Suggestions for Continuing Software Development at LCLS/CXI

T. J. Lane *
Primary Contact

Derek Mendez, Richard Kirian, Daniel Ratner, Jonas Sellberg
Contributions and Discussions

February 4, 2013

Based on our recent experience at LCLS beam time L648, the Doniach team would like to provide some constructive feedback for how the LCLS, and the CXI hutch in particular, could improve the user experience through additional software development. This report has been constructed based on feedback from multiple groups that have run at CXI, including the Doniach, Nilsson, and Chapman groups.

In preparation for our beam time, it came to our attention that there exist multiple concurrent software projects that aim to solve essentially the same problem: pre-processing of the pixel-intensity data written out from CXI's CSPAD detector. These software packages include: CHEETAH, of which there are at least three versions in development, the official PSANA and PYANA software provided by SLAC, and extensions to this software such as KITTY/GIRAFFE. Each of these codes aims to extract raw CSPAD data from run XTC files, assemble the CSPAD measured intensities onto a real-space detector geometry, and apply a series of corrections (for gain, pedestal, *etc.*). These needs are common to nearly all experiments performed at CXI.

While there exist multiple codes to perform data pre-processing at CXI, we found no one of them satisfied our experimental needs. Software developed by user groups was overly specific, difficult to use, and contained bugs. At the same time, the software provided by SLAC (PSANA and PYANA) did not contain all of the functionality we required, and we found it difficult to precisely understand the execution of this software given current documentation. Thus, we believe that both the user groups and SLAC personnel could greatly benefit from collaborating on a single, well-written and well-documented software package.

This report focuses primarily on the CXI hutch at LCLS, and the in-house software supported by CXI, PSANA and PYANA. It is our hope that these software suites can be developed to the point where they provide a general platform for data pre-processing of all experiments performed at CXI. This would provide a significant boon in scientific productivity for users, facilitate on-line data analysis, and decrease the workload on SLAC's software engineers. In what follows, we provide a set of suggestions that SLAC can implement immediately to help realize this goal. Considering the considerable investment SLAC has made in LCLS infrastructure and staff, software should not be a limiting factor in the scientific productivity of the LCLS.

*Stanford University, Department of Chemistry: tjlane@stanford.edu

Guiding Principles for Continuing Software Development at LCLS

There are a number of principles that we believe should guide further software development for use at the LCLS:

- **Documentation.** Any software in use at the LCLS will be used by groups with mixed backgrounds and familiarity with LCLS/CXI particulars. Therefore LCLS software (PSANA/PYANA) *should be clearly documented* in a browsable format. Currently, documentation is hosted on SLAC's confluence page; for users, the logical layout of this documentation is not clear from a user's perspective. For example, there are at least four pages that contain information about accessing the CSPAD geometry:

  `https://confluence.slac.stanford.edu/display/PCDS/CSPad+alignment`
  `https://confluence.slac.stanford.edu/display/PCDS/2011-06-20+CSPad+alignment+parameters`
  `https://confluence.slac.stanford.edu/display/PCDS/CSPad+metrology+and+calibration+files%2C+links`
  `https://confluence.slac.stanford.edu/display/PCDS/Tutorial+-+python%2C+pyana+and+matplotlib`

  each of which contain different and incomplete documentation about a critical piece of information: the detector geometry.

- **Transparency.** Concerns have been raised by LCLS user groups about employing software who's precise execution is unclear. No scientist wants to trust their results to a "black box". Therefore, any software provided by SLAC should be *transparent*. Documentation about how to use any algorithm should be accompanied by documentation about how those algorithms work. The user of software should *not* be forced to read source code to know what a piece of software is doing.

- **Extensibility.** Any code provided for analysis of LCLS data should be easily extensible by user groups. PSANA already provides a limited platform for extensibility, but this principle could be greatly advanced by moving to a modern scientific software model (see **Moving to a Modern Scientific Software Model** below).

- **Portability.** Currently, LCLS codes such as PSANA and PYANA only run on SLAC machines. These codes should be *portable* to non-SLAC machines. For example, the XFEL group at DESY, in Germany, has developed and maintained their own software in part due to this lack of portability.

Moving to a Modern Scientific Software Model

SLAC resources are limited. It would be unreasonable for user groups to expect all software challenges they face to be dealt with by SLAC or any of its daughter institutions (LCLS, CXI). Therefore we would like to suggest that the central analysis software projects in use at CXI, PSANA and PYANA, move to a modern collaborative software model. We believe this model could yield large productivity increases for not only SLAC software staff, but also LCLS user groups who instead of simultaneously developing equivalent codes, can contribute to one central project.

In this model software is hosted in a highly visible location where users can easily browse the code. This hosting platform should support revision control, bug reports, issue tracking, and functionality

for easily incorporating code contributions from third parties. A set of core developers (SLAC employees) act not only to develop new software functionality, but also as gatekeepers and overseers of the software project. This allows third parties (LCLS user groups) to easily contribute code, which can then be verified for correctness by SLAC software engineers.

This model results in large productivity gains for both groups. User groups can contribute to and employ a central code, and they can be assured that this code is well-documented and runs in a bug-free manner due to the oversight provided by SLAC. Meanwhile, SLAC's software engineering team can focus on higher-level software development issues (*e.g.* engineering extensibility and portability) and leave the details of algorithmic development to users. For this collaborative model to succeed, however, it is necessary to provide tools (beyond basic revision control) that allow users and SLAC employees to communicate and collaborate effectively.

Fortunately, tools for facilitating this communication exist and can be readily implemented. Our team has experience with GitHub, a powerful software development platform that supports revision control, code branching, issue and bug tracking, and hosts a discussion platform. GitHub provides an "enterprise" software suite (`https://enterprise.github.com`) that would allow for secure software development contained completely on-site at SLAC. GitHub is just one example of a way to implement a platform to support collaborative code development; we would be open to alternatives.

Functionality Currently Missing in PSANA

While PSANA provides a great deal of functionality, it lacks certain general features we believe would be helpful to all groups running at the CXI hutch. This gap in functionality has spawned the development of large software projects by user groups, who often reproduce each other's work. Adding this functionality to PSANA should greatly improve the scientific output of LCLS user groups. We have enumerated a few essential features that we believe PSANA lacks that should be immediately implemented, roughly in order of importance:

- **Access to the detector geometry:** A clear explanation of the provided geometry specification, with good diagrams, would be extremely valuable to the users. Reports from numerous people suggest that the current specification is difficult to understand. It would be favorable to have a single definitive explanation, clearly documented. We suggest a vectorized geometry specification would be easier to understand for most people than what currently exists; such a representation is used in *e.g.* Felipe Maia's HAWK, Tom White's CRYSTFEL, and CXIDB (`http://cxidb.org/`).
- **Detector centering:** The geometries provided at CXI do not contain definitive coordinates on the location of the x-ray beam relative to the CSPAD.
- **Automated quad shifting:** Further, the relative positions of each CSPAD quad are often incorrect. An automated way to correct this lacking geometrical information would be of great use.
- **Parallel execution:** While PSANA, implemented in C++, is quite fast, it runs only on a single core. Because most data processing tasks at CXI are embarrassingly parallel, it should be straightforward to implement a parallel version of PSANA. Parallel execution is currently provided by some user-developed software packages such as CHEETAH.

- **Non-linear gain correction:** While a gain-correction module exists in PSANA, the user is required to provide their own calibration file. A standard gain calibration for the CSPAD and a gain correction algorithm would be of great help to users.
- **Polarization correction:** An automated method to correct for beam polarization would be of use.
- **Solid angle correction:** Solid-angle corrections are provided in CHEETAH and would help complete PSANA's functionality.
- **Photon counting:** For certain experiments, counting individual photons is necessary. A standard algorithm to convert the CSPAD's ambiguous ADUs to a photon count would be of great benefit during these experiments.

Conclusions

The Doniach team's experience at the LCLS/CXI was overwhelmingly positive. The support we received from the staff at CXI could only be described as excellent. Our primary beam time challenge was choosing, using, and modifying software to pre-process our data. This undoubtedly reduced our scientific productivity while at CXI, and continues to pose a challenge to us weeks after our beam time has ended.

We believe that PSANA, if hosted on a transparent platform that encouraged user contributions, could provide a general software package for data pre-processing at CXI. We have laid out our vision for how to make this vision a reality. Further, we have provided a concrete list of additional functionality that, if implemented, would make PSANA the clear choice amongst currently available software.

We hope that continued feedback between users and LCLS staff can continue to increase the productivity of the unique resource the LCLS provides. We encourage any interested or involved party to contact us about the suggestions laid out here.