GΦL
Global Phasing Limited

# MXLIMS
# Working on a prototype

Global Phasing

MXCuBE/ISPyB Meeting, Trieste, November 2024

GΦL
Global Phasing Limited

# Introduction

Where we are and how we got there

Global Phasing Limited

# MXLIMS starting point (ALBA, end 2023)

- The ISPyB collaboration was disintegrating
  - There was never that much shared in ISPyB except for the database core
  - Moves towards sharing technology and back ends (PyISPyB) were not panning out, and major sites were looking at their own alternatives
- There was a need for supporting multiple LIMS systems
  - E.g. MXCuBE now also included non-ISPyB participants
- Various unmet needs:
  - Support for multi-sweep experiments was stymied by the rigidity of the ISPyB system
  - A lack of uniform, well-defined access at different synchrotrons for sample shipping and fetching home results.

Global Phasing Limited

# High hopes for MXLIMS

- A precise, documented data model for crystallography **scientific content**
- Acceptance as a shared API for the ISPyB collaboration
  - Replacing the SQL tables as the central point of the collaboration
  - Development by merging already existing data models
- Harmonisation with internal data models for LIMS systems
  - particularly the new and exciting developments at the ESRF

Global Phasing Limited

# MXLIMS purpose (MAX IV, mid-2024)

- Scientific API for communication with LIMS and beamlines
  - Message specification and data model
  - Usable across multiple Synchrotrons, beamlines and LIMS systems
  - Detailed and flexible enough to handle all traffic
  - Extensible to allow for additional site- and program- specific data
    - Each site and program gets its own namespace to define its own extension schemas
- Scope: MX and related techniques
  - Extension to SSX contemplated

Global Phasing Limited

# MXLIMS Implementation (MAX IV, mid-2024)

- Multiple messages defined as JSON schemas

- Simple core structure, to allow storage in Mongo-DB or data-catalogue type systems
  - Few core types, with most data as variable metadata / parameters
  - Parameters in JSON schemas that are specific to one kind of data; core model can be applied to multiple techniques
  - JSON Schemas can be nested

- *No* direct application to LIMS system implementation
  - MXLIMS defines data and the structure and level of complexity required
  - Data transfer to LIMS (or beamline control) systems via converters
  - MXLIMS does not limit implementations or program internal data models

Global Phasing Limited

# Processes and sources

- MXLIMS can be found at https://github.com/rhfogh/mxlims_data_model
- Developed through working group meetings, with most detailed modeling work done by Ed Daniel and RH Fogh
- Main sources of inspiration:
  - mmCIF, IceBear data model, ICAT, MXCuBE queue model with workflows, autoPROC multisweep data processing, working group discussions on use cases for SSX and complex samples.

GΦL
Global Phasing Limited

# Model core

The basic structure that the specific Metadata fit into

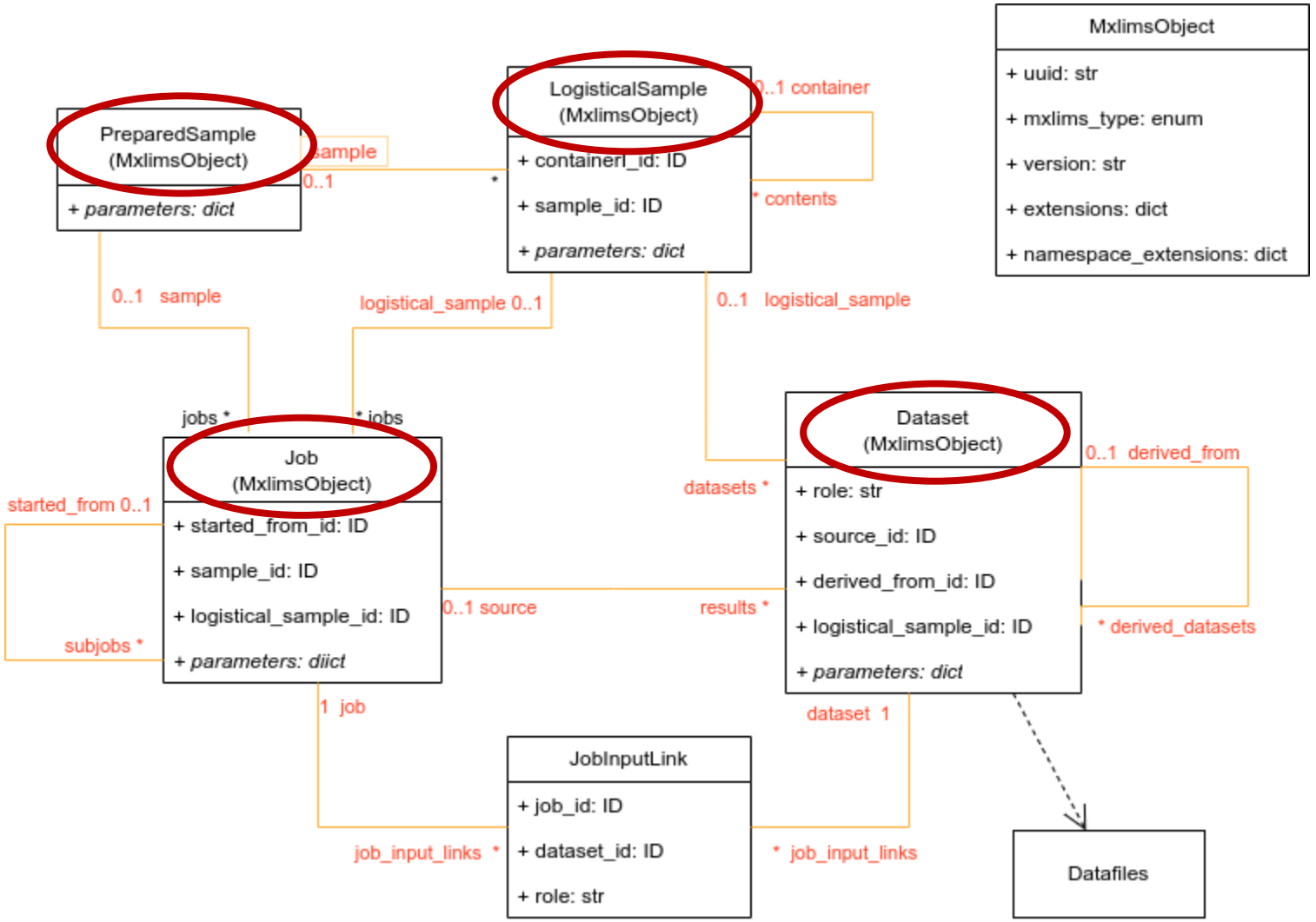
Global Phasing Limited

# Core model

Four core object types with links between them determined by foreign keys. Together they support provenance tracking.

Object *mxlims_type* (e.g. CollectionSweep, ReflectionSet) determines the actual schema, with the allowed *parameters* and types of linked-to objects.

The *version* allows for schema changes, and the *extensions* give room for site-specific additions

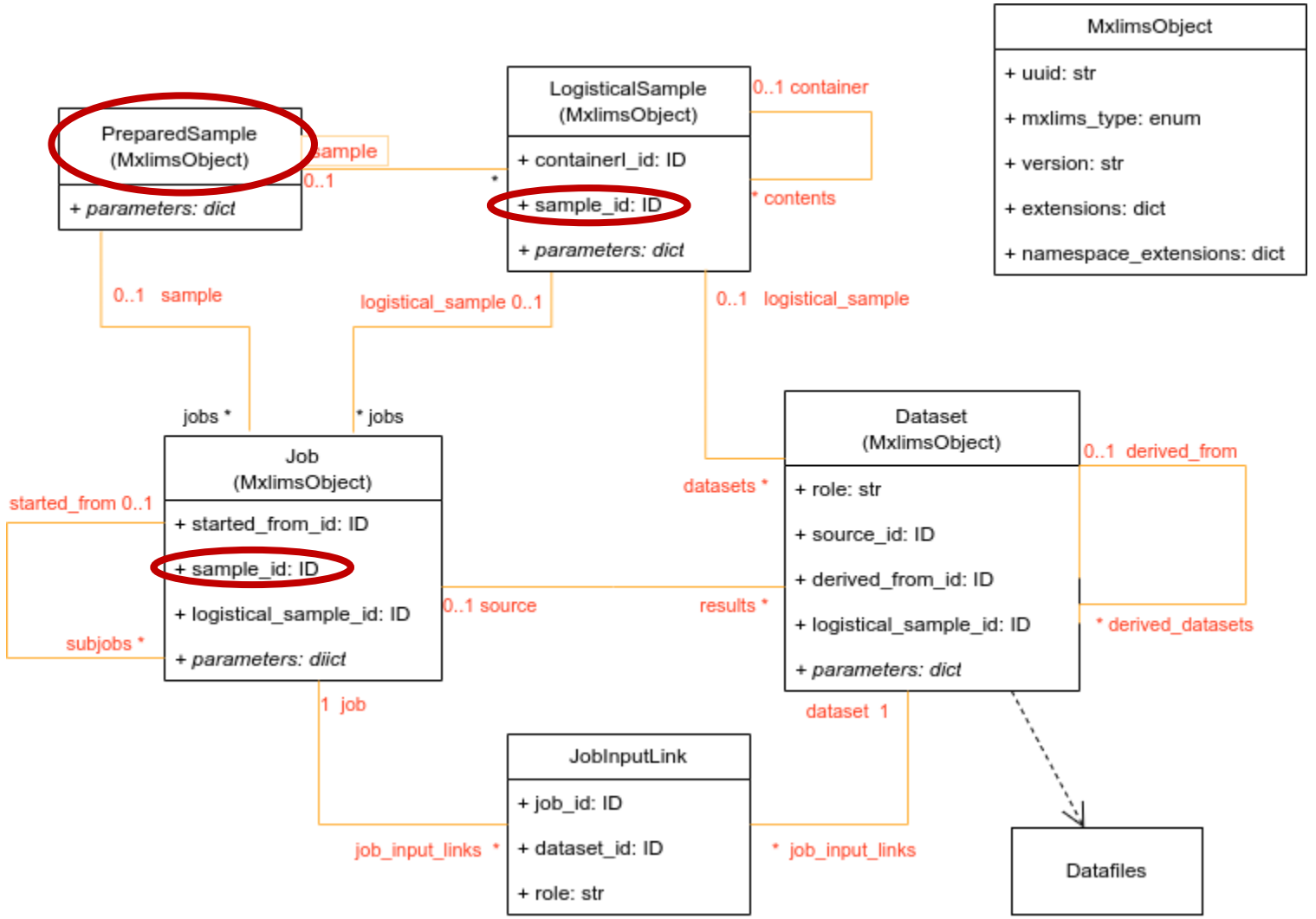


Global Phasing Limited

# Core model – PreparedSample

## PreparedSample type

Describes the sample preparation, (crystal, well, or SSX slurry), with macromolecule, other molecules, expected crystal spacegroup and characteristics, and lot identifiers.

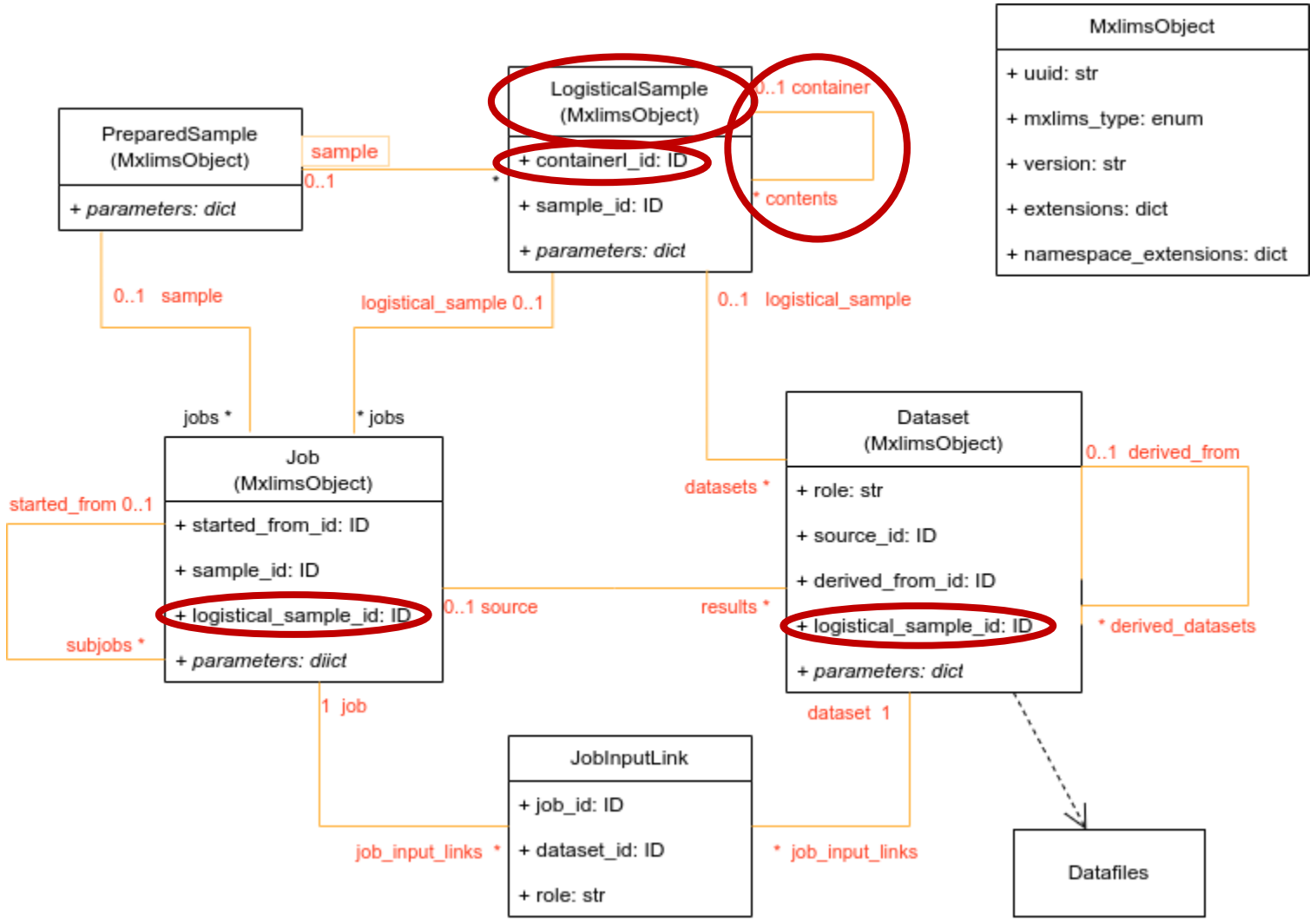


Global Phasing Limited

# Core model – Logistical_sample

## LogisticalSample type

A sample that is mounted for an experiment (Pin, grid, slurry flow), a (nested) container containing other samples, or a crystal or location where data are measured. Could be a Shipment, Plate, Dewar, Grid, Puck, Crystal, Location, …, each of which would have its own subtype schema



Global Phasing Limited

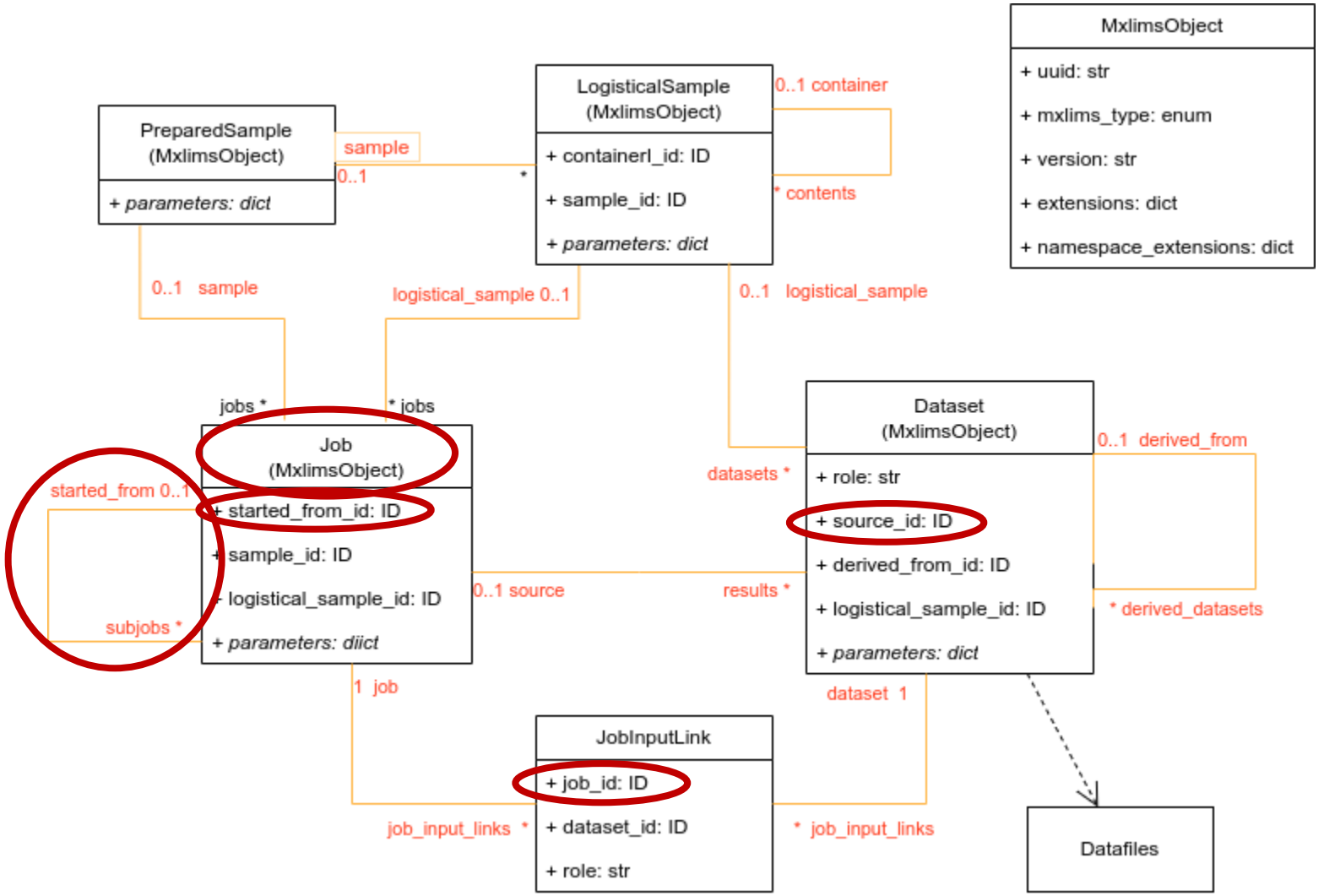
# Core model - Job

## Job type

A physical experiment or calculation that produces Datasets.

Jobs can start other subjobs, allowing a workflow job to start e.g. Xray-centring jobs or processing jobs.

Job input is handled by names many-to-many links defined in type schemas, using the JobInputLink table



Global Phasing Limited

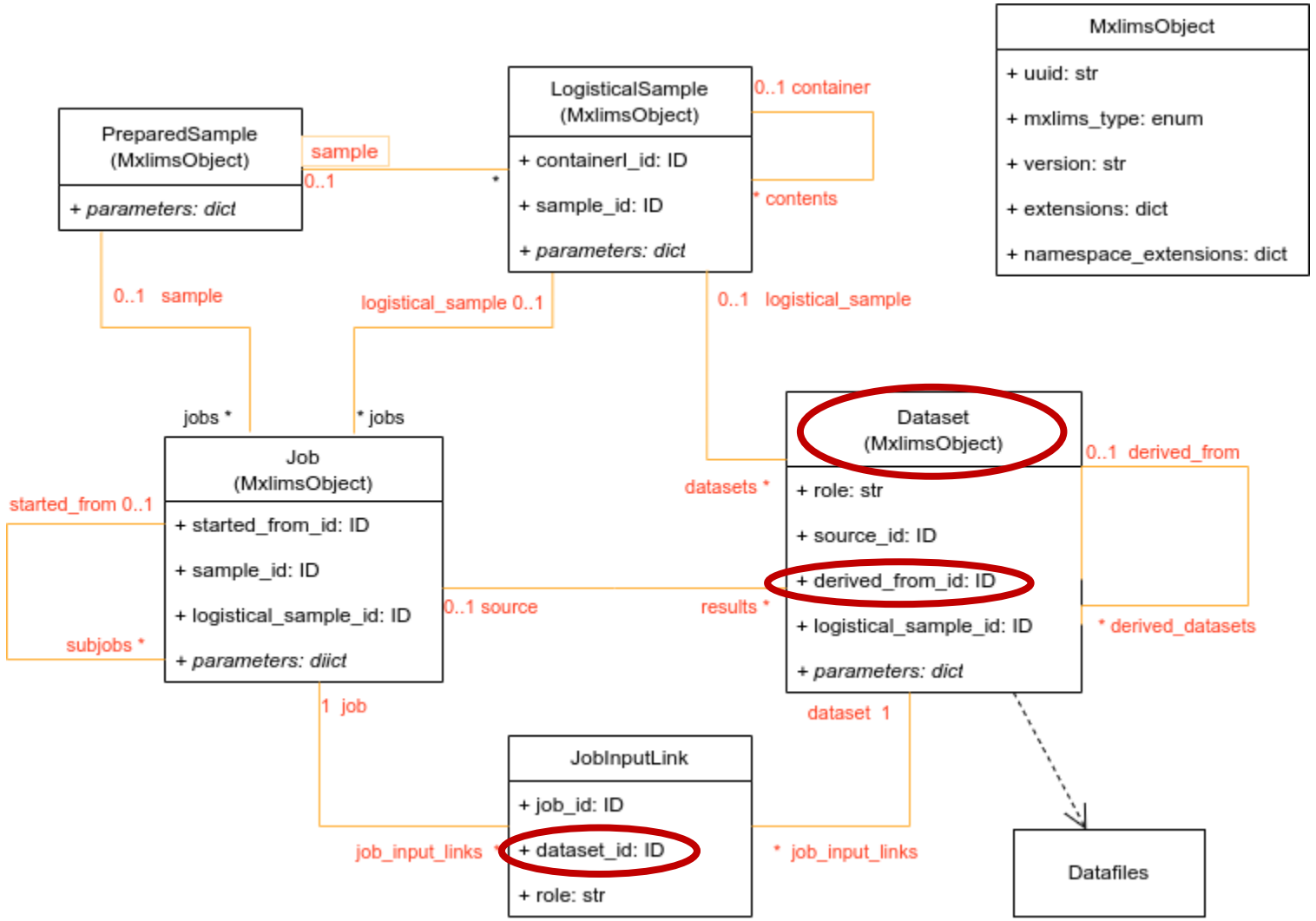
# Core model - Dataset

## Dataset type

A cohesive set of data, treated as a unit.

Datasets are created by source jobs, or can be derived from other Datasets, e.g. when removing empty images from an image set.

The same schemas are used for diffraction plans, parameter templates, or acquisition queue input (without files) and for results and processing input (with files)



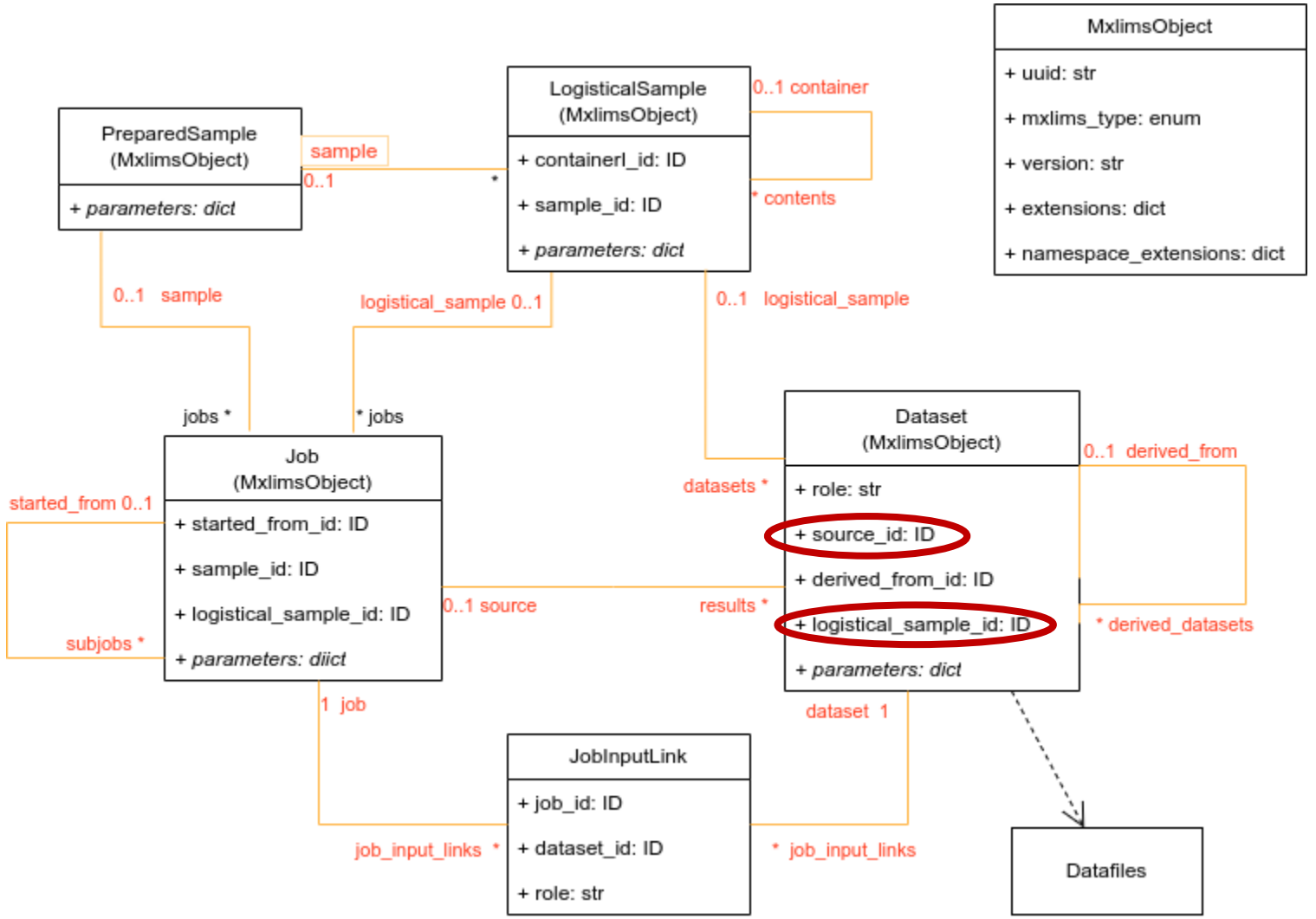
Global Phasing Limited

# Core model – ESRF matches

This structure matches some of the ID attributes that are passed from MXCuBE to ISPyB-DRAC.

Source_id matches workflow_uid

Logistical_sample_id matches location_id



GΦL
Global Phasing Limited

# Implementations and prototypes

JSON schemas and prototype applications

New developments since last meeting

GΦL
Global Phasing Limited

# Implementation of MX model

- The model has been coded in Pydantic
- JSON Schemas are exported from Pydantic
- Schema documentation made with json-schema-for-humans
- Data load/save (to Python) comes with Pydantic
- The result is checked in to https://github.com/rhfogh/mxlims_data_model/tree/rhfogh_develop (no pull request yet)

Global Phasing Limited

# MXLIMS export from MXCuBE

- An MXLIMS exporter for GPhL workflows and for single-sweep data collection has been built and tested
  - Support for complete MXLIMS MX model, including wavelength-interleaved MAD workflow experiments
  - The exporters works from within QueueEntries, so are not confined within GPhL code
  - Simple to code, once you have the Pydantic model.
- Clemens Vonrhein added JavaScript and html to build a display of the result.
  - All in the last couple of weeks

Global Phasing Limited

# JSON Schema - LogisticalSample

JSON Schema documentation, generated from the schema using json-schema-for-humans

The LogisticalSample class is the base class – the specific classes for Dewars, crystals, etc. have been modelled separately by Ed Daniel, but not yet merged with this part of the model

**Type: object**

Base class for MXLIMS Logistical Samples

describing Sample containers and locations
(from Dewars and Plates to drops, pins and crystals)

- mxlims_type
- version
- uuid
- extensions
- namespace_extensions
- sample_ref
- container_ref
- contents
- job_refs
- dataset_refs

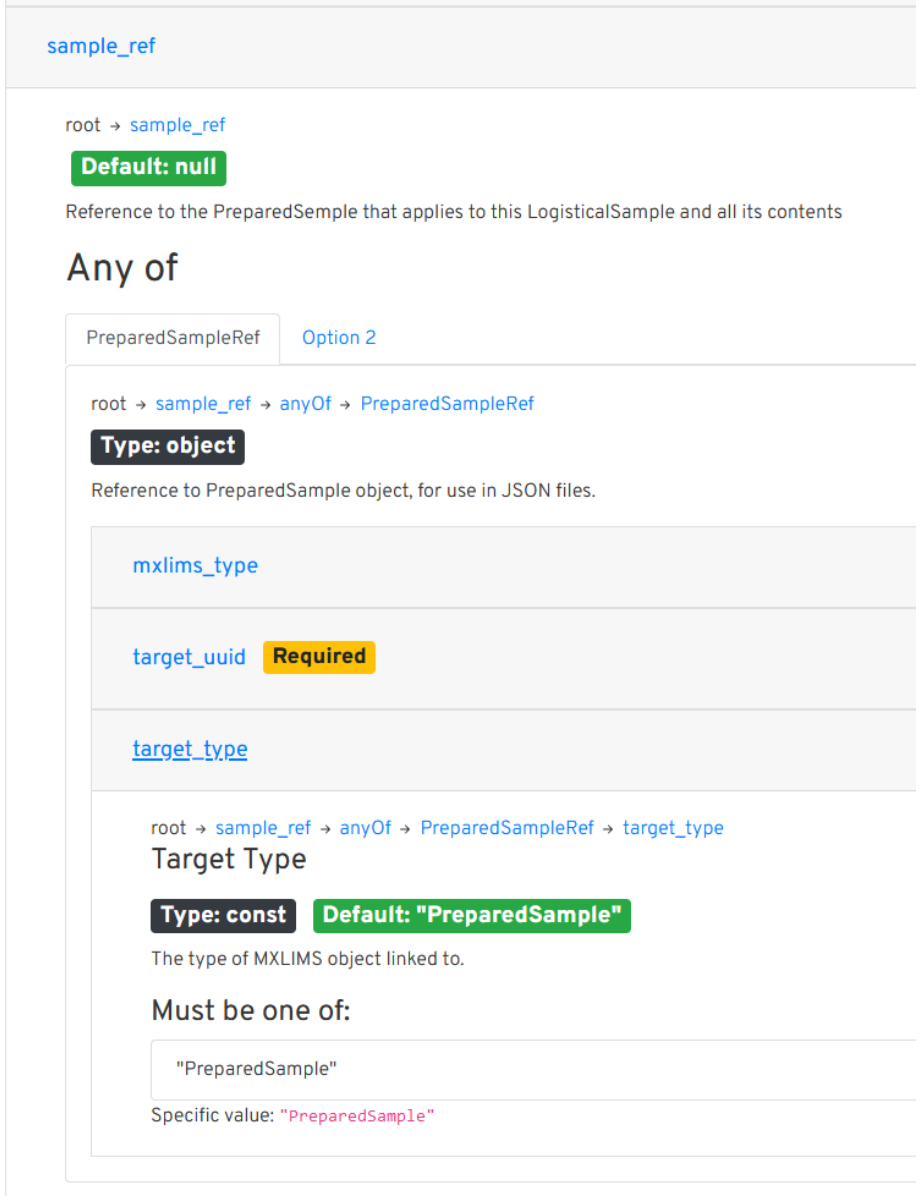
Global Phasing Limited

# JSON Schema – LogisticalSample detail

Schema documentation with one field expanded.

The sample_ref field shows the way cross-links between objects are stored.

The PreparedSampleRef fields mxlims_type and target_type have fixed values – only the target_uuid varies.



Global Phasing Limited

# Example part 1 – interleaved MAD

## Experiment record

Interleaved MAD experiment, mock workflow acquisition with simulation of individual images

Workflow produced nine sweeps, including characterisation.

Centrings are not recorded.

**MXExperiment_4k4k.json**

| | |
|---|---|
| mxlims_type | MXExperiment |
| version | 0.2.4 |
| uuid | bd3c2780-c126-4320-b7fd-21f25784a9c1 |
| ▸ sample | |
| ▸ logistical_sample | |
| ▾ results | |
| ▸ 0 | |
| ▸ 1 | |
| ▸ 2 | |
| ▸ 3 | |
| ▸ 4 | |
| ▸ 5 | |
| ▸ 6 | |
| ▸ 7 | |
| ▸ 8 | |
| start_time | 2024-11-12T20:18:23.230882 |
| end_time | 2024-11-12T20:26:11.941338 |
| expected_resolution | 1.033 |
| measured_flux | 229906068483.64087 |

# Example part 2 - CollectionSweep

CollectionSweep with parameters

The connection to the source job is stored as a reference record

▼ results

▶ 0

▼ 1

| | |
|---|---|
| mxlims_type | CollectionSweep |
| version | 0.2.4 |
| uuid | 7d2eba16-9ef1-4fc8-a112-bae1f718093c |
| ▼ source_ref | |
| mxlims_type | MxlimsObjectRef |
| target_uuid | bd3c2780-c126-4320-b7fd-21f25784a9c1 |
| target_type | MXExperiment |
| role | Result |
| ▶ logistical_sample_ref | |
| exposure_time | 0.04 |
| image_width | 0.2 |
| overlap | 0 |
| number_triggers | 0 |
| number_images_per_trigger | 0 |
| energy | 12.4 |
| transmission | 100 |
| resolution | 3.5489723032023557 |
| detector_roi_mode | 0 |
| ▶ beam_position | |
| ▶ beam_size | |
| beam_shape | rectangular |
| ▶ axis_positions_start | |
| ▶ axis_positions_end | |
| scan_axis | omega |
| ▶ scans | |
| file_type | cbf |
| prefix | 4k4k_G1B1 |
| filename_template | 4k4k_G1B1_6_%05d.cbf |
| path | /alt/rhfogh/calc/mxcube_data/visitor/idtest0/eh1/20241112/RAW_DATA/4k4k_001/main |

▶ 2

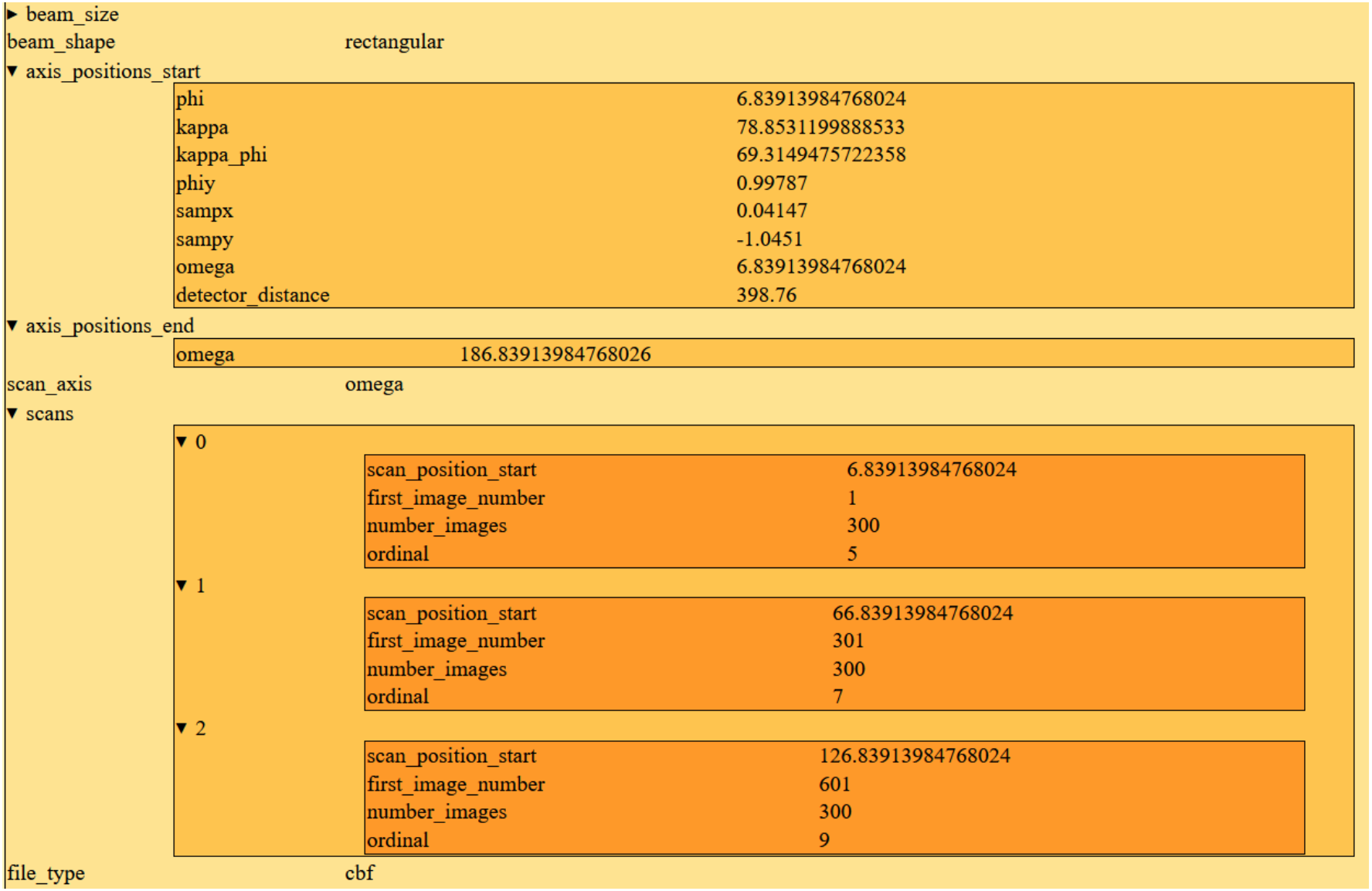
Global Phasing Limited

# Example part 3 – sweeps and scans

The sweep is a range of images with start and end positions.

It is divided into scans, that may be acquired interleaved (as here) or out of order.

A sweep matches an MXExpress orientation_id



Global Phasing Limited

# MXLIMS exporter from mmCIF

- Clemens Vonrhein has built an exporter that extracts Table1 data from mmCIF to MXLIMS.
- With additional Javascript and html this gives a nice viewable web page
- This information is available for *all* PDB entries at https://staraniso.globalphasing.org/table1/

Global Phasing Limited

# Table1 for 9CGQ

Table1 view for a recent PDB entry, generated via extracted MXLIMS JSON data.

The entire system required less than two weeks' work to produce.

Data quality metrics extracted from 9cqg.cif.gz by **aB_cif2table1** from BUSTER (Global Phasing Ltd.).

▸ Raw JSON data

**Table 1 generated from MXLIMS-JSON (via JavaScript):**

| | Overall | InnerShell | OuterShell |
|---|---|---|---|
| Low resolution limit [Å] | 116.500 | 6.992 | 2.709 |
| High resolution limit [Å] | 2.474 | 5.559 | 2.474 |
| R(merge) | 0.0852 | 0.0538 | 0.7860 |
| R(meas) | 0.1056 | 0.0676 | 0.9875 |
| R(pim) | 0.0614 | 0.0402 | 0.5899 |
| CC(1/2) | 0.995 | 0.992 | 0.455 |
| CC(ano) | -0.099 | - | - |
| \|DANO\|/σ(DANO) | 0.791 | - | - |
| Completeness (spherical) [%] | 69.4 | - | - |
| Completeness (ellipsoidal) [%] | 90.9 | - | - |
| Multiplicity | 2.6 | 2.4 | 2.6 |
| Anomalous completeness (spherical) [%] | 62.1 | - | - |
| Anomalous completeness (ellipsoidal) [%] | 81.4 | - | - |
| Anomalous multiplicity | 1.4 | - | - |

Global Phasing Limited

# 9CGQ - detail

Resolution bin data for 9CGQm extracted from mmCIF via MXLIMS.

**Binned data generated from MXLIMS-JSON (via JavaScript):**

| Resolution [Å] | R(merge) | R(meas) | R(pim) | CC(1/2) | Multiplicity |
|---|---|---|---|---|---|
| 116.500 - 6.992 | 0.0383 | 0.0474 | 0.0273 | 0.996 | 2.8 |
| 6.992 - 5.559 | 0.0538 | 0.0676 | 0.0402 | 0.992 | 2.4 |
| 5.559 - 4.867 | 0.0564 | 0.0707 | 0.0419 | 0.991 | 2.4 |
| 4.867 - 4.432 | 0.0563 | 0.0701 | 0.0411 | 0.991 | 2.6 |
| 4.432 - 4.128 | 0.0643 | 0.0796 | 0.0462 | 0.991 | 2.7 |
| 4.128 - 3.902 | 0.0776 | 0.0965 | 0.0563 | 0.988 | 2.7 |
| 3.902 - 3.724 | 0.0956 | 0.1184 | 0.0686 | 0.986 | 2.7 |
| 3.724 - 3.553 | 0.1239 | 0.1529 | 0.0881 | 0.980 | 2.7 |
| 3.553 - 3.436 | 0.1547 | 0.1902 | 0.1089 | 0.967 | 2.8 |
| 3.436 - 3.334 | 0.1951 | 0.2400 | 0.1375 | 0.954 | 2.8 |
| 3.334 - 3.240 | 0.2525 | 0.3133 | 0.1824 | 0.913 | 2.5 |
| 3.240 - 3.157 | 0.3137 | 0.3892 | 0.2266 | 0.884 | 2.5 |
| 3.157 - 3.083 | 0.4032 | 0.4979 | 0.2874 | 0.811 | 2.6 |
| 3.083 - 3.014 | 0.4503 | 0.5594 | 0.3262 | 0.766 | 2.5 |
| 3.014 - 2.950 | 0.5110 | 0.6340 | 0.3690 | 0.728 | 2.5 |
| 2.950 - 2.892 | 0.6407 | 0.7885 | 0.4524 | 0.640 | 2.7 |
| 2.892 - 2.839 | 0.6866 | 0.8450 | 0.4850 | 0.602 | 2.7 |
| 2.839 - 2.783 | 0.7059 | 0.8660 | 0.4942 | 0.569 | 2.8 |
| 2.783 - 2.709 | 0.7203 | 0.8926 | 0.5198 | 0.535 | 2.7 |
| 2.709 - 2.474 | 0.7860 | 0.9875 | 0.5899 | 0.455 | 2.6 |

GΦL
Global Phasing Limited

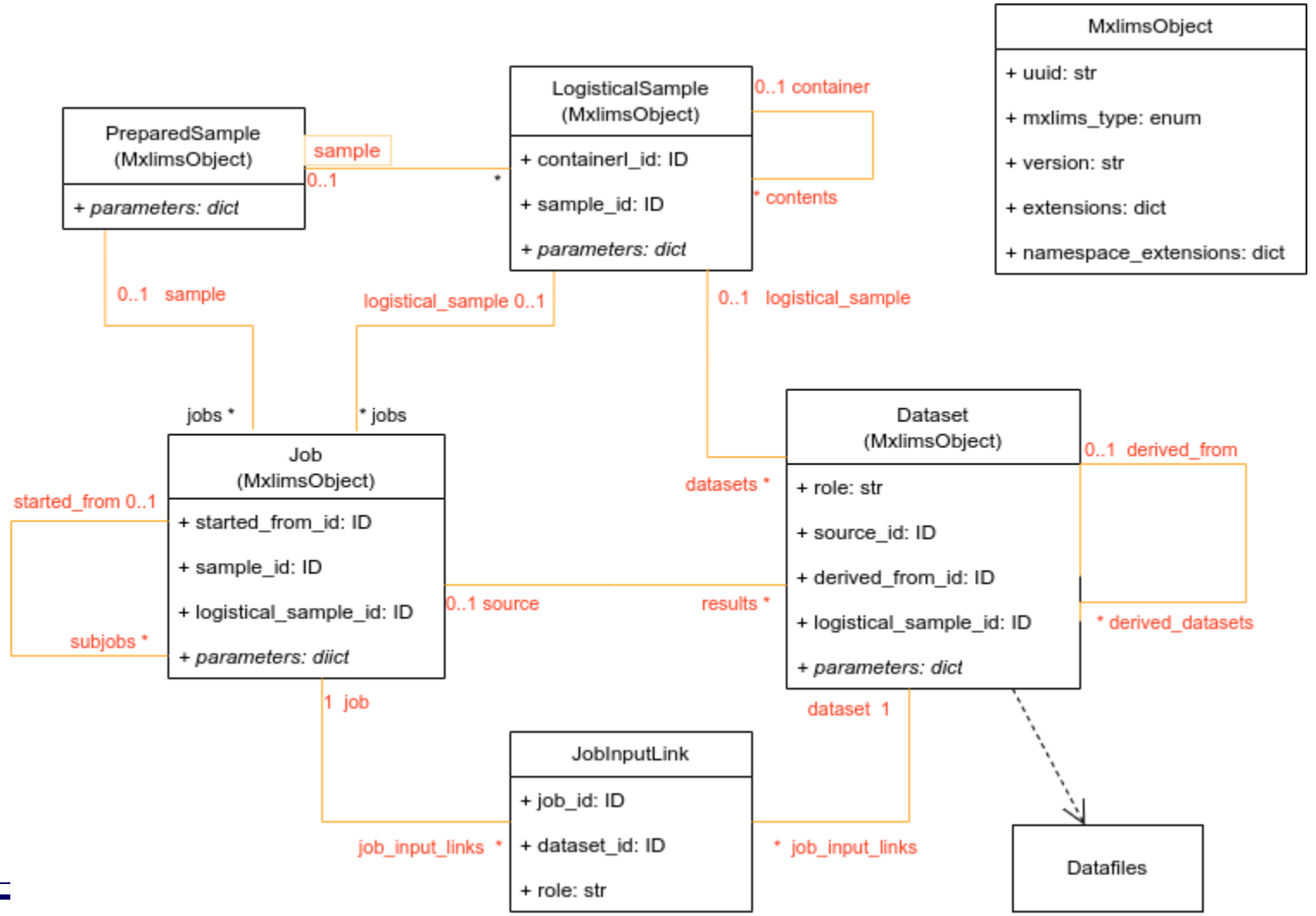
# A technical issue

How do you best implement the MXLIMS model either in code or as JSON?

Global Phasing Limited

# Core model - again

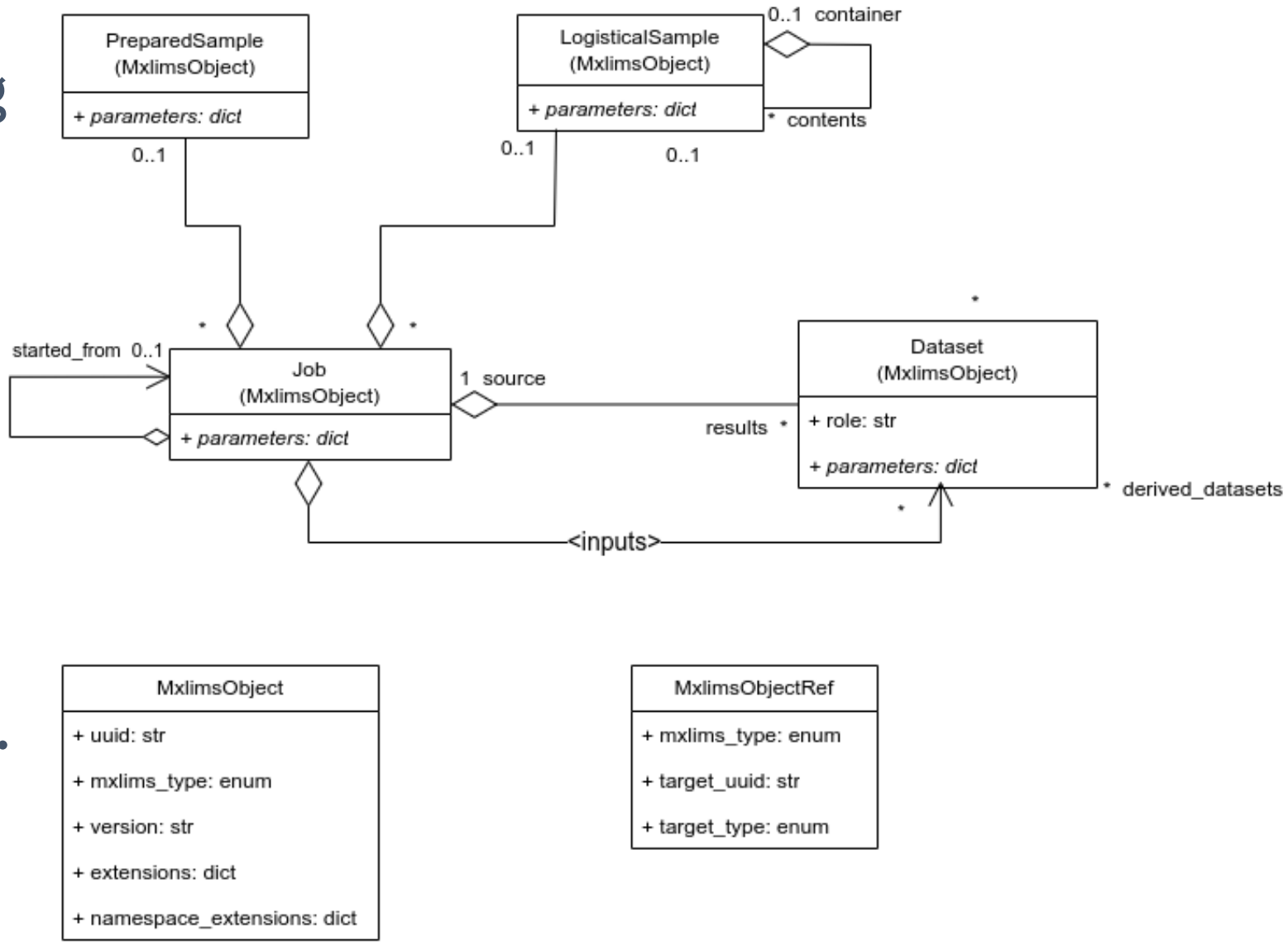
- The core model describes a *network* of objects, with two-way links and loops. This is necessary to store provenance properly. In a database this can be navigated as desired.
- JSON is limited to a *tree* structure; how should you map the model onto JSON?



Global Phasing Limited

# One-Job JSON Schema

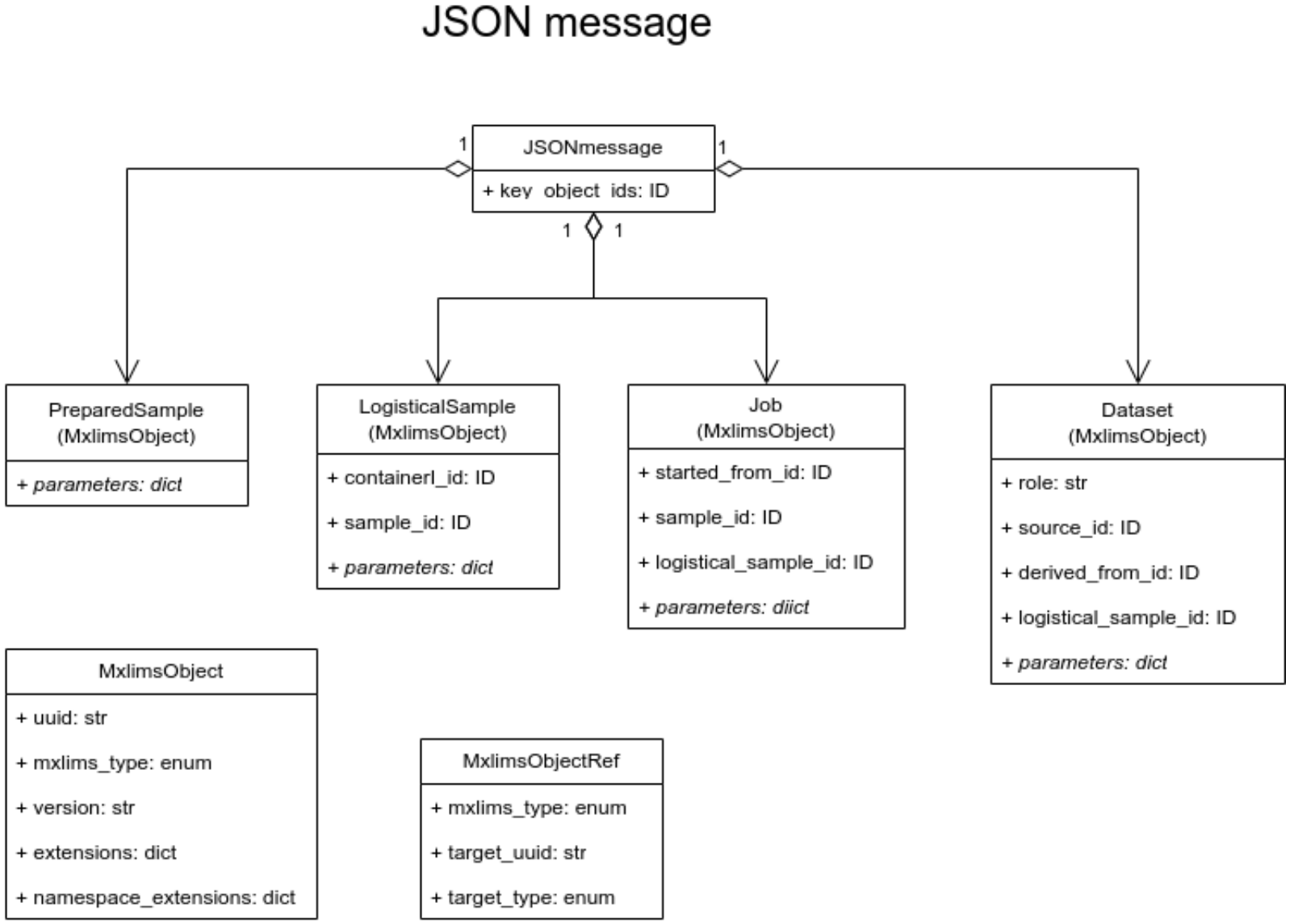
- A JSON tree rooted in a single Job, with everything else stored as JSON content.
- Crosslinks are stored as MxlimsObjectRef records
- For a processing job processing data from a single experiment job, the ‘started_from’ link allows you to fit all relevant data into a single schema (just).



Global Phasing Limited

# Object network JSON schema

- A (non-MXLIMS) JSON root with separate lists of Jobs, Datasets, PreparedSamples, and LogisticalSamples.
- All links are handled with MxlimsObjectRefs
- This can handle all object combinations, but is less intuitive and requires software to reconnect all the links.



GΦL
Global Phasing Limited

# Information needed

- What kinds of JSON schemas do people want to work with? One obvious addition would be a ‘shipment’ schema with a single LogisticalSample as root and everything else as content.
- We could have multiple schemas with different combinations of the same building blocks, but what are the use cases?
- How would one implement this internally in e.g. MXCuBE (using e.g. Pydantic? Something more complex?) if the data are sometimes too complex for simple trees?

GΦL
Global Phasing Limited

# Acknowledgements

Thanks to the many, many people

who have participated in the MXLIMS working group,

or contributed information, discussions and feedback.