Dripline: A Distributed Experiment Control System

Eric Gonzalez, Ben LaRoque, Noah Oblath

Introduction To Dripline

Dripline was originally a piece of software developed for Project 8, but is now being developed to be a general control system for various hardware. Currently, Dripline has one implementation for Python and one for C++. This has created unnecessary complications for researchers who are interested in using Dripline as they would have to manage two separate systems at the same time. This results in time spent doing management tasks rather than experimental ones. It is our overall goal to develop a piece of software that not only holds up to industry standards of having good documentation and extensive testing, but also works through a single Python application programming interface (API) which encapsulates the entirety of the C++ code to make the system more accessible to researchers.

Unit And Integrated Testing

Unit testing is where the smallest unit of a program is tested. For our purposes, the smallest unit of testing will be a single function. Unit testing is done to ensure that all pieces of the program at their most basic level work as intended. Integration testing is where we test the smallest units as a group in order to expose faults between the interactions of individual units. These tests can vary from testing a function which contains other functions to testing the interactions of classes. Features may be added or removed as the project continues, and these changes must be reflected in the code and properly tested. If a feature is added or removed, any part of the program that had some sort of interaction with that feature must be tested again to ensure code stability. This process happens consistently throughout development.

Wrapping The Code

Python and C++ handle some things differently, such as how each one may define integers, strings, or functions. The pybind11 library was created to expose the variable types of each language to the other which allows for “bindings” to be created between the two languages. For a Dripline user to be able to access a variable or function in C++, a “wrapper” class must be made for the respective class in which that variable or function is defined. Once the wrapper class is made, the user will have access to everything that has been binded inside that wrapper class via the Python API.

Class Hierarchy And Wrapping

We have demonstrated the core functionality of Dripline by creating a key value store (KVS). Using the KVS we are able to:

- Create a service
- Create endpoints and add them to a service
- Set attributes of endpoints
- Retrieve attributes of endpoints
- Receive replies when unexpected requests are sent

If the agent sends an OP_CMD request to endpoint waffles with the instruction “get price”, it will return the value 3 to the user. Similarly, if an OP_CMD request to endpoint peaches with the instruction “set price 8” is sent, it will change price from 10 to 8 and inform the user. If an invalid OP_CMD is sent with the instruction “get nutrition facts”, a message will be displayed to the user that no such attribute exists and no action will be taken.

Results And Future Work

The final product of this research is a basic implementation of the Dripline KVS which has the core functionality upon which all other features will be built. Future work will consist of continuing to add features to this now existing system as well as using Dripline to interact with the physical components of hardware rather than through a simulation.

Acknowledgements

The 2019 STEM Teacher and Researcher Program and this project have been made possible through support from Chevron (www.chvron.com), the National Science Foundation through the Robert Noyce Program under Grant #1836335 and 1340110, the California State University Office of the Chancellor, and California Polytechnic State University in partnership with Pacific Northwest National Laboratory. This work has been supported by the Laboratory Directed Research and Development Program at PNNL. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funders.