connection with *NX* (*Nomachine*; https://www.nomachine.com/). However, all those solutions are very bandwidth-consuming because of their principle of sending the full graphical interface to the end user, with therefore reduced



performances. As an illustration, the lag time for a remote MX experiment following the standard ESRF method (ESRF, 2017) was estimated, using *NX* client version 4 and *MXCuBE2*. These tests were carried out with several Internet connections [Eduroam (GÉANT, 2017) WIFI, ADSL 12 Mb/1 Mb and ADSL 4 Mb/256 Kb] and computer systems (Linux with *NX4* hard client and Windows *NX4* light client) combinations. The observed lag between the action request and the observed effect on the sample video was in the range 350–850 ms. Furthermore, *NX* includes a compression algorithm that adapts to available bandwidth to balance graphical quality and response time. But with slow connection, the compression rate becomes relatively aggressive and deteriorates the video quality with typical artifacts like macro-blocks, reducing some visible details.

To fix these issues, a web-based user interface was developed on FIP: *WIFIP* (*Web Interface for FIP*; Fig. 1). With the release of *WIFIP* in September 2015, FIP was the first beamline offering the full control of a MX experiment from anywhere based on a web interface and this was only the beginning of a general trend, with different projects being developed at several synchrotrons, like *YAIBEX* at the Australian Synchrotron (Jong *et al.*, 2015) and the collaboration project *MXCuBE3* (http://mxcube.github.io/mxcube/). Other web-based application also exist in the USA, such as *Web-Ice*, developed at SSRL (González *et al.*, 2008). To the best of our knowledge the *Web-Ice* WUI is designed to run with *Blu-Ice* (McPhillips *et al.*, 2002) that uses a traditional GUI based on Tcl/Tk, for the actual control of the experimental setup. In comparison with *WIFIP*, the objectives of *Web-Ice* are not exactly the same. Triggered by *Blu-Ice*, *Web-Ice* is mainly dedicated to the diffraction analysis and does not manage sample handling and beamline equipment control. It comes closer to what the integration of *ISPyB* (information system for protein crystallography beamlines) in *WIFIP*, presently in progress, will be: to answer the specific problem of data analysis, auto-processing, strategy calculation and results display in a web interface.

The general architecture of *WIFIP* and the implementation of its main features will be described first, followed by a discussion about its performances.

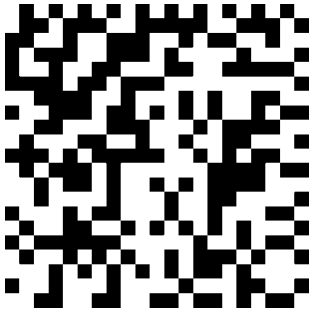


Figure 1
WIFIP video demo: https://youtu.be/v2zOM0C8-Qw. This video was recorded outside ESRF from a personal computer connected to the Internet with an ADSL 12 Mbps data connection.

2. Technical implementation

2.1. Language and development tools

The concept behind a web-based user interface like *WIFIP* is to create a link between a web browser and the beamline control software, in our case with TANGO (Chaize *et al.*, 1999; Taurel *et al.*, 2005) and SPEC[1]. This middle-ware should be able to:

(i) Be a web server itself or link to an external one like Apache (https://httpd.apache.org/).

(ii) Translate a browser request into a beamline control software command.

(iii) Translate an event from the beamline control software or command response into information suitable for a web page.

The *WIFIP* server is written in Python, which has the advantage of providing libraries for TANGO and SPEC, but also for Modbus (Modbus, 2004), the communication protocols used by some specific FIP equipment, such as the sample changer [a six-axis robot named G-Rob (Jacquamet *et al.*, 2004*a*)], and the Programmable Logic Controllers (PLCs). In addition, with Python language, many web frameworks such as *Django* (https://www.djangoproject.com) and *Flask* (Ronacher, 2010) are available for the development of the web server part. *WIFIP* was written using Bottle (Hellkamp, 2009), a simple and light web framework. The web page itself is written in HTML5 (W3C, 2014). JavaScript is based on the JQuery library (https://jquery.com), facilitating the design and the handling of Document Object Model (DOM) (Chakrabarti, 2001). The X-ray fluorescence spectrum plot is produced using the JavaScript library DyGraphs (http://dygraphs.com/). HTML5 and JQuery provide a pleasant display on all major web browsers, accounting for differences in the standard web implementations in those browsers. Furthermore, JQuery simplifies the access to the AJAX (Asynchronous JavaScript and XML) (Lawton, 2008) method used to send data and retrieve it from a server asynchronously. In practice, this method makes it possible to send information from the web page to the Python web server without leaving the current web page. When the result is retrieved, an event triggers off the appropriate JavaScript function to process the incoming data.

2.2. Communication

Fig. 2 shows a summary of the interconnections between key elements. One can distinguish two cases. The first one is an update of the web page according to the beamline status. The second case is an action from the web page (a button pushed for example) to the beamline control. Upon beamline status update, the Python script uses mainly event call back (for example, a call back after a motor has moved) from TANGO and SPEC libraries, and a loop for others, to fill a JSON object (Crockford, 2006). The web page uses a loop to read this object each second *via* Bottle, and then parses it. This information is treated by the JavaScript in the web page to even-

[1] SPEC and *C-PLOT* are trademarks of certified scientific software (https://www.certif.com).

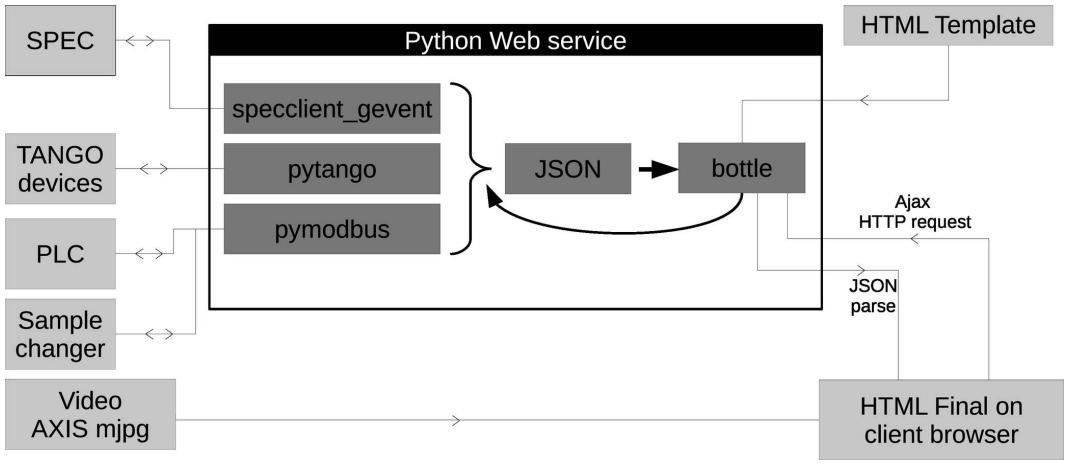


Figure 2
Connection diagram between beamline software/hardware and *WIFIP* components: the Python web service is in the middle, the beamline control and hardware parts are represented on the left and the web part on the right.

tually update itself, through a DOM update, without reloading the web page. For the second case, when a user makes an action on the web page, the request is sent by HTTP POST with AJAX. Using Bottle, the Python script creates a URL to obtain information from this request and executes a Python function to communicate with the appropriate part of the beamline software and/or hardware. As an example, on the *WIFIP* web page, one can find a slider to select the sample lighting intensity. When a user moves the slider to 50%, a JavaScript function, linked to this event, is executed. In this function an HTTP POST request is sent to the 'WIFIP URL' / frontlight/ with 'value = 50' as input. In the Python script, the request on this URL is linked by Bottle to a function able to read the 'value' variable and send the appropriate Modbus command to the hardware (here a PLC).

Another critical aspect on this project is to provide a sample view. On the FIP beamline, sample viewing is performed by an analogical on-axis video camera installed on a mini-diffractometer MD2 (Maatel/Arinax, France). In addition, several other analogical cameras ensure the experiment surveillance. Q7401 and M7014 encoders (AXIS, Sweden) produce a video flux as MJPEG (Kamath & Jackson, 2004) streams directly displayed on the *WIFIP* web page.

To use *WIFIP* from outside the ESRF through the firewall, a 'direct' connection through the SSL Gateway of the ESRF was tested. It has the advantage of working fully in a browser without any other software or manipulation. However, the availability of this solution to all FIP users is still under discussion, and will depend on the future site policy regarding remote access. In the meantime, communication through the firewall is ensured by a SSH tunneling with port forwarding: users initiate a SSH tunneling with their ESRF login and navigate to 'http://localhost:9000' with their own browser. Linux or Mac OS users can use a single command line on their terminal to establish the SSH connection. For Windows users, a standalone, self extracted and scripted executable was built to automatically create this connection. In this latter case, the SSH client is based on *PuTTY* (Tatham, 1997).

3. User interface and features

3.1. Overview

The concept of *WIFIP* is to offer, in one page, all required actions for users to control their experiment, such as:

(i) Displaying and centering a sample.

(ii) Mounting/dismounting a frozen sample.

(iii) Mounting/dismounting samples *in situ* (Jacquamet *et al.*, 2004*b*).

(iv) Setting up and running data collection.

(v) Displaying the diffraction image.

(vi) Changing beamline energy and making use of a multi-channel analyzer (MCA).

On the top of the web page, a tab selector allows choosing between the steps of an experiment. The two leftmost tabs are dedicated to the handling of samples of the two different types: frozen samples mounted on loops or room-temperature samples *in situ* in crystallization plates or microchips. The third tab is used for the MCA and beam energy management. The last tab is the interface for data collection.

3.2. Sample handling

The two tabs for the sample handling give access to the sample transfer from storage location to beam position, performed by the G-Rob system (Jacquamet *et al.*, 2004*a*), and to the sample centering. In addition, in the 'frozen sample' mode extra features are available such as sample washing and annealing, that are also performed by the G-Rob system.

For each of these two tabs, the interface is split into two parts (see Figs. 3 and 4). The left-hand side is dedicated to viewing, sample lighting and centering. The right-hand side is dedicated to sample transfer (top right) and control of components of the sample environment (bottom right) such as slits for beam definition, monitoring diode, beam stop, *etc*.

Finally, at the very bottom, a status bar displays emergency alarms, such as cryogenic stream malfunction. It also displays a link selector with advanced tools (goniometer calibration,

beamline health monitor, manual beam alignment) with restricted access to beamline staff.

The left-hand side displays the on-axis camera stream (720 × 576 pixels). This area is augmented with some semi-transparent overlay buttons to select the zoom level, to adjust the focus, and to display or hide an automatic zoom-adjusted measure grid. This display also reacts to the mouse click to move the sample at the clicked coordinate. The beam size is represented by a semi-transparent rectangle (its size is defined by the vertical and horizontal slits opening, which defines the beam size, scaled according to the zoom value). Below the main display area, two smaller video streams from two surveillance cameras (one near to the sample and the other giving a wide view of the experiment) are displayed. Upon clicking on one of these streams the image is swapped with the main one (by default the on-axis camera view). The front and back light adjustment can be performed by two sliders with a non-linear scale, for better accuracy at low intensity.

In the 'frozen sample' tab, the sample transfer area shows the sample positions in the storage Dewar organized in three SPINE-format pucks, with notification of the puck detection (right-hand side of Fig. 3). The user has information on the current sample in use and can select by a single click a new one to be mounted. Then the G-Rob system, acting as a sample changer, equipped with a double gripper (NatX-ray, France), performs the dismount/mount operation on the MD2 goniometer in a single robot movement.

In the '*in situ*' tab (Fig. 4), the sample transfer area displays a drawing of a crystallization plate (a list box allows the manufacturer and the type of the plate to be selected). Upon selection of a new plate type, the drawing is automatically refreshed. The user can choose a well by clicking on the plate. Then sample centering is performed in the same way as for a frozen crystal, by clicking on the crystals displayed by the on-axis camera on the left-hand part of the tab. Plate handling is performed by the G-Rob system, that acts as a sample changer and as a goniometer.

An important feature of the *in situ* mode is the possibility for the user to register several sample positions by centering the sample and asking to record this position in a list. This list of positions can then be used for automated data collections on the series of positions (samples) without any other user intervention. The present limitation of this mode is that the list is valid only for the plate currently handled. Upon one dismount/mount cycle, the positioning reproducibility being about 0.1 mm, sample positions may be shifted. In a first approach, a global correction can be given by the user, based on the shift observed on a few of the listed samples. A most elaborate correction, based on past experience with automated drop location and sample recognition, will be considered in the future (Heidari Khajepour *et al.*, 2013).

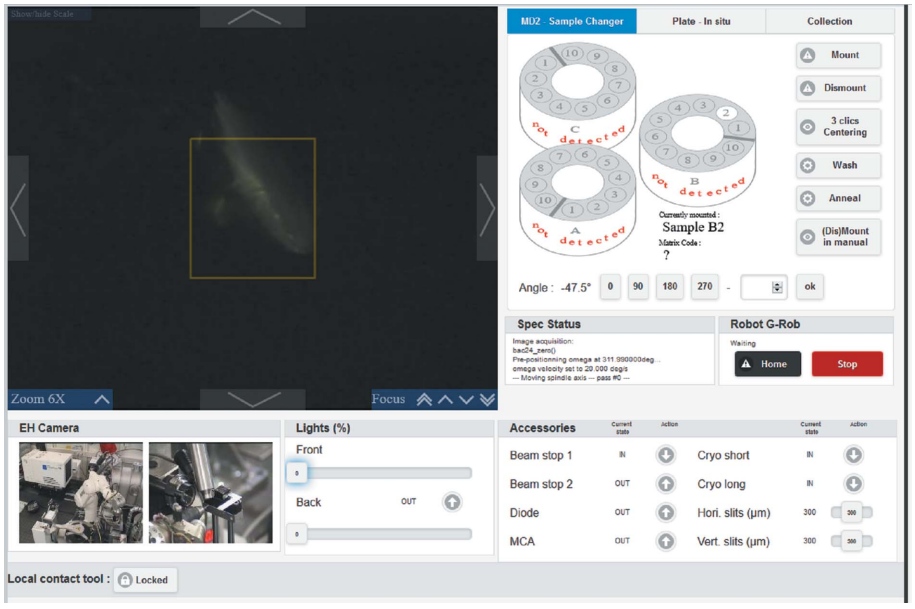


Figure 3
Screen-shot of the *WIFIP* tab for the handling of frozen samples.

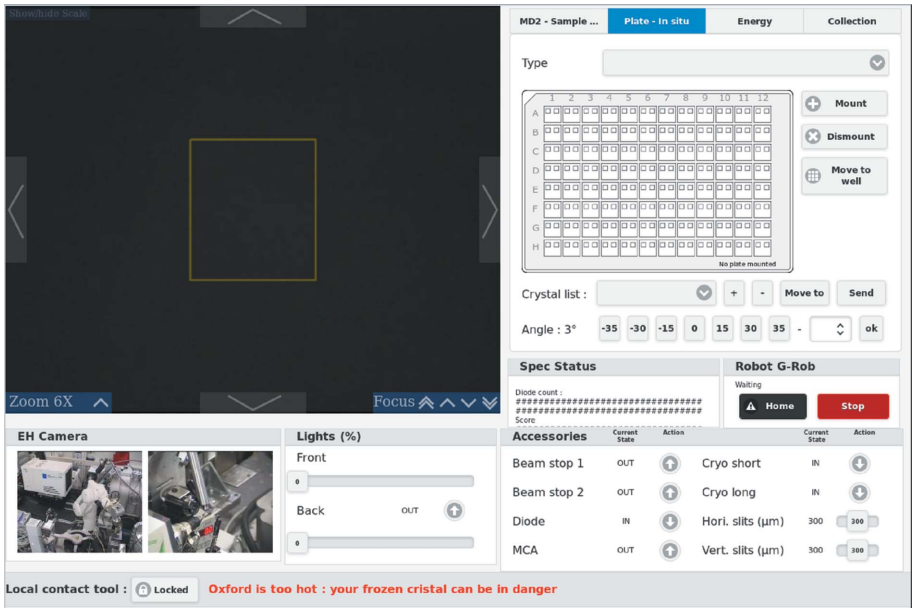


Figure 4
Screen-shot of the *WIFIP* tab for the handling of samples *in situ*.

All drawings (pucks and plates) use a canvas method in HTML5. The drawing is made locally by the browser from the JavaScript, using primitives (line, circle, box).

The currently mounted sample information is stored in a TANGO attribute in the robot device server. This information consists of the type of sample (crystallization plate or frozen sample) and is used by *WIFIP* to adapt the interface design dynamically:

(i) If a plate is mounted, the frozen sample changer tab is disabled until the plate is dismounted, and *vice versa*.

(ii) Data collection options differ between these two types of sample and the interface shows only the adequate one.

3.3. Energy and multi-channel analyzer

From the 'Energy' tab (Fig. 5), the user can choose the beamline energy from a list of pre-calibrated values. Then the beamline optical elements move to the tabulated values, prior to automated final optimization based on diode monitors placed between optical elements.

In this tab, one can also access the MCA and measure an X-ray fluorescence spectrum of the sample, displayed on the left-hand side of the tab. When the user clicks on the plot, the five closer spectral lines are displayed with extra information (element and line type). A clickable periodic table also allows annotations to be added on the plot about the desired element.

In the same tab, one can also measure the fluorescence spectrum at the absorption edge of a selected element (energy scan). This spectrum is automatically analyzed using *CHOOCH* (Evans & Pettifer, 2001) and f' and f'' are displayed at the bottom of the interface, with estimation of minimum/maximum values.

For each mode, MCA data are autoscaled and real-time plotted in *WIFIP*.

3.4. Data collection

In the 'data collection' tab (Fig. 6), the on-axis display is replaced by a diffraction image from the detector. When the user clicks on this image, a full-scale zoom is displayed in the bottom left-hand corner. At the bottom in the middle, a plot shows the intensity of the collected diffraction frames. The right-hand part of the interface is used for setting up, starting, stopping and pausing the data collection.

3.5. Sessions

Since the beamline state is stored in the web server, there is a persistence of the *WIFIP* state. Indeed, when opening *WIFIP*, starting a data collection and closing the browser windows, opening a new session will show the same state as before closing, without affecting the experiment in progress. This session persistence enables the beamline staff to remotely assist the user. Also, it makes it possible for a supervisor or a collaborator to easily check on the progress and take over if necessary. In this way, remote collaborative work and student training is facilitated.

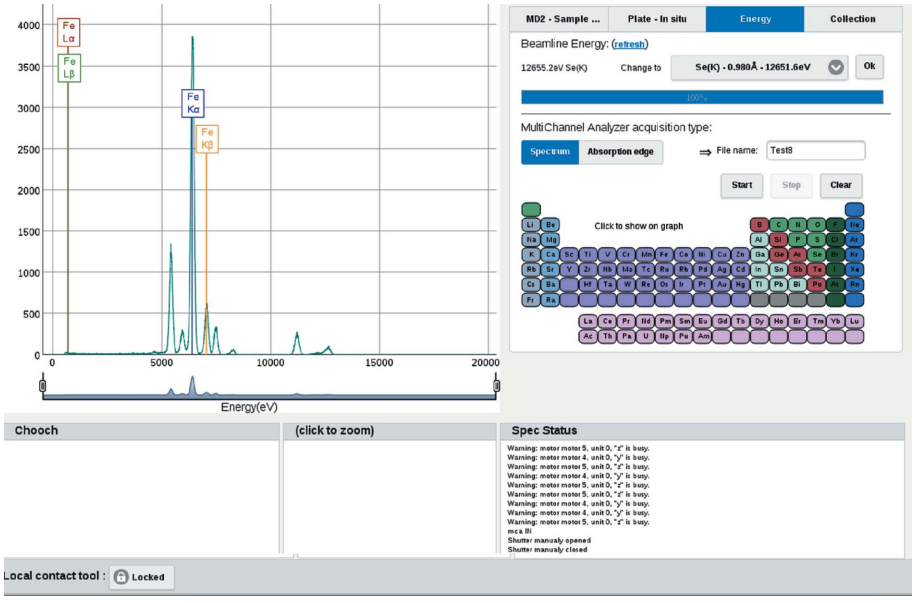


Figure 5
Screen-shot of the *WIFIP* page in energy and multi-channel analyzer mode.

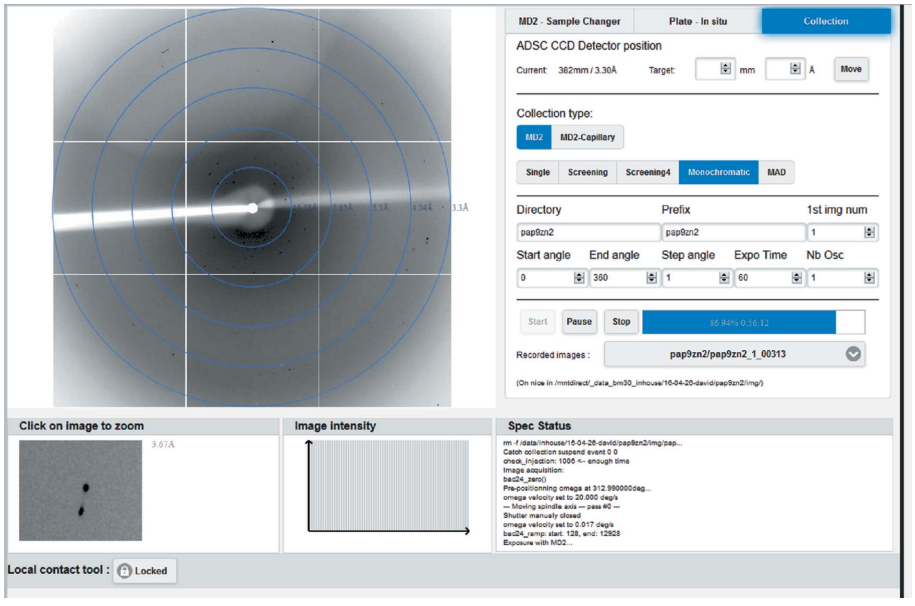


Figure 6
Screen-shot of the *WIFIP* page in the data-collection mode.

Table 1
Browser requirements for *WIFIP* compatibility.

Browser	Minimum version required
Mozilla Firefox	17
Microsoft Edge	1
Browser using Webkit engine or a fork	535.11
Google Chrome	17
Chromium	17
Opera	15
Apple Safari	6

Multiple users can access the interface simultaneously. For now we choose to allow each logged user on *WIFIP* to be master of the interface simultaneously. A more secured access management will be implemented in the future if needed.

4. Compatibility and performances

The technological choices were made keeping in mind the desire of compatibility with the largest number of IT platforms. *WIFIP* was tested on different platform and browser versions to determine compatibility limits. It shows that use of the JQuery library and HTML5 canvas run without any compatibility issue on configurations that are described in Table 1. From January 2016 onwards, this requirement is satisfied by more than 80% of the worldwide browsers on the market (*StatCounter*, http://gs.statcounter.com). Apart from Safari and Edge, other browsers are available on Linux, Windows and Mac OS platforms.

The reactivity of such a system is quite satisfactory. From the local network the time between a user action (a front light switched to 'on' is used in this measurement) and its result (seen on camera) is around 60 ms. From outside of the ESRF (SSH tunneling with port forwarding), this time ranges from 80 ms [test made from a laboratory at the Université Paris Sud connected to the RENATER network (D'Anfray & Simon, 2006)] and 150 ms with a standard ADSL 12 Mb/1 Mb connection. It is good enough for the user to feel an instant action in most cases. The slowest part is the status return delay, due to the state read in the JSON object each second. But most of the elements with changing status are slower in comparison (detector distance moving, sample transfer, *etc.*). In addition, the Python web server detects the user connection source and limits the camera rate to 15 frames s^{-1} when the connection is established from outside of the site. In this case the required bandwidth is only of 300 kb s^{-1}.

5. Conclusion

Since its conception in September 2015 on FIP, more features have been added to *WIFIP*. In parallel, staff-dedicated tools have also been migrated to *WIFIP* to help the staff to intervene remotely. Now users and staff share the main part of this web-based interface on the beamline with all the advantages discussed above. No specific issue was encountered caused by its technology during the migration to this interface. Extra features are under development, such as the integration of *ISPyB* and the auto-processing queue (Monaco *et al.*, 2013). *WIFIP* demonstrates the feasibility of web-based technology for the beamline control. Version 3 of *MXCuBE*, presently under development, will also use a web interface, and will be based on the same kind of architecture. Therefore, *WIFIP* is also a proof of concept for the technologies to be used during future WUI developments.

Acknowledgements

The FIP beamline is funded by the French Alternative Energies and Atomic Energy Commission (CEA) and National Center for Scientific Research (CNRS). We would like to acknowledge fruitful feedback and comments from users and staff of the FIP beamline during the commissioning of this interface. We would like to thank Stéphanie Roccia for her careful reading of the manuscript.

References

Chaize, J.-M, Götz, A., Klotz, W.-D., Meyer, J., Perez, M. & Taurel, E. (1999). *7th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'99)*, 4–8 October 1999, Trieste, Italy.
Chakrabarti, S. (2001). *Proceedings of the 10th International Conference on World Wide Web (WWW '01)*, 1–5 May 2001, Hong Kong, pp. 211–220 (http://doi.acm.org/10.1145/371920.372054).
Crockford, D. (2006). *The application/json Media Type for JavaScript Object Notation (JSON)*, Internet Engineering Tasking Force (IETF) RFC4627 (https://datatracker.ietf.org/doc/rfc4627/).
D'Anfray, P. & Simon, F. (2006). *HPDC 2006. IEEE International Symposium on High Performance Distributed Computing.* Paris, France (https://hal.inria.Fr/hal-00684404).
ESRF (2017). *ESRF remote access*, http://www.esrf.eu/UsersAndScience/Experiments/MX/How_to_use_our_beamlines/remote-access.
Evans, G. & Pettifer, R. F. (2001). *J. Appl. Cryst.* **34**, 82–86.
Gabadinho, J., Beteva, A., Guijarro, M., Rey-Bakaikoa, V., Spruce, D., Bowler, M. W., Brockhauser, S., Flot, D., Gordon, E. J., Hall, D. R., Lavault, B., McCarthy, A. A., McCarthy, J., Mitchell, E., Monaco, S., Mueller-Dieckmann, C., Nurizzo, D., Ravelli, R. B. G., Thibault, X., Walsh, M. A., Leonard, G. A. & McSweeney, S. M. (2010). *J. Synchrotron Rad.* **17**, 700–707.
GÉANT (2017). *Eduroam*, https://www.eduroam.org/.
González, A., Moorhead, P., McPhillips, S. E., Song, J., Sharp, K., Taylor, J. R., Adams, P. D., Sauter, N. K. & Soltis, S. M. (2008). *J. Appl. Cryst.* **41**, 176–184.
Heidari Khajepour, M. Y. H., Lebrette, H., Vernede, X., Rogues, P. & Ferrer, J.-L. (2013). *J. Appl. Cryst.* **46**, 740–745.
Hellkamp, M. (2009). *Bottle: Python web framework*, http://bottlepy.org.
Jacquamet, L., Ohana, J., Joly, J., Borel, F., Pirocchi, M., Charrault, P., Bertoni, A., Israel-Gouy, P., Carpentier, P., Kozielski, F., Blot, D. & Ferrer, J.-L. (2004*b*). *Structure*, **12**, 1219–1225.
Jacquamet, L., Ohana, J., Joly, J., Legrand, P., Kahn, R., Borel, F., Pirocchi, M., Charrault, P., Carpentier, P. & Ferrer, J.-L. (2004*a*). *Acta Cryst.* D**60**, 888–894.
Jong, L. M., Aragao, D., Caradoc-Davies, T., Clift, M., Cowieson, N., Felzmann, C. U. & Mudie, N. (2015). Presented at the *15th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPCS 2015)* Melbourne, Australia.
Kamath, S. & Jackson, J. R. (2004). *Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems and Computers*, Vol. 2, pp. 1723–1726. IEEE.
Lawton, G. (2008). *Computer*, **41**, 10–12.

McPhillips, T. M., McPhillips, S. E., Chiu, H.-J., Cohen, A. E., Deacon, A. M., Ellis, P. J., Garman, E., Gonzalez, A., Sauter, N. K., Phizackerley, R. P., Soltis, S. M. & Kuhn, P. (2002). *J. Synchrotron Rad.* **9**, 401–406.
Modbus (2004). *Modbus Specifications and Implementation Guides*, http://www.modbus.org/specs.php.
Monaco, S., Gordon, E., Bowler, M. W., Delagenière, S., Guijarro, M., Spruce, D., Svensson, O., McSweeney, S. M., McCarthy, A. A., Leonard, G. & Nanao, M. H. (2013). *J. Appl. Cryst.* **46**, 804–810.
Ronacher, A. (2010). *Flask*, http://flask.pocoo.org.
Roth, M., Carpentier, P., Kaïkati, O., Joly, J., Charrault, P., Pirocchi, M., Kahn, R., Fanchon, E., Jacquamet, L., Borel, F., Bertoni, A., Israel-Gouy, P. & Ferrer, J.-L. (2002). *Acta Cryst.* D**58**, 805–814.
Tatham, S. (1997). *PuTTY*, http://www.putty.org/.
Taurel, E., Fernandez, D., Ounsy, M. & Scafuri, C. (2005). *Proceedings of the 10th International Conference on Accelerator and Large Experimental Physics Control Systems (ICALEPS 2005)*, 10–14 October 2005, Geneva, Switzerland.
W3C (2014). *HTML5 Differences from HTML4*, https://www.w3.org/TR/html5-diff/.