ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEMPR001

DATA ANALYSIS INFRASTRUCTURE FOR DIAMOND LIGHT SOURCE MACROMOLECULAR & CHEMICAL CRYSTALLOGRAPHY AND BEYOND

M. Gerstel†, A. W. Ashton, R. J. Gildea, K. E. Levik, G. Winter
Diamond Light Source, Oxfordshire, United Kingdom

Abstract

The Diamond Light Source data analysis infrastructure, Zocalo, is built on a messaging framework. Analysis tasks are processed by a scalable pool of workers running on cluster nodes. Results can be written to a common file system, sent to another worker for further downstream processing and/or streamed to a LIMS (Laboratory Information Management System). Zocalo allows increased parallelization of computationally expensive tasks and makes the use of computational resources more efficient. The infrastructure is low-latency, fault-tolerant, and allows for highly dynamic data processing. Moving away from static workflows expressed in shell scripts we can easily re-trigger processing tasks in the event that an issue is found. It allows users to re-run tasks with additional input and ensures that automatically and manually triggered processing results are treated equally. Zocalo was originally conceived to cope with the additional demand on infrastructure by the introduction of Eiger detectors with up to 18 Mpixels and running at up to 560 Hz framerate on single crystal diffraction beamlines. We are now adapting Zocalo to manage processing tasks for ptychography, tomography, cryo-EM, and serial crystallography workloads.

INTRODUCTION

Data collected at single crystal diffraction beamlines are processed automatically at Diamond Light Source (DLS) [1]. These experiments generally involve the generation of a substantial amount of data. For a typical data collection at a macromolecular (MX) beamline with a DECTRIS Pilatus 6M detector a crystal is rotated through 360° while being exposed to X-rays. At each oscillation step of 0.1° an image is read out from the detector, resulting in 3,600 6 MB images (21 GB). Depending on the beamline parameters and equipment these data may be collected in 36 seconds (100 images/s) to 2.4 minutes (25 images/s). A different type of preparatory experiment is a grid scan (Fig. 1), for which usually fewer than 1,000 still images are obtained within 1-2 minutes across a sample area to locate diffracting material. Since users may often wait for initial analysis results before deciding on how to proceed with their experiment, the time to process the experimental data is critical to the overall facility efficiency. Technological advances work against the requirement to provide speedy feedback to the experimenters: DLS recently installed DECTRIS Eiger2 XE 16M detectors on beamlines I03 and I04 and an Eiger 2 X 4M on beamline VMXi. These considerably increase both the size of individual images as well as achievable frame rates. While the data processing infrastructure at DLS has provided reliable service in the past, these technical developments as well as the launch of two new MX beamlines, VMXi and VMXm, required a major overhaul to ensure smooth data processing for the future.

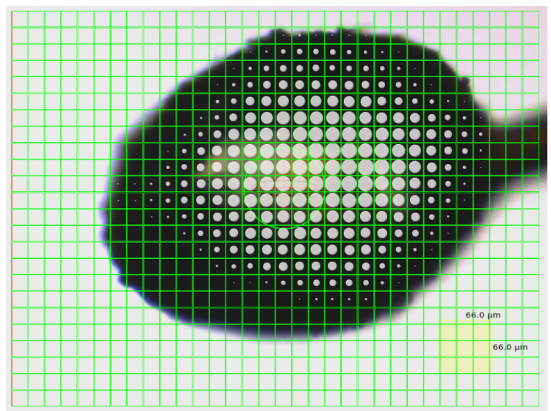

Figure 1: Results of grid scan per-image analysis overlayed with optical image of sample. Circles indicate presence of diffracting material.

To achieve this we implemented a distributed infrastructure called ‘Zocalo’. Zocalo is built on a messaging framework where fine-grained tasks are submitted to a queue, picked up and processed by a flexible number of specialist services, which run on high-performance cluster nodes. Services can be slotted together to form a larger processing pipeline, the results of each step can be stored on the file system, sent to a downstream processing service, or both. This provides many advantages over the previous analysis system, including low-latency data processing, self-monitoring, automatic resource allocation, fault-tolerance and more efficient use of computational resources.

There are many elements to full automated data processing of diffraction data, such as strategy calculations, per-image analysis, data reduction, experimental phasing, molecular replacement, and difference map calculation. In this paper two tasks will be highlighted to demonstrate how they are implemented in the new processing infrastructure: the per-image analysis, which is central to grid scans, and the initial data reduction for data collections. Both are currently run for every macromolecular (MX) and chemical crystallography (CX) data collection at DLS, and they pose different and representative challenges in data processing.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEMPR001

ZOCALO – A NEW DATA PROCESSING FRAMEWORK

Previously, the data analysis pipeline at Diamond was controlled by a number of bash scripts that were executed by the Generic Data Acquisition (GDA) program [2]. These scripts could launch autoprocessing tasks that were run while the data are being collected or initiate tasks after the data collection was complete. In either case the scripts would interact with the ISPyB database [3] both to determine information about the data collection in order to decide what autoprocessing tasks should be performed, and then to insert the results of the autoprocessing into ISPyB so that they could be displayed to the user, e.g. via SynchWeb [4]. The actual processing took place on one of the DLS cluster nodes by scheduling a batch job. Depending on the job type and the current load on the cluster the job could run immediately or had to wait in a job queue. Communication between the processing job and the user generally only happened by writing results into the ISPyB database. For grid scans a UDP messaging protocol was used to notify GDA to look for new entries in the database.

With Zocalo the DLS network file system and the ISPyB database are still used for durable storage of experimental data and meta-data respectively. At the centre of Zocalo stands an Apache ActiveMQ messaging server, also called a 'broker', which passes ephemeral data between interested parties in an events-based fashion. A messaging server provides a low level, content-agnostic way of routing self-contained blocks of data from one or many senders to one or many recipients. Various message passing semantics are possible: messages can be broadcast, sent to exactly one recipient, stored until such a recipient appears, and even retransmitted to the same or different recipient in case a recipient did not confirm that it received and processed the message.

With a messaging framework a large number of interacting services can be connected easily and reliably, Fig. 2. For example a file monitoring service can observe files arriving on disk and notify a spot finding service. Neither service needs to know exactly on which machine the other service is running, or in fact how many instances are running. It is possible to observe the messages being passed from a processing status viewer, and still guarantee via the message passing semantics that every message is received and processed by exactly one spot finding service.

Figure 2: ActiveMQ (dark blue) facilitates communication between data acquisition (green), data processing services (light blue), a controller (red) overseeing these services, immediate feedback systems (orange) for users and staff, and the long-term result storage (purple).

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEMPR001

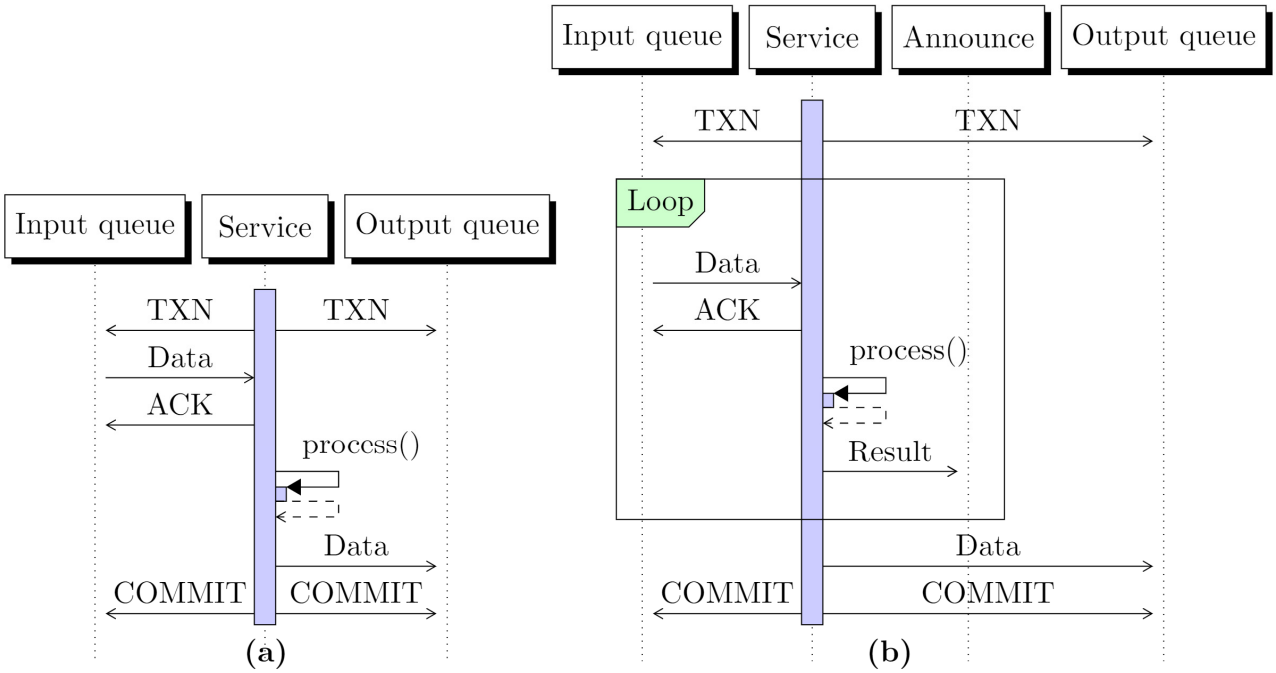

Figure 3: Example message passing protocols to ensure delivery and processing. Time passes from top to bottom. (a) A message pattern guaranteeing that each message is processed exactly once. (b) A message pattern guaranteeing that a group of messages is processed exactly once together by a single service. Additionally, the result is made available to an unspecified list of recipients.

Figure 3(a) describes an example for such a message protocol. Here a service reads (consumes) a single message from an input queue (left) and writes (produces) a single message to an output queue (right). The service does not have exclusive access to either queue and does not know whether other copies of the service are running. To ensure each message is processed exactly once it reads the message within a transaction (TXN) and then conditionally acknowledges (ACK) the receipt of the message, and begins processing. Once the data is successfully processed the result is written to the output queue and the transaction completed (COMMIT). Only then is the message removed from the input queue by the broker and, simultaneously, the data released to the output queue. Should the service crash the broker would make the message available again, and guarantees that nothing is written to the output queue. One example of such a service is the spot finding service, which receives file names and produces per-image information.

A more complex example is shown in Fig. 3(b). In this case multiple related messages are read from an input queue, incrementally processed, with a status message written to an additional announcement queue. The results are only written to the output after the last message is received, and the transaction is completed. An example for such a service would be an indexing service, which requires spot information of all images of a data collection to produce a final indexing solution. Intermediate indexing results can be announced to any real-time processing viewers to give the user immediate feedback. Whether the user is actually running a processing viewer or not, is irrelevant for the processing – if there is no listener to the 'Announce' queue the broker simply discards those messages.

The number of messages in a queue, the rates of incoming and consumed messages can be monitored by a process monitoring service. This service can then dynamically react to current processing demand and, for example when multiple beamlines are running fast grid scans, temporarily increase the number of spot finding services, or warn administrators before problems arise.

CASE STUDY: PER-IMAGE ANALYSIS IN GRID SCANS

While a grid scan is what is known as an "embarrassingly parallel" problem, meaning individual images can be processed independently from one another, this property was not fully exploited. Previously GDA exclusively reserved a single node in the Diamond computing cluster for processing each grid scan, Fig. 4. Images were processed in parallel by this node as they arrived on disk. Every image of a grid scan was analysed for presence of Bragg diffraction, and a number of per-image metrics, including the number of Bragg peaks detected and an estimate of the resolution to which diffraction extends, were reported to the user to enable them to select the best part of the sample for further data collection.

This processing setup was less than optimal for two reasons. When the data were coming in faster than the

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEMPR001

cluster node could process them the grid scan could take longer to process than strictly necessary. Other cluster nodes could not be used to speed up processing. Conversely, when the cluster node processed the data more quickly than it arrived the cluster node would spend some time idling, and the computational resources were not used efficiently.

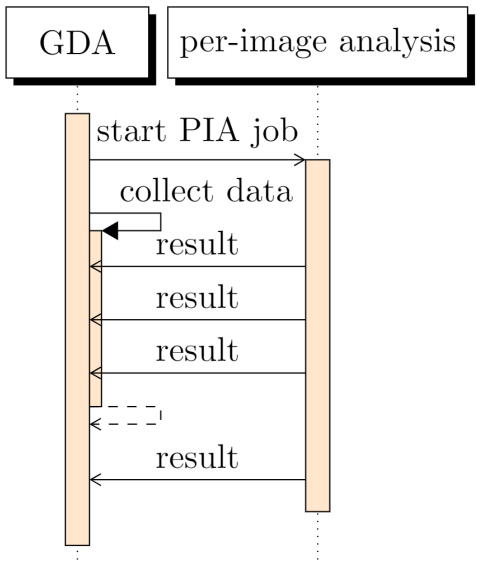

Figure 4: Previous per-image analysis process. GDA runs one image analysis thread on one cluster node to process images as they come in, and to report results back.

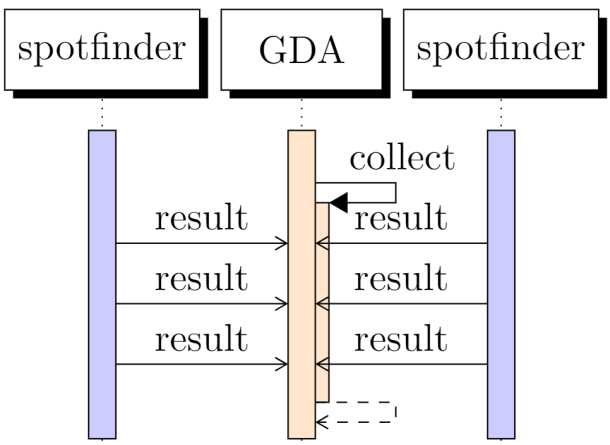

Figure 5: Per-image analysis process in Zocalo. In the new architecture images are processed by many spotfinder services.

In Zocalo a number of spot finding service instances are constantly running and waiting for images to appear, Fig. 5, which eliminates the delay required to reserve a cluster node and start up the processing software. A process monitoring service ensures an appropriate number of spot finding service instances are available depending on the current demand. Each image is analysed by the next available spot finding instance, which communicates the result back to GDA directly, rather than via the database. This reduces the load on the ISPyB database

CASE STUDY: DATA REDUCTION

The aim of data reduction is to extract experimentally relevant data, in this particular case the diffraction reflection amplitudes, from the raw detector image data. The list of reflection amplitudes is a comparatively very small file that can be handled more easily. To arrive at this file a number of steps are required: spots must be found on the raw images; an indexing solution, which predicts where these spots will occur, must be prepared; the intensities of all predicted spot locations must be calculated (integration), then scaled and merged together across the entire data set, Fig. 6. Statistics obtained at the merging step can give the user an indication of the quality of their diffraction data, which may be used to inform the next data collection to be carried out, for example whether the quality of the reduced data is sufficient to answer the questions of the experiment. All these steps are run automatically at DLS using the software xia2 [5]. Currently xia2 is started once the last image of the data set has arrived on disk.

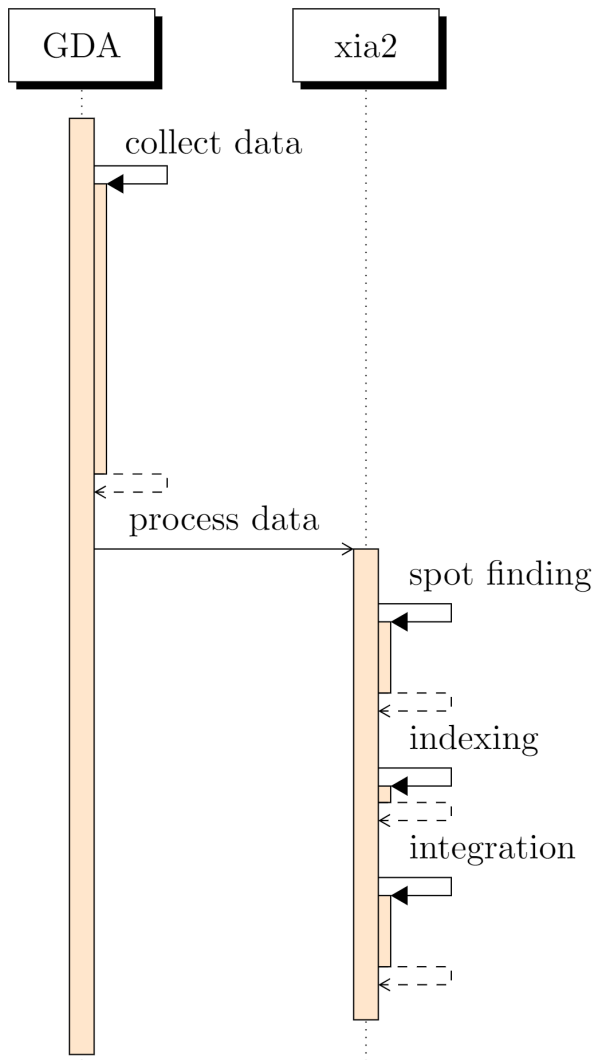

Figure 6: A typical data reduction workflow started after completion of data collection.

ISBN: 978-3-95450-209-7 ISSN: 2226-0358 doi:10.18429/JACoW-ICALEPCS2019-WEMPR001

Experiments may be run exposing the same sample to different X-ray wavelengths, moving the detector to different positions or by orienting and rotating the sample in a different way. At CX beamline I19-1 such a complex experiment may consist of more than 10 individual data collection sweeps, containing more than 20,000 2.5 MB images (50 GB) collected over 90 minutes. In other instances, data from multiple samples may need to be processed together to obtain a data set of suitable quality. Particularly these complex experiments will result in a very long delay between completing the experiment and the results becoming available to the user.

Using the messaging architecture, part of the required processing can be 'front-loaded' and moved to the time of data collection. For example by only reusing the spot finding service already employed for grid scans a considerable part of the processing can be completed before the data collection has finished, Fig. 7. Similarly the indexing and potentially even the integration can be moved closer to the point of data acquisition and therefore drastically reduce the experienced waiting times for the user.

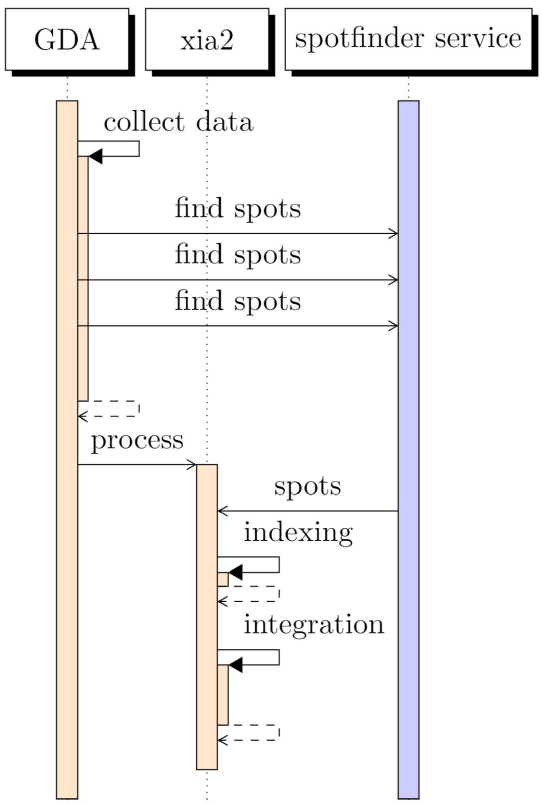

Figure 7: An alternative data reduction workflow where parts can be performed while data are still being collected, reducing the overall runtime.

CONCLUSIONS

Diamond started using Zocalo in production in April 2017 with Zocalo taking over the coordinating of data archiving, and later in 2017 the on-demand reprocessing of data via Synchweb, and parts of grid scan processing. By the end of 2018 all MX/CX autoprocessing has moved to Zocalo.

The Zocalo architecture has proven itself useful in a number of ways. The message-base architecture allows early identification of any processing pipeline issues, a centralised logging component greatly simplifies debugging, the service architecture allows for selective deployment of fixes and fast failure recovery. Staff benefit from the live monitoring capabilities which can be used to identify and rectify any error conditions while the system is running. The architecture enables a more efficient use of computing infrastructure, faster processing, and immediate feedback for users. The system easily coped with the installation of the new Eiger2 XE 16M detectors.

With the successful deployment of Zocalo for the processing of MX and CX data the scope of Zocalo is currently expanding. We have already used the existing Zocalo infrastructure to process serial crystallography workloads, and Zocalo is being extended to manage processing tasks for ptychography, tomography, and cryo-EM.

REFERENCES

[1] G. Winter, and K. E. McAuley, "Automated data collection for macromolecular crystallography", *Methods*, vol. 55, pp. 81-93, 2011.

[2] E. P. Gibbons, M. T. Heron, and N. P. Rees, "GDA and EPICS: Working in unison for science driven data acquisition and control at Diamond Light Source", in *Proc. 13th Int. Conf. on Accelerator and Large Experimental Physics Control Systems (ICALEPCS'11)*, Grenoble, France, Oct. 2011, pp. 529-532.

[3] S. Delagenière, P. Brenchereau, L. Launer, A. W. Ashton, R. Leal, S. Veyrier, J. Gabadinho, E. J. Gordon, S. D. Jones, K. E. Levik *et al.*, "ISPyB: an information management system for synchrotron macromolecular crystallography", *Bioinformatics*, vol. 27, no. 22, pp. 3186-3192, 2011.

[4] S. J. Fisher, K. E. Levik, M. A. Williams, A. W. Ashton, and K. E. McAuley, "SynchWeb: a modern interface for ISPyB", *J. Appl. Cryst.*, vol 48, pp. 927-933, 2015.

[5] G. Winter, "xia2: an expert system for macromolecular crystallography data reduction", *J. Appl. Cryst.*, vol. 43, pp. 186-190, 2010.