CS486C – Senior Capstone Design in Computer Science

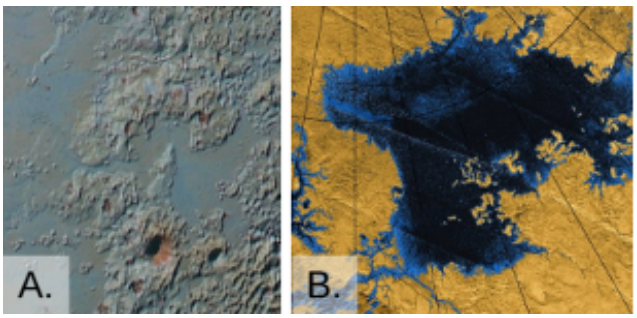
Project Description

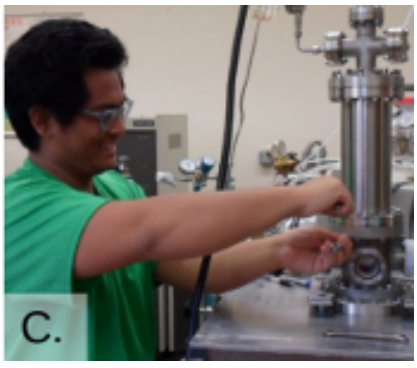
Project Title: Astrophysical Materials Lab Data Logging and Instrument Control Tool	
Sponsor Information: LOWELL OBSERVATORY 125 YEARS | 1894 - 2019	Dr. Will Grundy, Astronomer Dr. Jennifer Hanley, Astronmer grundy@lowell.edu, jhanley@lowell.edu Lowell Observatory Flagstaff AZ

Project overview:

Humans have sent spacecraft to explore many planetary bodies in our Solar System, discovering strange and exotic extraterrestrial environments which are surely just a taste of the diversity the universe is capable of. What makes an environment work is the combination of materials and energy sources available to mobilize them. Earth's geology is built of silicate rocks melted and uplifted by energy from Earth's deep interior, and then sculpted by air and water mobilized by sunlight. Water's ability to change state between liquid, ice, and vapor is key to eroding rocks via rain and snow, rivers, glaciers, and ocean waves. Our geology would be unrecognizable without it. But other materials and energy sources can dominate geology elsewhere, with results that can be truly bizarre.

Our laboratory investigates materials that are generally gases at room temperature, but liquefy and freeze solid at the extremely cold temperatures prevalent far from the heat of the Sun. These materials play important roles in outer solar system environments, enabling all sorts of exotic low-temperature geology. For instance, the glaciers of Pluto consist of frozen nitrogen, with methane and carbon monoxide, while the seas on Titan are mostly liquid methane and ethane (right).





To be able to study such materials, we have developed a sophisticated low temperature laboratory at NAU (left), in which we can simulate the extreme conditions encountered in extraterrestrial environments like those on Pluto and Titan. Very carefully monitoring and tightly controlling the temperature of samples in the low temperature chamber is absolutely essential to this research. We are currently using a rudimentary homebuilt monitoring tool for this purpose, but it is based on outdated frameworks and is missing many features that one might expect in a professionally-designed solution. What is needed is a full re-design to provide a modern GUI solution: a modular architecture using modern actively-maintained frameworks that allows connection, configuration, and monitoring of additional instruments, e.g., such as pressure gauges and cameras, and the ability to send notifications when certain criteria have been met. Specifically, we envision an application that runs on a Linux machine in our laboratory; some key features would include:

Bare bones basics (Minimum viable product). Able only to effectively replace existing rudimentary system

- Able to connect to and read temperature measurements and heater settings from existing temperature control devices.

- Able to transmit user commands to and receive responses from existing temperature control devices. These commands are logged along with temperature data.
- Able to configure basic monitoring parameters, e.g., a sampling interval.
- Able to display current temperature and temperature history, and enable user to adjust display parameters such as plot scales for temperature and time, which temperatures are plotted, colors used, etc.
- Record data to a log file and be able to pick up and resume recording after a restart/power glitch.
- Able to set alarms for user-selectable temperature criteria.
- A basic test harness that simulates the temperature sensor for testing purposes, i.e., presents same API as the physical sensor/logging system, but returns artificial (but plausible) values. In end phase, product will be tested with actual sensors in our lab.

A satisfying product: a complete, flexible solution that is truly usable

- A more refined GUI including a dashboard view to show status of all sensors and monitors.
- Ability to define and connect arbitrary new sensor types/brands (e.g., pressure gauges, cameras, and others). Once defined, these appear in a library for selection and deployment as new sensors are connected.
- New sensors can be connected for monitoring: attach sensor, specify connection parameters, select the appropriate sensor type/brand from library, and configure monitoring; flip sensor to "active" and monitoring/recording begins.
- Simulated (test harness) sensors of all types supported, for development purposes. End stage development will then be in the lab with these simulated devices replaced by the real ones.
- Ability to specify and modify more complex sampling parameters such as averaging over multiple samples, applying a calibration correction to the result from the sensor, co-adding multiple images and saving the result, etc.
- Ability to specify a full range of monitoring parameters, e.g., sampling interval, daily time periods in which to monitor, etc. Some parameters are standard, others may be associated with specific types/brands of sensor hardware.
- For any sensor or combination of sensors, able to set "monitors" that specify one or more criteria, plus an action (sound alarm, send text, send email, etc.) to be taken if those conditions are met.
- For any sensor, be able to specify disposition of recorded data, e.g., save off tabular summary file to specified directory, mail it to some address, etc.
- Provides a sophisticated analytic viewer tool for graphically reviewing various sensor values over time, including old log files. Allows timescale zoom in/out, indications of timing of logged commands (click to view), loading multiple sensors for juxtaposed graphing (multiple lines on the same graph), etc.
- Must maintain backward compatibility with data logs produced by existing system. Specify desired recording/monitoring.

Fancy extras (stretch goals):

- Provides multiple users and permissions/roles, allowing differentiated access across users, e.g., only admins can change the settings, lab members may monitor (but not change) instruments, and guests might have access only to some overall aggregate information to support citizen science/educational efforts.
- Provide operation in various distinct modes, such as "active experiment", "standby safety monitoring", etc.
- Moves the whole architecture to a meta level: a laboratory can register an account, register their laboratory server, and use the monitoring architecture to monitor instruments in their lab.

Knowledge, skills, and expertise required:

- Knowledge of modern programming techniques, frameworks, and languages required to develop the application.
- Knowledge of analog/digital sensor interfaces and communication.
- Experience in GUI design and coding (including principles of usability and discoverability).

- Familiarity with Linux computing environment.

Equipment requirements:

- A standard Linux software development environment and open-source IDEs/tools. No specialized software is required.
- Final testing will need to be done in the laboratory, since they involve interfacing with special purpose laboratory hardware that is only available there (via USB, Ethernet, and RS232 interfaces). Most development can be done with simulated sensor harness.

Deliverables:

- Fully functioning new software that runs in the laboratory computing environment (currently Fedora Core 32, but upgraded as new versions are released).
- A strong as-built report detailing the design and implementation of the product in a complete, clear and professional manner. This document should provide a basis for future development of the product.
- Complete professionally-documented codebase, delivered both as a repository in GitHub, BitBucket,or some other version control repository; and as a physical archive on a USB drive.