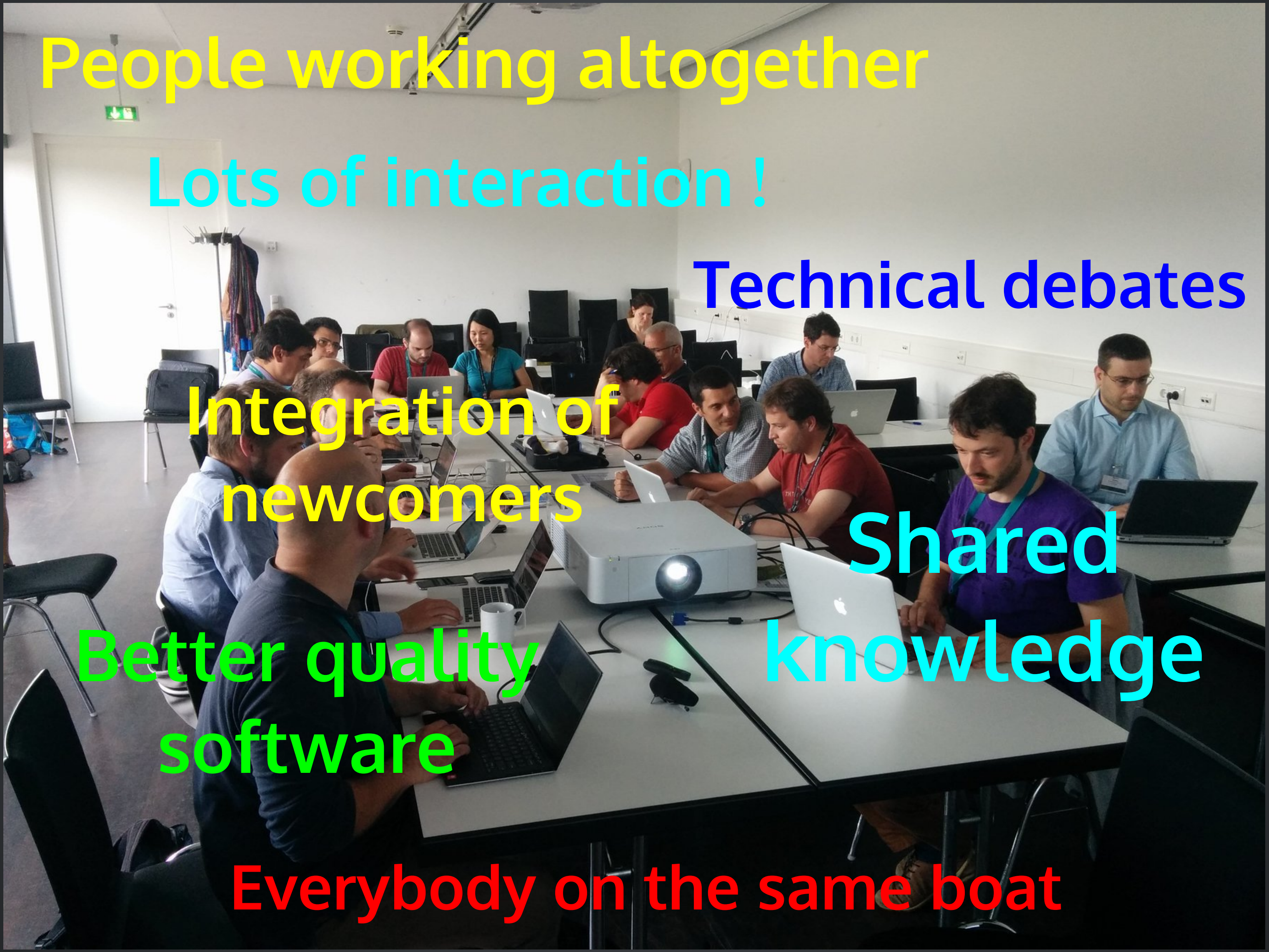
MXCuBE meeting @ Diamond Light Source, 1 February 2018

Less is more:

a proposal to enhance collaboration

Minimalistic presentation by M. Guijarro

Collaboration from a developer perspective



MXCuBE collaboration

What do we share ? (from the MOU)

- 2 graphical frontends
- Beamline control abstraction layer

New developers have difficulties to participate

3 developers are pushing the vast majority of commits

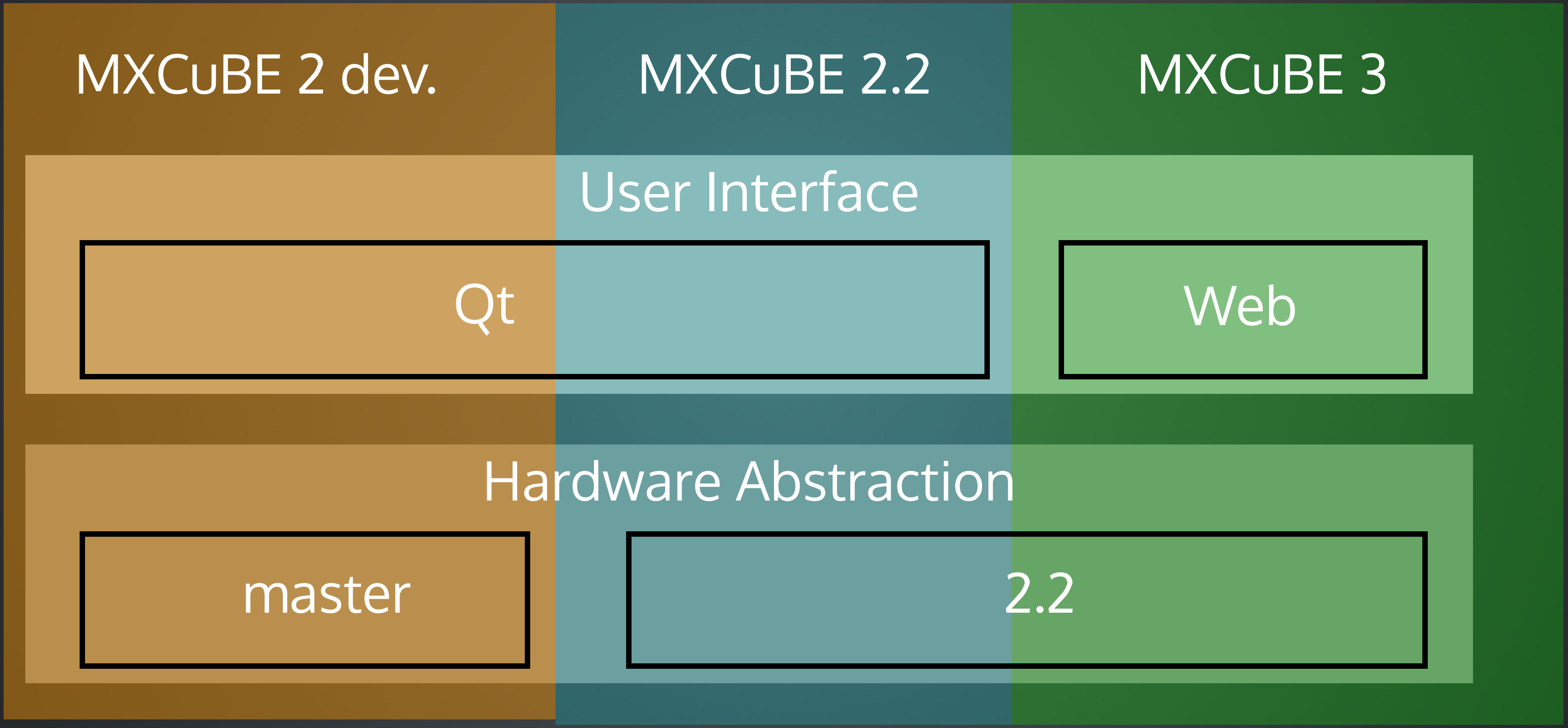
Challenges ahead: test suite, Python 3, modernizing software architecture

Impediments to developers collaboration

- Code organization
- Lack of documentation
- Code complexity
- No tests suite

Code Organization

- Git repositories with branches & submodules



⚠ **Hard to see the big picture, hard to make pull requests, hard to keep coherency**

Code Organization: Hardware Abstraction

- Abstract Hardware Objects -- 10
- Mockup Hardware Objects -- 31
- Site-specific Hardware Objects
  - 48 (ESRF)
  - 23 (EMBL-HH)
  - 14 (MAXIV)
  - 70 (SOLEIL)

⚠ **Missing abstract classes**

⚠ **80% of the code is specific, impossible to test**

Code complexity

- Loosely tied components
  - subscription and events emitting
- Many inheritance levels
- Long callback chains
- Beamline control code

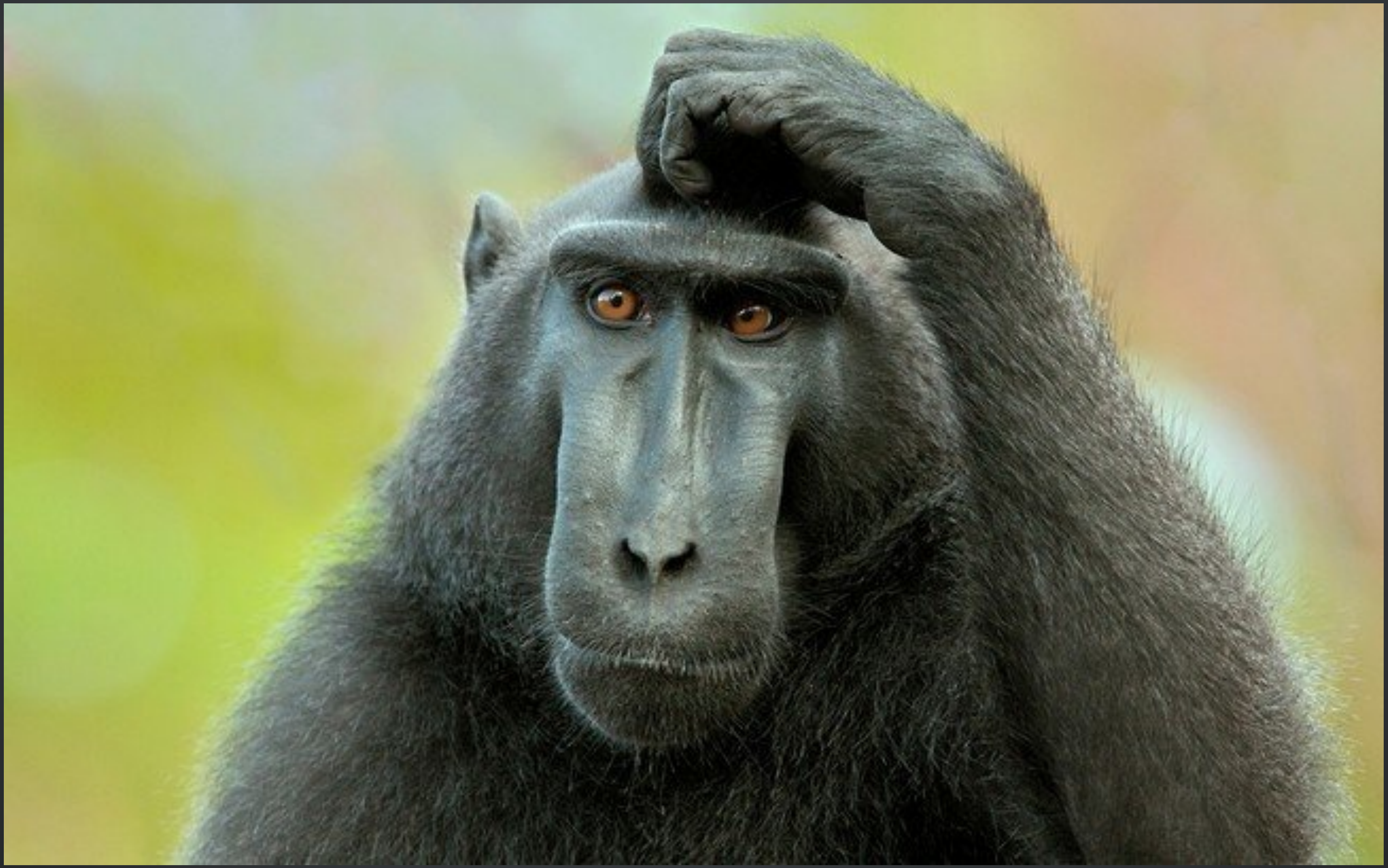
Hard to get into the code, hard to debug

No automatic testing: regression steps in !

MXCuBE collaboration entry price is too high

MXCuBE maintenance cost is high too

Is MXCuBE architecture adapted to current needs?



Digression:

something to tell about MXCuBE 3 development

MXCuBE 3 control API

- Queue

```
@mxcube.route("/mxcube/api/v0.1/queue/start", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/stop", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/abort", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/pause", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/unpause", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/clear", methods=['PUT', 'GET'])
@mxcube.route("/mxcube/api/v0.1/queue", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue_state", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/<sid>/<tindex>/execute", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/<sqid>/<tqid>", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/delete", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/set_enabled", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/<sid>/<ti1>/<ti2>/swap", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/<sid>/<ti1>/<ti2>/move", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/sample-order", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/queue/<sample_id>", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/<node_id>/toggle", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/queue/dc", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/char_acq", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/char", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/mesh", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/<id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/<sample_id>/<int:method_id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/queue/json", methods=["GET"])
@mxcube.route("/mxcube/api/v0.1/queue/automount", methods=["POST"])
@mxcube.route("/mxcube/api/v0.1/queue/num_snapshots", methods=["PUT"])
@mxcube.route("/mxcube/api/v0.1/queue/group_folder", methods=["POST"])
@mxcube.route("/mxcube/api/v0.1/queue/group_folder", methods=["GET"])
@mxcube.route("/mxcube/api/v0.1/queue/auto_add_diffplan", methods=["POST"])
```

MXCuBE 3 control API

- Data collection

```
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>/mode", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>", methods=['DELETE'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/status", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/samples/<id>/collections/<colid>/status", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/samples/<sampleid>/collections/<colid>/run", methods=['POST'])
```

- Access to beamline setup

```
@mxcube.route("/mxcube/api/v0.1/beamline", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/beamline/<name>/abort", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/beamline/<name>/run", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/beamline/<name>", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/beamline/<name>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/beam/info", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/beamline/datapath", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/beamline/prepare_beamline", methods=['PUT'])
```

MXCuBE 3 control API

- Sample changer

```
@mxcube.route("/mxcube/api/v0.1/sample_changer/samples_list", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/state", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/loaded_sample", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/contents", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/select/<loc>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/scan/<loc>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/mount/<loc>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/unmount/<loc>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/unmount_current/", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/mount", methods=["POST"])
@mxcube.route("/mxcube/api/v0.1/sample_changer/unmount", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/get_maintenance_cmds", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/get_global_state", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/get_initial_state", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sample_changer/send_command/<cmdparts>", methods=['GET'])
```

- Login

```
@mxcube.route("/mxcube/api/v0.1/login", methods=["POST"])
@mxcube.route("/mxcube/api/v0.1/signout")
@mxcube.route("/mxcube/api/v0.1/login_info", methods=["GET"])
@mxcube.route("/mxcube/api/v0.1/login/request_control", methods=["POST"])
@mxcube.route("/mxcube/api/v0.1/login/observers", methods=["GET"])
@mxcube.route("/mxcube/api/v0.1/login/request_control_response", methods=["POST"])
```

MXCuBE 3 control API

- Sample centring, sample view handling

```
@mxcube.route("/mxcube/api/v0.1/sampleview/camera/subscribe", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/camera/unsubscribe", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/camera/save", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/camera", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/camera", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/<point_id>/moveto", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/shapes", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/shapes/<sid>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/shape_mock_result/<sid>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/shapes", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/sampleview/shapes/<sid>", methods=['DELETE'])
@mxcube.route("/mxcube/api/v0.1/sampleview/zoom", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/backlighton", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/backlightoff", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/frontlighton", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/frontlightoff", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/<motid>/<newpos>", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/<elem_id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/startauto", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/start3click", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/abort", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/click", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/accept", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/reject", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/movetobeam", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/sampleview/centring/centring_method", methods=['PUT'])
```

MXCuBE 3 control API

- Diffractometer

```
@mxcube.route("/mxcube/api/v0.1/diffractometer/phase", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/phaselist", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/phase", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/platemode", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/movables/state", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/aperture", methods=['PUT'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/aperture", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/diffractometer/info", methods=['GET'])
```

- LIMS (ISPyB) integration

```
@mxcube.route("/mxcube/api/v0.1/lims/samples/<proposal_id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/lims/dc/thumbnail/<image_id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/lims/dc/<dc_id>", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/lims/proposal", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/lims/proposal", methods=['GET'])
```

- External experiment control (workflows, need extra server)

```
@mxcube.route("/mxcube/api/v0.1/workflow", methods=['GET'])
@mxcube.route("/mxcube/api/v0.1/workflow", methods=['POST'])
@mxcube.route("/mxcube/api/v0.1/workflow/dialog/<wf>", methods=['GET'])
```

Only 120 API functions are needed to have all features of MXCuBE

For completeness: 21 signals have to be emitted too

```
# diffractometer
socketio.emit('diff_phase_changed', data, namespace='/hwr')

# sample changer
socketio.emit('sc', msg, namespace='/hwr')
socketio.emit('sc_state', state_str, namespace='/hwr')
socketio.emit("loaded_sample_changed", {'address': address, 'barcode': barcode}, namespace="/hwr")
socketio.emit("set_current_sample", sample , namespace="/hwr")
socketio.emit("sc_contents_update")
socketio.emit("sc_maintenance_update", {'state': json.dumps(state_list), 'commands_state': json.dumps(cmd_state), 'me

# sample centring
socketio.emit('sample_centring', msg, namespace='/hwr')
socketio.emit('update_shapes', {'shapes': shape_dict}, namespace='/hwr')
socketio.emit('update_pixels_per_mm', {"pixelsPerMm": ppm}, namespace='/hwr')
socketio.emit('beam_changed', {'data': ret}, namespace='/hwr')

# results and plotting
socketio.emit('grid_result_available', {'shape': shape}, namespace='/hwr')
socketio.emit('energy_scan_result', {'pk': pk, 'ip': ip, 'rm': rm}, namespace='/hwr')
socketio.emit("new_plot", plot_info, namespace="/hwr")
socketio.emit("plot_data", data, namespace="/hwr")
socketio.emit("plot_end", data, namespace="/hwr")

# beamline setup
socketio.emit('motor_position', movable, namespace='/hwr')
socketio.emit('motor_state', movable, namespace='/hwr')
socketio.emit("beamline_action", msg, namespace="/hwr")
socketio.emit("beamline_value_change", data, namespace="/hwr")
socketio.emit("mach_info_changed", values, namespace="/hwr")
```

Identification of MXCuBE base building blocks

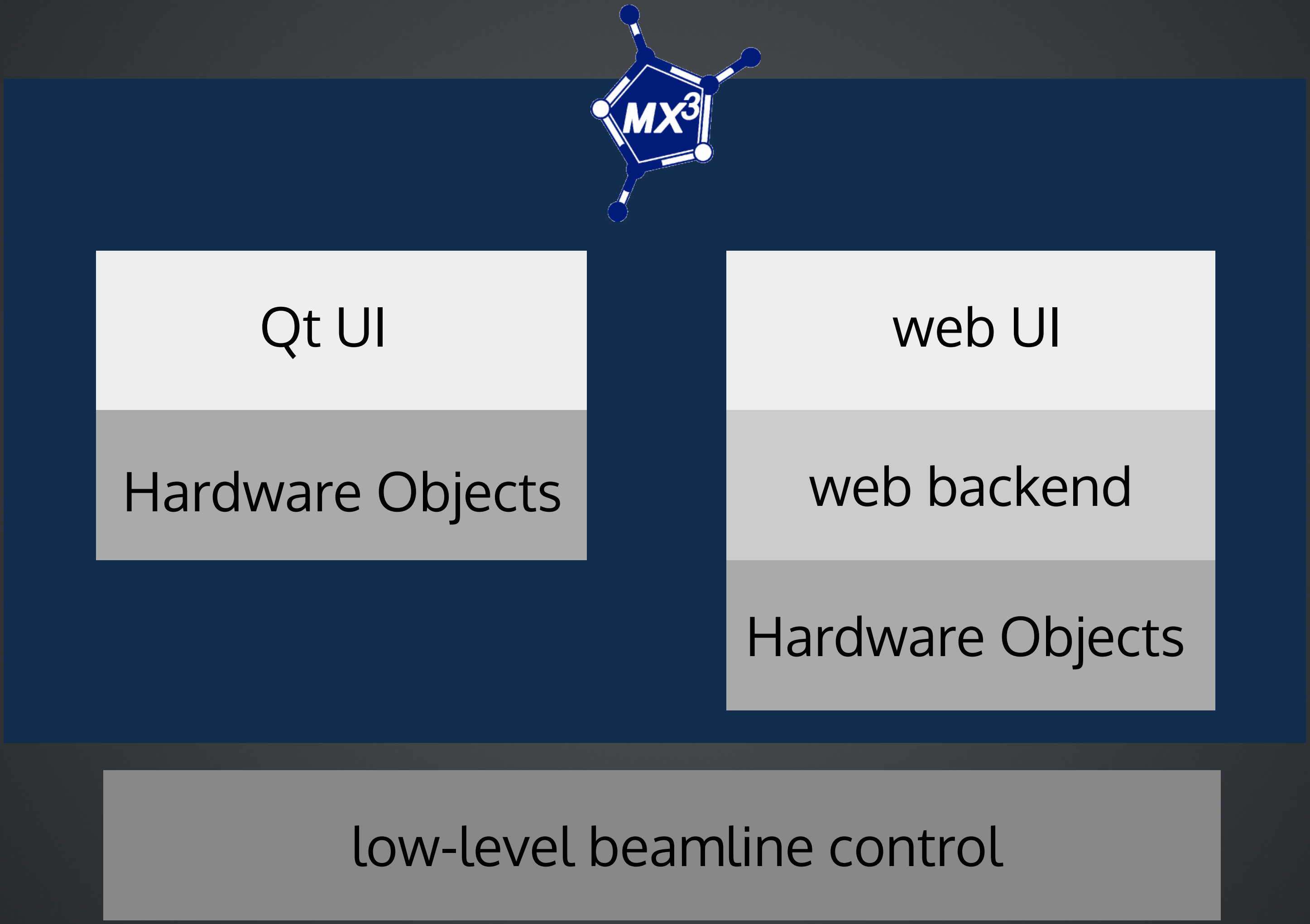
- Login
- LIMS
- Beamline setup
- Sample Changer
- Sample centring
- Diffractometer
- Queue
- Data collection
- External exp. control

A proposal to enhance collaboration

Let's upgrade the Abstraction idea

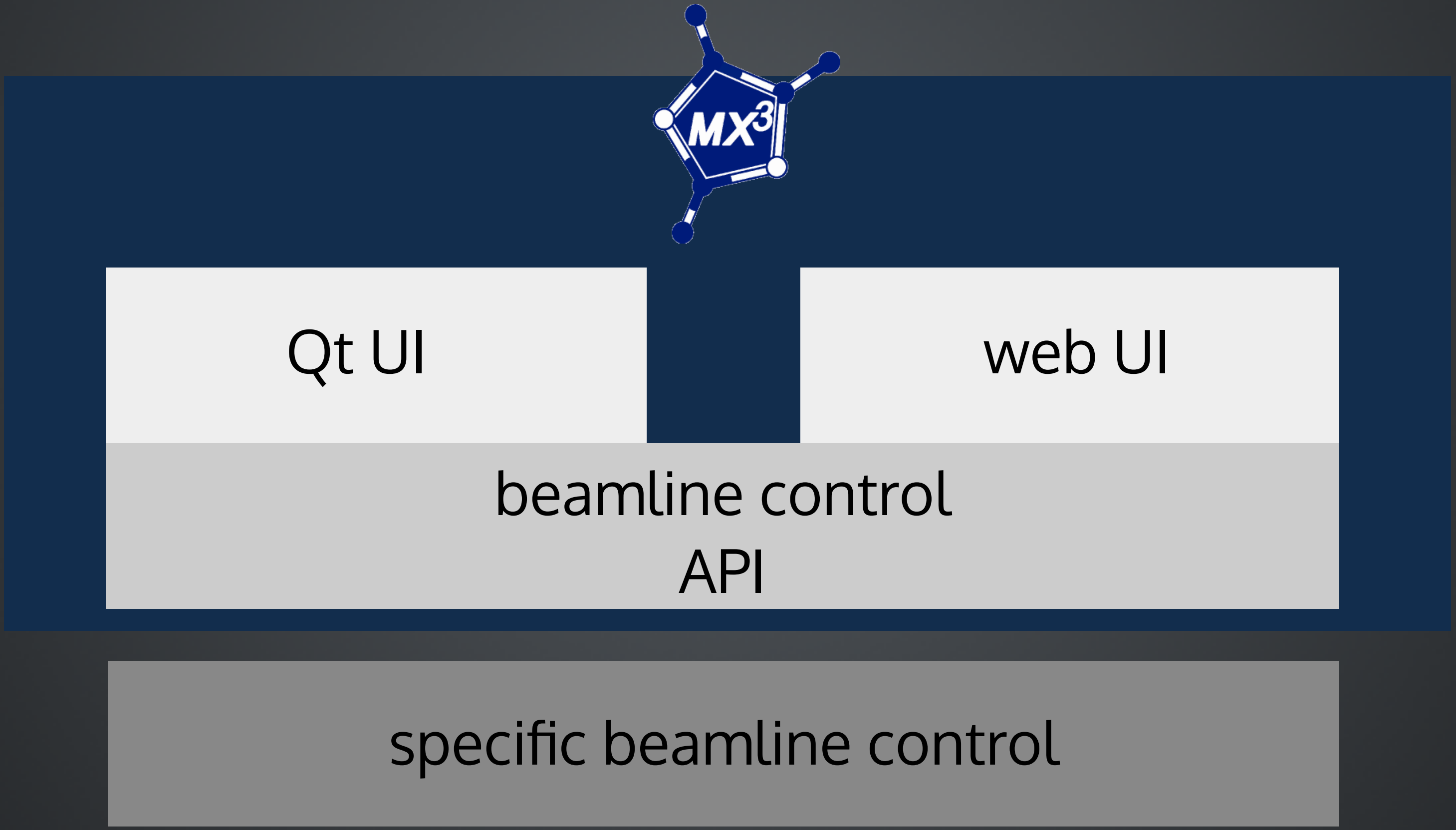
Let's facilitate test/simulation

Current architecture



Moving to a higher-level abstraction

- New architecture proposal inspired by work on MXCuBE 3



Conclusion

Less is more

- What about removing beamline-specific code from MXCuBE repository ?
    - makes it more clear what is really shared of MXCuBE
- Beamline control layer inspired by MXCuBE 3 as a "contract" between UI and underlying hardware control
    - only 120 functions

Much cleaner API for User Interfaces

API documentation would be straightforward to write

Good use case for semantic version numbers

Complete simulation environment is possible

Continuous Integration objective could be achieved

Open Questions

About the speaker...

- Since this autumn the time I can dedicate to MXCuBE development has been drastically reduced
- New duties at ESRF: team leader of the BLISS project

This is not good-bye BUT...

please Steering Committee make sure MXCuBE has enough developers !