Development Environment for SDN Systems

I. Abstract

Software-Defined Networking (SDN) allows network administrators to provide a centralized, simple programming model over a complex set of networks. Challenges within SDN include the programming framework -- that is, the language and abstractions. The MAPLE system sets forth a model for SDN programming of algorithmic policies. This project involves creating the front-end development environment for this type of SDN programming. The aim of the project is to create an intuitive, user-friendly front end of a development environment in order to make using and working with this system easier.

II. Background / Literature Review

I was drawn to this area of computer science because both areas of the project interest me: this type of network system, based on abstraction of higher-level functionality (and its real-world applications, including CERN using an SDN to share big data from its Large Hadron Collider).\(^1\) I am also primarily interested in front-end, user-facing systems, and how to make them elegant, easy-to-use, and efficiently functional for all of their intended uses. This project combines both of these interests in a salient way.

There are two main areas of consideration for this project: literature on Software-Defined Networking and background on development environments.

I am primarily reviewing Software-Defined Networking within two contexts: the field as a whole, in order to better understand the general background, and SDN for this specific project: MAPLE (http://conferences.sigcomm.org/sigcomm/2013/papers/sigcomm/p87.pdf), which aims to simplify SDN programming by allowing a programmer to design a centralized algorithm for deciding the behaviors of the entire network. One notable SDN resource is OpenDayLight, an open source platform for building programmable, software-defined networks.

\(^1\) https://www.youtube.com/watch?v=rSLaEha85Dw&feature=youtu.be
networks. This project contains a framework and platform intended to accelerate adoption of SDN systems.

To begin on development environments, I have considered several different popular current interfaces - particularly those used by large technology companies. These include Apple’s XCode environment, Google’s Android Programming Environment, Microsoft’s Visual Studio, and Eclipse. These systems share some commonalities with each other -- among them, general layout of the environment often looks similar from system to system (navigation along one side, tools on the opposite side, workspace in the middle, is a frequently-found layout, for example). Development environments also share many similar functionalities, from basic console usage, to a large workspace, to some sort of GUI system to see progress of a running system. The main challenge in these designs seems to be balancing the necessary complexity that a user could want with maintaining a simple, mostly-intuitive system accessible to users from many different backgrounds.

III. Approach

I aim to design and implement a user-facing development environment for this larger project by considering several factors:

1. A layout that would be most beneficial to a user of this system
2. Which functionalities are most important to a user of this system
3. Which functionalities are less important, but perhaps desirable if possible

In order to decide these points, I will be both researching SDN and development environments, as well as working with Professor Yang to design a system that will be useful for the larger project.

Once a design of the system is figured out at both a high-level overview and in a more detailed plan, I will implement the front-end system, using the tools that would be most effective for building the specific design (some possibilities include AngularJS and D3.js).

During this entire process, I will be keeping track of the work in a research document, and writing up the work as I go along, in order to both have a cogent document to reference, and to maintain ideas and write-ups as decisions about the project are being made.

---

2 https://www.opendaylight.org/
IV. Deliverables

At the end of this project there should be:

1. A detailed plan for design and implementation of a development environment system to be used in working with SDN.
2. Code for an implemented, functional front-end system based on the ultimate design, agreed upon by myself and Professor Yang, that can be used for this SDN project.
3. A write-up that includes project background, approach, design process, tools used, and areas of future work or further possible implementations

V. Timeline

My approach to the project will follow this high-level overview:

- Two weeks to review current systems, background information, and propose a new system
  - January 31 - February 14
- Two weeks to design the system
  - February 14 - February 28
- Five weeks for implementation of the project
  - February 28 - April 3
- Two weeks to polish up the project
  - April 3 - April 17
- Two weeks for wrap-up and any miscellaneous work that needs to be done
  - April 17 - May 1

The five weeks of implementation could be shortened to four as well. In addition, this timeline gives an extra few days (before the final due date) if necessary for final wrap-up of the project.