Science Working Group Steering Committee

Google Hangouts meeting 2016.07.20

Attendees: Jay Jay Billings, Torkild Resheim, Andrea Ross, Matthew Gerring
Absentees: Tracy Miranda

The previous meeting was at 2016.05.18.

Old business

Project updates

DAWNSci - Not much to report. Baha has simplified some of the dependencies. Still chugging along. Not a fully fledged Eclipse project any time soon.
PTP - Two things. Participated in Neon. Greg is working on a new protocol for the remote parts of PTP so that it can work with the largest HPC in the US, including Titan.
Triquetrum - Python actor coming in. Preparing for the release.
Eclipse Advanced Visualization Project - A lot of work going on and huge amount of interest. Refactored the geometry editor on a request from MARINTEK. Can now build using triangles. Data model has been completely reworked. Not quite ready for a paper pitch for EclipseCon Europe. Put in a lot of work in other types of visualization, i.e. ParaView. Some contributions went to the January project.
Eclipse ICE - Had a tutorial at XSEDE16. None showed up as the map pointed to the wrong room. A lot of clean up and refactoring. Getting ready for the science release. Contributed data structures to the January project.
ChemClipse - No changes. The foundation is concerned with the work required to approve the CQs. Will try to push through one data format and learn from that. CQ 11618 is being used to track.
Rich Beans - Hit an IP problem with the Meta widgets where some code was copied from Google. That was the only issue. The code in question will be removed.
January - Good news is that Jonah has prepared the initial contribution. Some work must be redone. Will contain IDataset from DAWN, form based data structures from ICE and new geometry data structures from EAVP.

Science Top-Level Project

Need to decide on PMC members & esp. PMC lead, which requires Eclipse Foundation board approval. This committee will write a proposal for the group to discuss. Some experienced

Eclipse veterans should be on the PMC such as Greg (lead) and Torkild. In addition each of the project leads should be a member.

There was quite a bit of dissent regarding the LGPL bit. Jay will work with the Foundation to try to get some LGPL prerequisites in place. We must ask the board for permission.

Science 2016 Release

Jay thinks we're on track. ICE and EAVP has been cleared. Still some issues with January. Should be an easy fix. Triquetrum is one CQ short of getting through. We should join up with the release train to avoid some IP related issues.

Website Updates

Torkild has been updating the site with meeting minutes and activities. Most minutes that are available have been uploaded. Planning to finish everything before end of August.

New business

EclipseCon Europe Science Track Update

Only five proposals came in, one of them a keynote about detection of gravitational waves :)

Scanning Project

Project proposal is with Wayne. Matt will say the elevator pitch for the project to give the committee an idea of what it is about.

"scanning allows experiments to be conducted by coordinating the operation of scientific instruments, for example motors or detectors. It sequences the movements of these instruments (or devices) in order to scan different parts of the experimental space. For instance you might scan a temperature controller to conduct an experiment at different temperatures or move a goniometer through a range of optical angles or combine both in a two dimensional scan. Scanning is useful as an open source project because the algorithms which complete scans during experiments are the same in many areas of research. Hardware is experiment specific so scanning algorithms can be used in many settings, whereever electronically controlled hardware does automated experiments."

Jay talks a bit about the quantum computing project: Code is going through a release review. Will be completely open source with no export control limitations. May be part of the SWG but maybe not because of the LGPL restrictions. Some of the dependencies are created by a few or one persons, the only ones who knows how to do this, and we cannot go to these and ask them to redo the code. The code does minor graph embedding and works as a quantum compiler.

Donald Raab from Goldman Sachs would be willing to introduce Eclipse Collections to the Eclipse Foundation. Any interest?

Jay connected Andrea to Sandia re: membership. There doesn't seem to be traction at the moment.

Andrea reached out to HDF group on July 15, 2016 since it had been a little while since the last contact. No response yet.

Los Alamos asked whether they could vote on the TLP, and because they were not a member no. Andrea will reach out to chat with them.

Andrea has been talking up Science with LocationTech members to see if some might join.

IP revision

Andrea is offering to chat about the IP revision if people have any questions after reading Mike's blog post about it

Appendix 1 - Scanning proposal

People on the committee cannot see the proposal unless they are part of it. Therefore I have dumped a copy here:

Background:

Diamond Light Source facility would like to share code for scanning which is possible to reuse at other facilities. The Eclipse Foundation provides a great foundation to do this. The code is in existence and being used to drive a major scientific facility so we would like a long term lifecycle for it. Eclipse foundation helps to provide this.

Scope:

The project will provide the ability to scan scientific hardware. There are a number of core algorithms for scanning, which are common across scientific experiments. The scope of the project is to:

1. Allow any hardware corresponding to a simple Java interface to be integrated with scanning algorithms.

2. Provide user interface for scanning, setting up scans, executing and monitoring them. (Initially in SWT/jface but HTML5 or other front ends would later be in scope.)
3. Provide a python layer, cpython and jython, which is able to drive the scanning algorithms.

The goal is to abstract scanning such that any scientific facility may reuse existing algorithms.

Description:

Executive Summary

Scanning is a project which allows experiments to be conducted by coordinating the operation of scientific instruments, for instance motors or detectors. It is useful as an open source project because the algorithms which complete scans during experiments are the same in many areas of research. It is the hardware which is specific, therefore scanning algorithms can be used in many scenarios. Because of this generic nature, they warrant an eclipse project which abstracts just this part of the software and makes it reusable for different experiments.

Detailed Description

Introduction

Scientific facilities operating high end hardware, for instance robots, motorized stages and detectors, have well defined layers for integrating these devices. For instance it is possible for a call to be made to move a motor or expose a detector (similar to taking an image with a digital camera). Examples of these control layers are the **EPICS** framework, and the **TANGO** framework. These frameworks are open source and in wide use at facilities around the globe. Facilities also require low latency operation, for instance a fast acting motor and a detector may need to coordinate specialized hardware. For people not familiar with this idea, one example of this is the **Zebra** box which combines an ARM processor with an FPGA and can orchestrate hardware times down to the nanosecond scale. There is also a device called **TFG**.

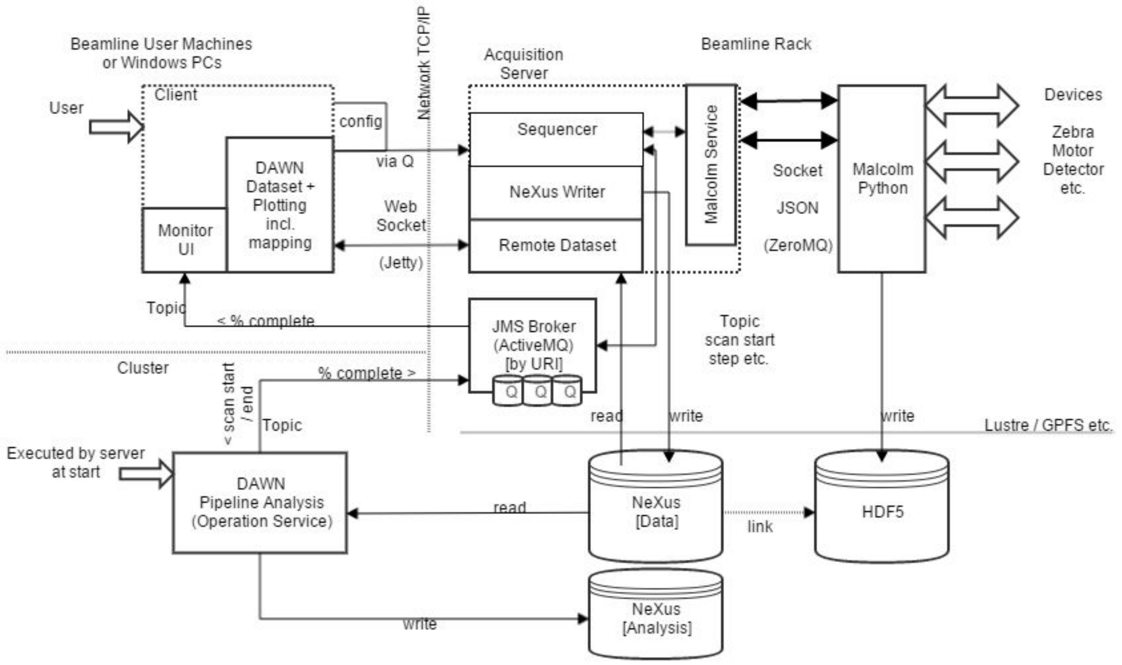
Design

Introduction

The project will be divided into a services layer to provide the scanning features, an example user interface layer implemented in RCP/jface (users may choose to use another UI layer) (this UI will be used at Diamond Light Source) and a cpython/jython layer for easily scripting the scans.

Server and Clients

The services layer may be operated on a separate server and messages passed between the client and server with a messaging system provided by the project. The scanning project does not prescribe exactly how services are used however if they are started in an OSGi container, declarative services will work. If they are not some manual ‘wiring together’ might be required. One possible arrangement of the services for scanning is shown below. This arrangement is site specific to Diamond Light Source and represents a possible deployment of the services layer. (DAWN is used for plotting and analysis here but any package may be used with the scanning project for this purpose.) Different facilities would use the various services in this project different ways.



Services Layer

IPointGeneratorService

The point generator service takes a model and provides a generator, IPointGenerator, which gives each nD point in the scan. So for instance in a mapping scan, each point has a x,y stage value for the two-dimensional scan.

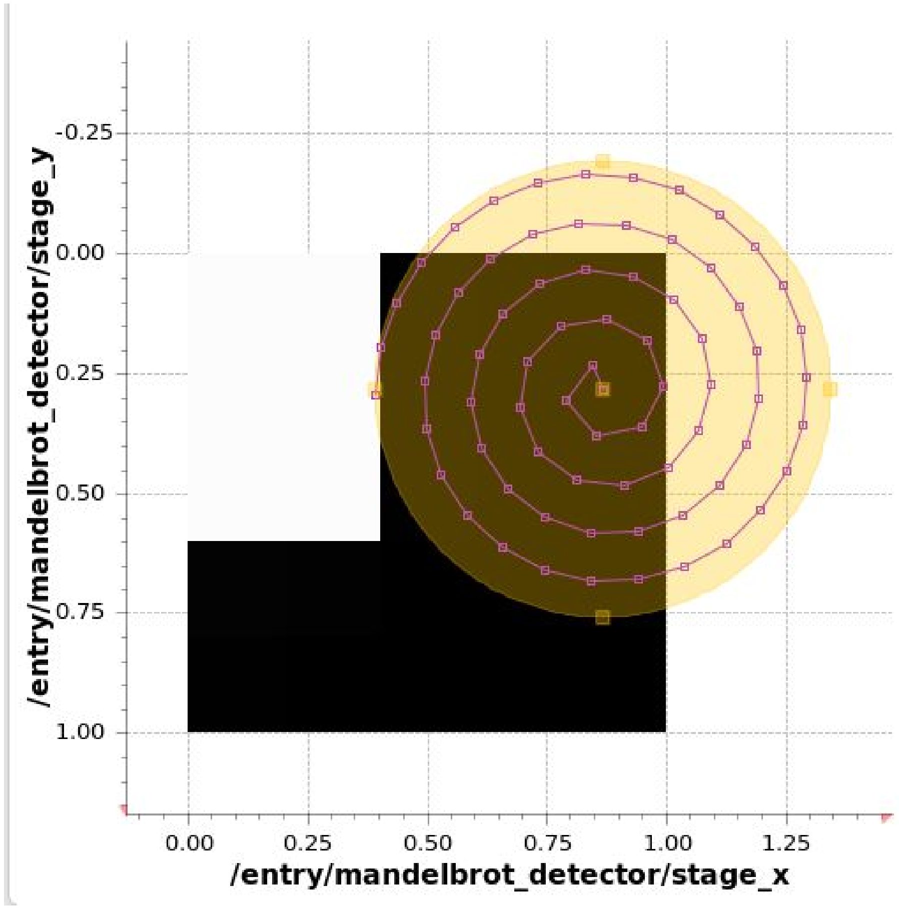
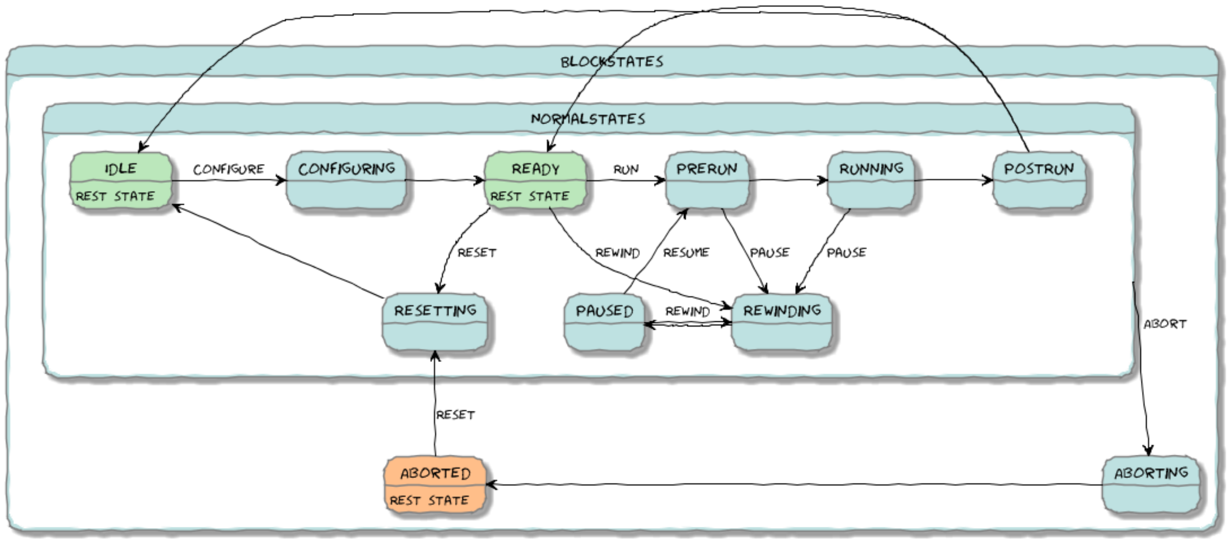


Image showing a spiral scan path over a 2D stage

Generators may be nested (unlimited) to provide complex scans. For instance it is possible to combine a grid with temperature step scan. Generators may be added by extension point so that scan paths not envisaged by the developer of the service can be added. Generators may be added by cpython/jython to allow users to define experimental specific scan procedures, including logic to be executed for each point.

IRunnableDeviceService

This service returns devices, IRunnableDevice, which conform to a state machine for driving scans. It is used to run a scan and is the core interface for scanning. It returns the top level scan which uses a point generator to define each scan point. It returns each device in the scan, for instance 2D detectors or devices driving **EPICS Area Detector** or a **Malcolm Device**, conform to this interface and pass through the states defined.



See **http://pymalcolm.readthedocs.io/en/latest/arch/statemachine.html**

AcquisitionDevice (IRunnableDevice)

When doing a CPU scan the Java-based runnable device uses a thread pool to manage the scan. It combines a move with a detector readout to maximize the scan speed. So for scanning one point we have:

```
_|....|________   run() Tell detector(s) to collect current position


______|.....|___  write() Tell detector(s) to write data,
______|.............|___ setPosition() move motor(s) to next position
```

IDeviceConnectorService

Provides a connection to devices which are to be scanned. The devices take part in the scan by setting/getting value so they represent things like motors, temperature controllers and goniometer angles. Each device connected to must conform to a simple interface called IScannable.

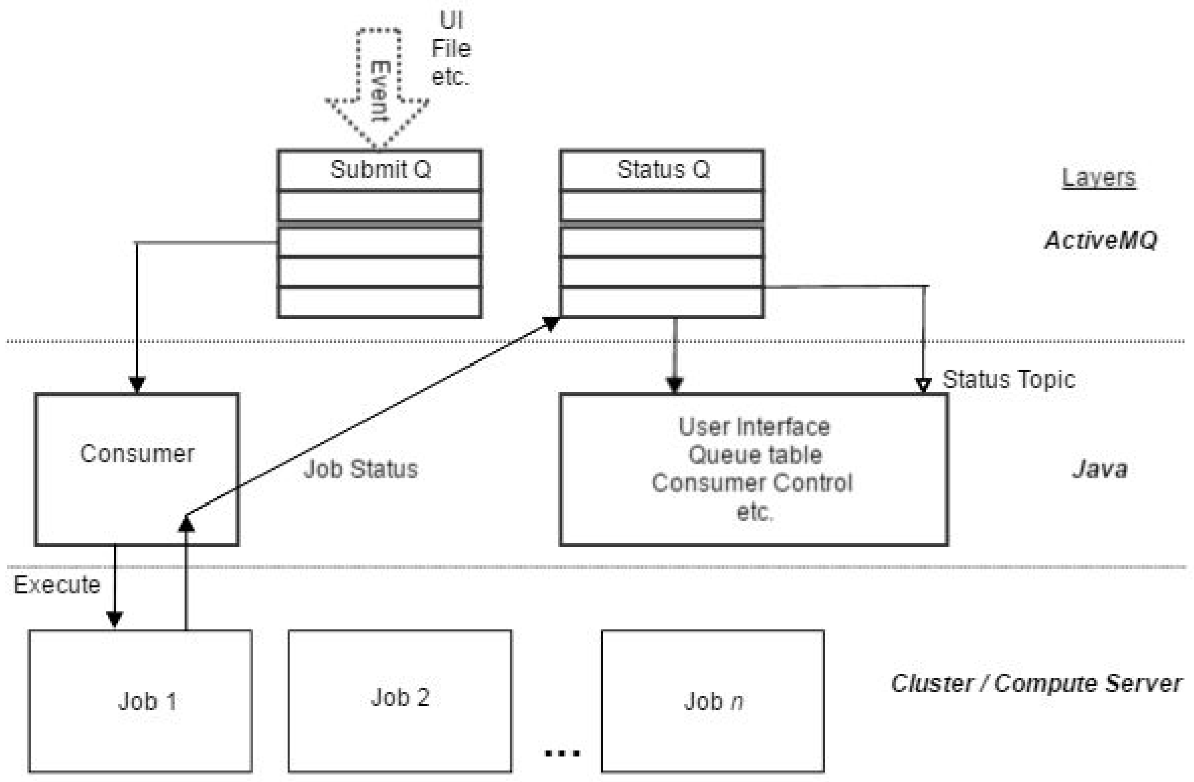
```
IScannable<T> … {

 public T getPosition() throws Exception;

 public void setPosition(T value, IPosition position) throws Exception;

 ...

}
```

IEventService

An event system is required to receive scan requests, notify the user of scan progress and maintain queues. The event service provides the following models:

- Publish/Subscribe   (events like scan finished)
- Submit/Consume   (queues like those for running scans)
- Request/Response (ask the server a question such as how many detectors)

This event system is also used to manage analysis algorithms which are outside the scope of the scanning project.



IMalcolmService

To make it really easy to talk to any hardware, a middleware layer has been designed called 'Malcolm' which bridges between hardware and the runnable device interface/state machine. Malcolm is outside the scope of the scanning project. Malcolm devices which implement IRunnableDevice are made available from it using the IMalcolmService internally to the IRunnableDeviceService. (Malcolm is similar in this respect to TANGO and it may be desirable in future to allow TANGO devices to be scanned and/or allow the scanning device to conform to a TANGO device.)

NeXus Builder Service

The NexusBuilderFactory is used in the scanning to write legal NeXus HDF5 files. NeXus is a self-describing binary file format used in many facilities to record large numerical data efficiently. The **DAWN** product can read any correctly written NeXus file and provides a large armoury of tools with which to analyse data, for example running fast analysis pipelines on clusters.

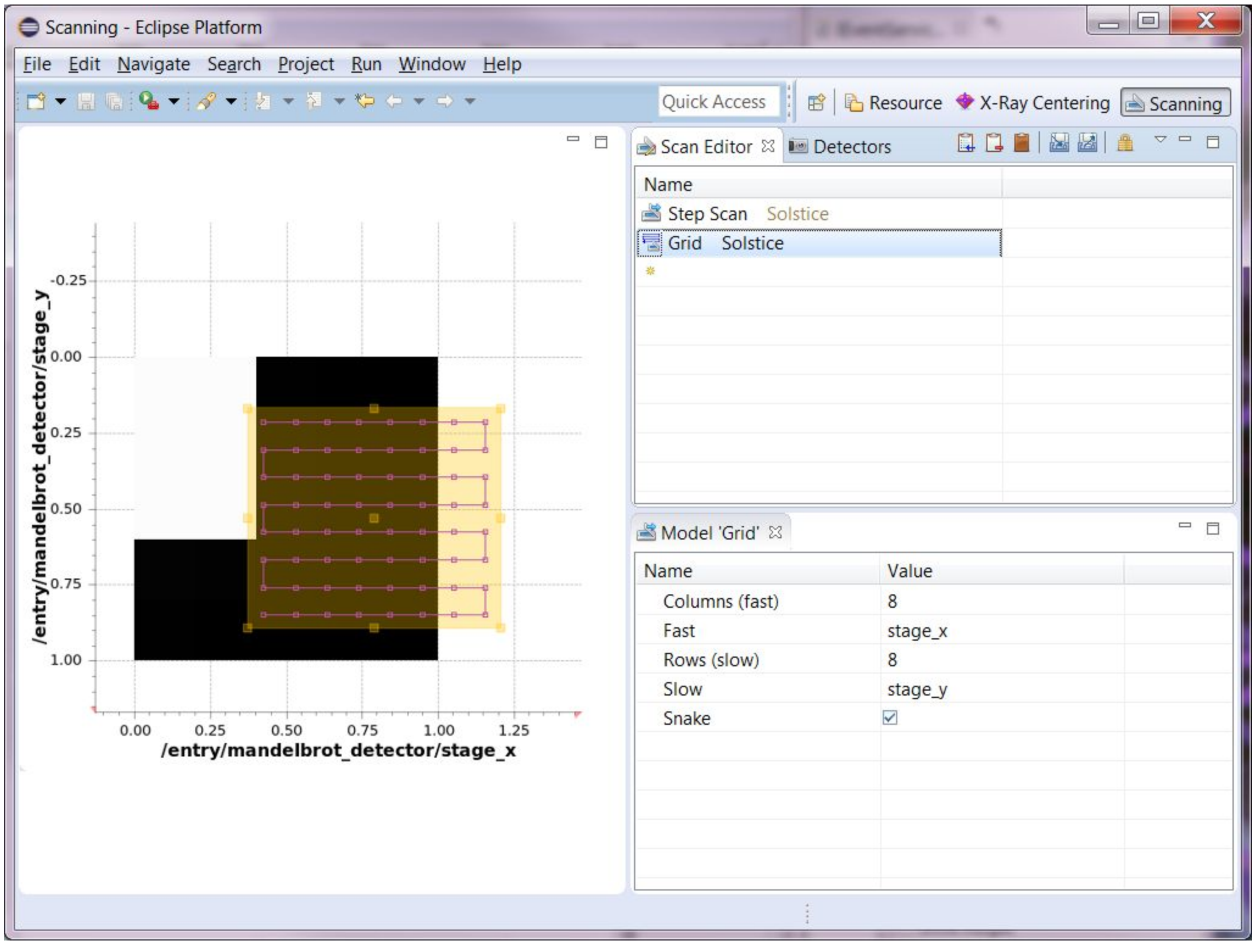
Devices can be integrated with NeXus by implementing a declarative interface called INexusDevice.

Other Services

There are several other services for connecting to python, running scripts etc. which are included in the project.

User Interface Layer

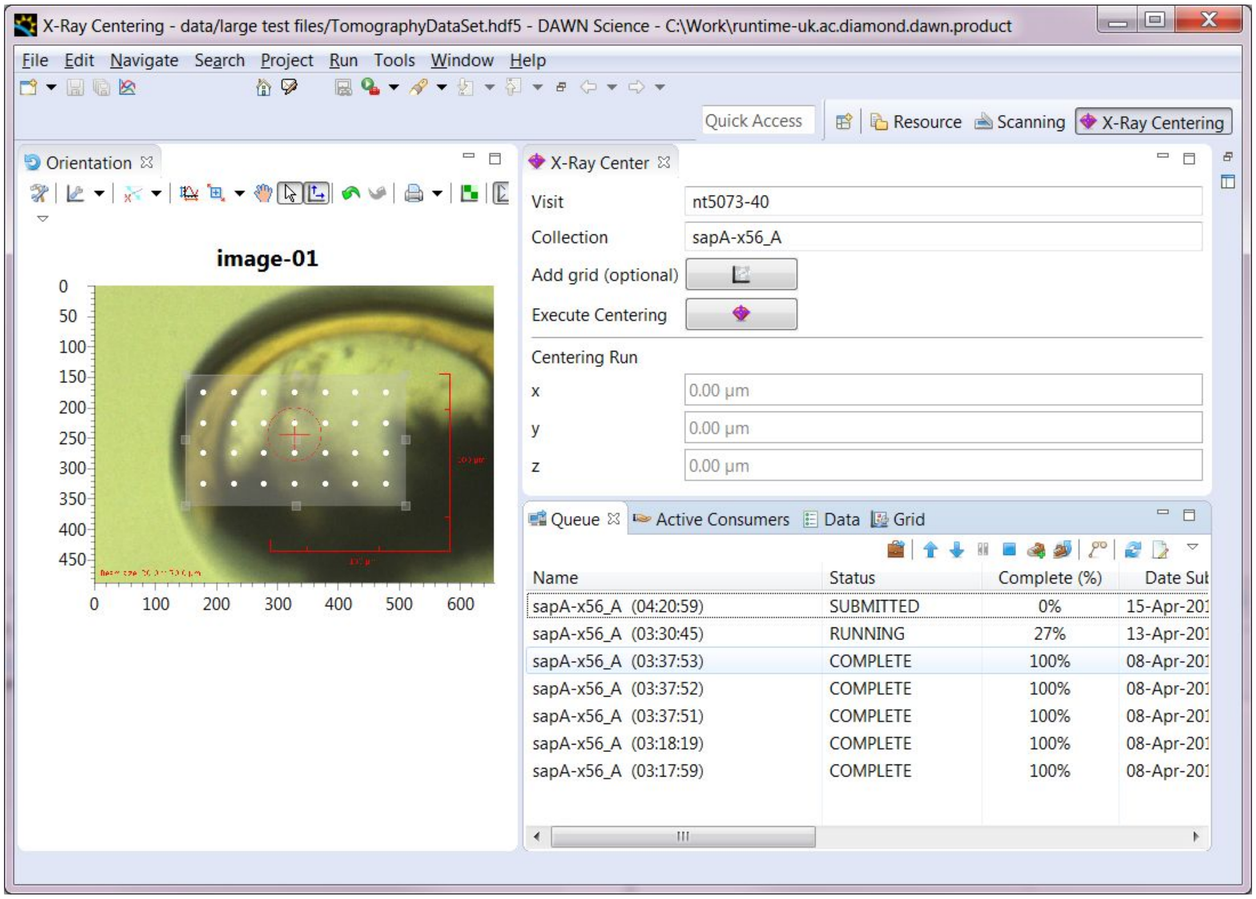
The project provides user interface parts for visualizing the queues of scans. It also provides a simple scan builder perspective, called ‘Scanning’, which allows a scan to be created and run. This perspective will work without configuration to the scanning project and execute mock devices provided in the project examples. The user interface allows a scan to be defined and configured with available devices and then submitted to the scanning service using the event service. This allows scans to be received on a server if one is implemented. This may be desirable if the client and acquisition are separate, for instance in the case of a remote thick client or a web client.


Scanning - Eclipse Platform
File Edit Navigate Search Project Run Window Help
Quick Access
Resource
X-Ray Centering
Scanning
Scan Editor
Detectors
Model 'Grid'
-0.25
0.00
0.25
0.50
0.75
1.00
0.00
0.25
0.50
0.75
1.00
1.25
/entry/mandelbrot_detector/stage_y
/entry/mandelbrot_detector/stage_x

Name	
Step Scan Solstice	
Grid Solstice	

Name	Value
Columns (fast)	8
Fast	stage_x
Rows (slow)	8
Slow	stage_y
Snake	✓

There is also be an example called 'X-Ray Centering' which shows a simple submit/consume using the scanning project. This example requires DAWN libraries to be available because it uses the DAWN plotting system as a service. In the future it may be desirable to remove this dependency as the dawnsci project intends to release plotting separately.


X-Ray Centering - data/large test files/TomographyDataSet.hdf5 - DAWN Science - C:\Work\runtime-uk.ac.diamond.dawn.product
File Edit Navigate Search Project Run Tools Window Help
Quick Access
Resource
Scanning
X-Ray Centering
Orientation
image-01
X-Ray Center
Add grid (optional)
Execute Centering
Queue
Active Consumers
Data
Grid

Visit	nt5073-40
Collection	sapA-x56_A

Centering Run

x	0.00 µm
y	0.00 µm
z	0.00 µm

Name	Status	Complete (%)	Date Sub
sapA-x56_A (04:20:59)	SUBMITTED	0%	15-Apr-201
sapA-x56_A (03:30:45)	RUNNING	27%	13-Apr-201
sapA-x56_A (03:37:53)	COMPLETE	100%	08-Apr-201
sapA-x56_A (03:37:52)	COMPLETE	100%	08-Apr-201
sapA-x56_A (03:37:51)	COMPLETE	100%	08-Apr-201
sapA-x56_A (03:18:19)	COMPLETE	100%	08-Apr-201
sapA-x56_A (03:17:59)	COMPLETE	100%	08-Apr-201

Scripting Layer

The scripting layer is intended to provide a python API which is easy to use and drives the runnable device service. It can submit scans to the scan queue or to be directly run. It uses a ScanRequest object on the java side of the service which is identical to the user interface. The doc from the mscan method is shown here which defines how it works.

Usage:

mscan(scan model(s), detector model(s))

A simple usage of this function is as follows:

```
> mscan(step(my_scannable, 0, 10, 1), det=mandelbrot(0.1))
```

The above invokation says "please perform a mapping scan over my scannable from 0 to 10 with step size 1, collecting data from the 'Mandelbrot' detector with an exposure time of 0.1 seconds at each step".

You can specify multiple detectors with a list (square brackets):

```
> mscan(..., det=[mandelbrot(0.1), another_detector(0.4)])
```

You can specify a scannable or list of scannables to monitor:

```
> mscan(..., mon=my_scannable, ...)  # or:
> mscan(..., mon=[my_scannable, another_scannable], ...)
```

You can embed one scan path inside another to create a compound scan path:

```
> mscan([step(s, 0, 10, 1), step(f, 1, 5, 1)], ...)
```

The above invocation says "for each point from 0 to 10 on my slow axis, do a scan from 1 to 5 on my fast axis". In fact, for the above case, a grid-type scan would be more idiomatic:

```
> mscan(grid(axes=(f, s), step=(1, 1), origin=(0, 0), size=(10, 4)), ...)
```

By default, this function will submit the scan request to a queue and return immediately. You may override this behaviour with the "now" and "block" keywords:

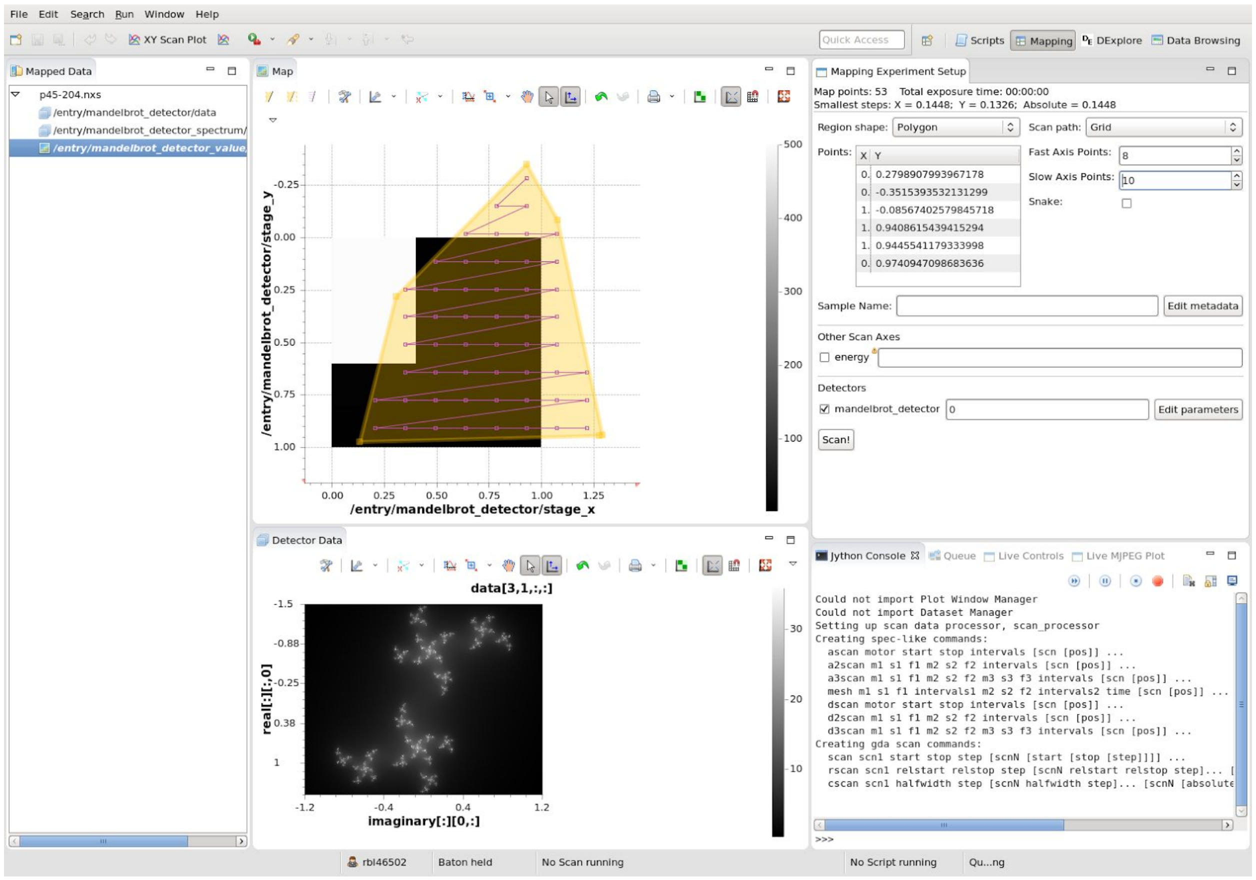
```
> # Don't return until the scan is complete.
> mscan(..., ..., block=True)


> # Skip the queue and run the scan now (but don't wait for completion).
> mscan(..., ..., now=True)


> # Skip the queue and return once the scan is complete.
> # For some beamlines now=True, block=True may need to be defaulted in localstation.
> mscan(..., ..., now=True, block=True)
```

NeXus Writing

It is important to ensure that writing of NeXus files is performant. Diamond Light Source have partially funded the new SWMR ("swimmer") functionality in HDF5 which ensures that one process may write to a file, quickly, while other processes read from the same binary file. This is HDF5 library version 2.10 and has been tested with the scanning project. The project has been designed for devices to write correct NeXus records by implementing a single interface i.e. it should be straightforward to add devices. Other file formats would be out of the scope of the initial phase of the project but would be desirable in the future depending on those willing to be involved with the project.

Current Deployment

The project is deployed at Diamond Light Source and is intended to be the backbone of the next generation of data acquisition at the facility. A screenshot of a user interface developed using the project is included below to show how scanning can be reused. The user interface is for mapping experiments, these experiments per-se are outside the scope of the scanning project and their user interfaces not included however the controller and model from scanning are used to deliver a mapping functionality at Diamond Light Source.



Mapping Experiment Setup

Map points: 53   Total exposure time: 00:00:00
Smallest steps: X = 0.1448; Y = 0.1326; Absolute = 0.1448

Region shape: Polygon   Scan path: Grid

Points:

X	Y
0.	0.2798907993967178
0.	-0.3515393532131299
1.	-0.08567402579845718
1.	0.9408615439415294
1.	0.9445541179333998
0.	0.9740947098683636

Fast Axis Points: 8
Slow Axis Points: 10
Snake: ☐

Sample Name:

Other Scan Axes
☐ energy

Detectors
☑ mandelbrot_detector 0

```
Could not import Plot Window Manager
Could not import Dataset Manager
Setting up scan data processor, scan_processor
Creating spec-like commands:
  ascan motor start stop intervals [scn [pos]] ...
  a2scan m1 s1 f1 m2 s2 f2 intervals [scn [pos]] ...
  a3scan m1 s1 f1 m2 s2 f2 m3 s3 f3 intervals [scn [pos]] ...
  mesh m1 s1 f1 intervals1 m2 s2 f2 intervals2 time [scn [pos]] ...
  dscan motor start stop intervals [scn [pos]] ...
  d2scan m1 s1 f1 m2 s2 f2 intervals [scn [pos]] ...
  d3scan m1 s1 f1 m2 s2 f2 m3 s3 f3 intervals [scn [pos]] ...
Creating gda scan commands:
  scan scn1 start stop step [scnN [start [stop [step]]]] ...
  rscan scn1 relstart relstop step [scnN relstart relstop step]... [
  cscan scn1 halfwidth step [scnN halfwidth step]... [scnN [absolute
>>>
```

Example of polygonal scanning in a mapping experiment

Why Here?:

The Eclipse Foundation is the right place to collaborate for scanning because of the Science Working Group. This group has attracted several universities and software companies. The Eclipse Foundation gives the most opportunity for discovering new projects that scanning can make use of and add value to the scientists working at or visiting Diamond Light Source.

Licenses:

Eclipse Public License 1.0

Legal Issues:

Currently the project has dependencies from two known non-eclipse sources, some apache ones (in orbit) and some other eclipse projects

1. HDF5 - which has been checked by the foundation but not yet passed with Eclipse legal team. It has a home grown license https://www.hdfgroup.org/ftp/HDF5/releases/hdf5-1.10/hdf5-1.10.0/src/unpacked/COPYING
2. jeromq - This has an LGPL license. We hope to remove this dependency or abstract it out of the project.
3. EPICS - Another home grown license http://www.aps.anl.gov/epics/license/open.php We will ask the EPICs organization to duel license the part we need Apache.

Other dependencies are Mars, Orbit bundles, apache with existing CQs, eclipse-dawnsci eclipse-january and eclipse-richbeans.

Project Scheduling:

Scanning will have releases scheduled compatible with the shutdown phases of the facilities adopting the project, to be negotiated with the committers. If many facilities adopt the project, it may better serve them to release on the Eclipse release train which is independent. This should be decided once the committers are engaged with the incubation project.

Future Work:

Initial contribution 2016, first releases 2017.

In future, we plan to add new scanning algorithms to allow automation not only of mapping experiments but of macromolecular crystallography. In particular a new technique called **VMXi**.

We would like to present the project at conferences, specifically **NoBUGS** and **icalepcs**.

HIDEPEOPLE

Project Leads:

Matthew Gerring

Committers:

James Mudd

Keith Ralphs

Eric Berryman

Robert Walton

Mark Booth

Mentors:

Jay Billings

Wayne Beaton

Interested Parties:

Eclipse Science Working Group members, science.eclipse.org.

Several facilities using EPICS at icalepcs 2015 conference. I will approach these facilities directly to suggest committers for the project.

HIDESOURCE CODE

Initial Contribution:

The current bundles are available on github at https://github.com/DiamondLightSource/daq-eclipse. (Those starting with uk.ac.diamond will not be part of the initial contribution if you are looking on there at the current state.)

org.eclipse.scanning.example.feature

org.eclipse.scanning.feature

org.eclipse.scanning.malcolm.feature

org.eclipse.scanning.releng

org.eclipse.scanning.repository

org.eclipse.scanning.target.platform

org.eclipse.scanning.ui.feature

org.eclipse.scanning.api

org.eclipse.scanning.command

org.eclipse.scanning.event

org.eclipse.scanning.event.ui

org.eclipse.scanning.example

org.eclipse.scanning.example.xcen

org.eclipse.scanning.example.xcen.test

org.eclipse.scanning.example.xcen.ui

org.eclipse.scanning.malcolm.core

org.eclipse.scanning.points

org.eclipse.scanning.sequencer

org.eclipse.scanning.server

org.eclipse.scanning.test

Source Repository Type:

GitHub

Source Repositories:

https://github.com/DiamondLightSource/daq-eclipse