SLAC NATIONAL ACCELERATOR LABORATORY

Further Development of LCLS-II Simulacrum

Helen Chaffee[1+], William Colocho[2+]

[1]Harvey Mudd College, 301 Platt Blvd, Claremont, CA 91711, USA.

[2]Linac Coherent Light Source, SLAC National Accelerator Laboratory, 2575 Sand Hill Road, Menlo Park, CA 94025, USA.

[+]Contact: hchaffee@g.hmc.edu , colocho@slac.stanford.edu

Introduction

Beamtime at LCLS is limited, making it difficult to manage instrument maintenance, software updates, and data collection schedules.

A mathematical simulation of LCLS would make beamtime more accessible to all, allowing scientists to run their own, personal LCLS-II before bringing their ideas to the live machine.

This "**simulacrum**" project is already semi-functional. The goal of this internship was to address current demands for simulacrum by adding a wide variety of additional features to existing programs.

Keywords: simulacrum, simulation, linear accelerator

Research

The simulacrum team had identified more improvements than could be accomplished within the scope of a summer internship. I selected a few that interest me. All these projects involve "**service**" programs. These programs mimic the behavior of major devices that determine the properties of the beam (Fig 2.)

Most of these projects are unfinished, given that I am extending my internship.

Dynamic TMIT

We made the transmitted intensity (TMIT) of the beam go up to 1e9 electrons when the beam is on, and 0 when the beam is off or malfunctioning (Fig 1.)

This project familiarized me with the functionality of a beam position monitor (**BPM**), which tracks the spread of the beam along a vertical cross-section. Anywhere a BPM did not detect the beam, TMIT was made 0. Otherwise, TMIT was 1e9 electrons. This change allows those writing simulacrum GUIs to detect whether the beam is on or off at a certain location. This makes it easier to write GUIs using simulacrum. This is necessary because simulacrum needs to be friendly to GUI authors to reach widespread use at LCLS.

Cryogenic, SRF Services

My second project gave me a chance to build services from scratch. As part of LCLS-II, which produced a first lasing last month, devices are being built to make LCLS-II superconducting. Currently, LCLS does not have superconducting capabilities, so new services must be written to simulate the new machinery. These include: cryogenic modules and superconducting radio frequency (SRF) cavities. The cryogenics supercool the magnets, making them superconducting. The SRF cavities produce the electric potential to accelerate the beam.

Currently, we have a working draft of a cryogenic service, with the help of Ahmed Osman in the simulacrum group. However, we still need implement temperature change according to liquid helium levels. The SRF service is still in the design stage of production.

Alias Recognition

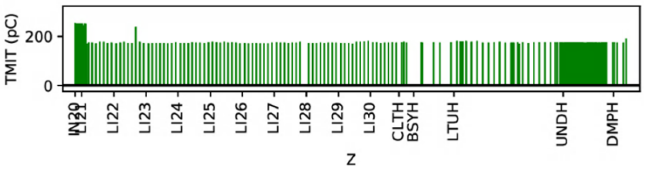
Simulacrum has to talk to various applications, but that poses a problem when applications use different databases. One inconsistency is the naming convention for devices. When applications have different names for a device, simulacrum needs to know all those aliases to facilitate the flow of data that corresponds with each device.

In the same line that establishes data flow with another application, I made each service request its own alias. This bypasses the old method, in which the service would refer to a hard-coded translator file. This translator file had to be updated by hand whenever new devices were added. Now, we can edit any service to retrieve its own alias directly from the external application.

Generate Simulacrum Data

As of this week, I will start generating simulacrum data for a group that is creating anomaly detection software for LCLS. This would speed up the time to troubleshoot breakdowns in the accelerator by monitoring live data for abnormal readings.

Fig 1. Below: Live machine data for TMIT. Z is the horizontal position along the beamline. The Z tags (e.g., LI22) demarcate the sectors that house the beamline. The vertical axis is TMIT in pico-Coulombs (pC). The conversion from 1e9 electrons to pC is 160.2 pC. This agrees with the live machine data. Therefore, simulacrum uses TMIT = 1e9 electrons as the default for a working BPM.



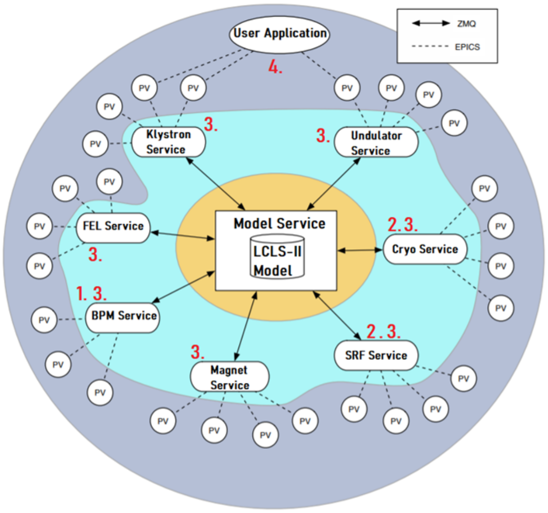


Fig 2. Above: structure of simulacrum. Working across all the services has been necessary to ensure that my changes do not break the interconnected web of code. Red annotations represent where I have made changes.

1. Adding dynamic TMIT to the BPM service
2. Adding new services
3. Alias Recognition
4. Generating simulacrum data

Conclusions

Since joining LCLS last month, I have taken on a small, diverse set of projects to develop a holistic understanding simulacrum. Now, I can add new variables to device services and draft new services. For the future, I am looking forward to seeing the data that the services can generate when they work in tandem.

I have decided to extend my internship so that I can continue to explore the scope of this project and contribute to its development.

Acknowledgments

I'd like to thank William, Lisa, and everyone else who is excited to see where this project goes next. Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.

Date: 08/14/2020