Bringing MX experiments to the web
MXCuBE 3
● An MX Beamline at glance

● Project background

● Backend, beamline control layer

● Web service layer

● Frontend Development

● Screenshots
An MX beamline

- Aim is to resolve **protein structure**

An X-ray beam is focused on a **crystal**

The crystal is rotated and **diffraction images** are captured

An **electron density map** is calculated from the diffraction images

It’s possible to resolve the **molecular structure** from the electron density map
An MX beamline

MX-Beamline

- 2D - Detector
- Diffractometer
- Sample changer
- And various other instruments
  - Monochromator
  - Mirrors
  - Filters
  - Aperture
  - ...

MXCuBE is the application used for instrument control and data acquisition
Project history and background

MXCuBE 1 in 2005

MXCuBE 2 released in May 2012

- MXCuBE is now an international **collaboration**, with 8 partner institutes
- **Same familiar interface** on all sites
- Partners can easily adapt to **existing solutions**

New software needed, decision made to implement MXCuBE as a Web Application

- Better remote access performance
- Keeping up with new instrument capabilities
- Improved user experience
- Easing the maintenance and client install
MXCuBE 3 Development started by ESRF and MAXIV in September 2015

MXCuBE 3.0 to be released in early November 2017
  ○ Pre release already in use at MAXIV since June 2016
  ○ Preliminary user feedback is very positive (Poster presentation by Mikel Eguiraun TUMPL08 this afternoon)

Version 3.1 scheduled for second quarter 2018
REST webservices
Asynchronous message broker
SocketIO
Thin utility layer for accessing Hardware Repository

Beamline control layer
Hardware and procedure abstraction
(Hardware Objects)

WSGI Container (Flask Webserver)

- Built on top of the same beamlne control layer as MXCuBE 2 (Hardware Objects)
- Instruments and procedures are implemented as what is called Hardware Objects
- The beamlne control layer is control system agnostic and supports for instance SPEC, EPICS, Sardana, BLISS and TANGO, *(BLISS Talk by Matias Guijarro, WEBPL05)*
- Base classes define a common API for a particular instrument or procedure, which facilitates cross site adaption
- Defines an API for clients to access the HardwareObjects, and relays events between Hardware Objects and clients (not necessarily a browser).

- Thin utility layer for providing new functionality exclusive to MXCuBE 3 and ease access to Hardware Objects.

- Websockets, via SocketIO, used to relay events from backend.

- Implemented on top of a Flask web server, WSGI container.
- Application written in HTML 5, Javascript 6 (JS6) and CSS
- JS6 gives us the possibility to use reusable components and modules
- Problem, no browser have full JS6 support

Babel and webpack allows us to use reusable modules and classes (https://babeljs.io/) and (https://webpack.github.io/)

ES6 Code is “transpiled” with babel to ES5 which have good support in most browsers

Webpack is used to bundle the various assets, JS, CSS, LESS, Fonts and images to a set of static files that can be loaded by the browser.
React

- React is a library for creating user interfaces
- React makes it possible to use widgets like in traditional UI development
- Provides a way to express the UI in a markup language called JSX
- Can be used with state management library, in order to avoid per widget state

Redux

- Application wide state, only source of data for components.
- The redux store is an immutable data structure and can only be updated (replaced) by a pure function, a reducer
- The reducer function is called by dispatching an action for instance when user interacts with UI
- Provides unidirectional data flow, easy to debug

---

**React**

https://facebook.github.io/react/

- Component development with React
  - React is a library for creating user interfaces
  - React makes it possible to use widgets like in traditional UI development
  - Provides a way to express the UI in a markup language called JSX
  - Can be used with state management library, in order to avoid per widget state

**Redux**

http://redux.js.org/

- Application wide state, only source of data for components.
- The redux store is an immutable data structure and can only be updated (replaced) by a pure function, a reducer
- The reducer function is called by dispatching an action for instance when user interacts with UI
- Provides unidirectional data flow, easy to debug
Data collection view, for interactive data collection and sample alignment
Sample overview, samples represented as cards. Gives the possibility to apply data collections over a set of samples and run them in an automatic sequence.
Thanks to everybody involved in the project, especially staff from MAX IV and ESRF
(Picture from last MXCuBE ISPyB meeting at Soleil)