Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Software Development Project

Dashboard for ROS-based System

January 16, 2020

Team Members

1. **Lokesh Veeramacheneni**
2. **Zuha Karim**
3. **Anargh Viswanath**

Problem Setting

- In ROS-based system running on a robot and being controlled remotely - several processes being carried-out simultaneously.
- Serious problem $\rightarrow$ **Some process stops eg.Nodes**$\rightarrow$ **Application Failure.**
- Need for efficient and real-time monitoring for remedial measures.

Project Objective

Developing a Dashboard UI for monitoring the ROS system running on a robot remotely through a computer system via web services.

Client and Requirements

Client: Deebul Nair

It is intended to be used by **Robocup @work lab.**

Requirements:

1. Should be implemented inside Cockpit - open web-based interface for servers
2. Smooth monitoring of ROS and Robot's system metrics.
3. Effortless visualization of ROS Nodes.
4. Start and kill the ROS nodes(if possible).

Main Components of Software

1. **Cockpit** - Integrated open web-based interface for GNU/Linux server.

   Features of Cockpit:

   - Monitor and administer several servers at the same time.
   - Uses the system's normal user logins and privileges by default.
   - Network login supported.
   - When inactive, no extra load on the server.
   - Inbuilt packages show the status of the system.
   - Embedded terminal present within interface.

2. **ROS Kinetic** - Installed on the robot which has to be monitored.

Coding standards

1. **Python**
   - PEP8
2. **Javascript**
   - Google JavaScript Style Guide

Organization of Work

- Project development, management and release carried out through Github.
- Each member has a separate branch for development.
- Master branch has reviewed components merged from branches.
- Issues, Sprints and progress also being placed.
- Link to Github Repository

Project Implementation

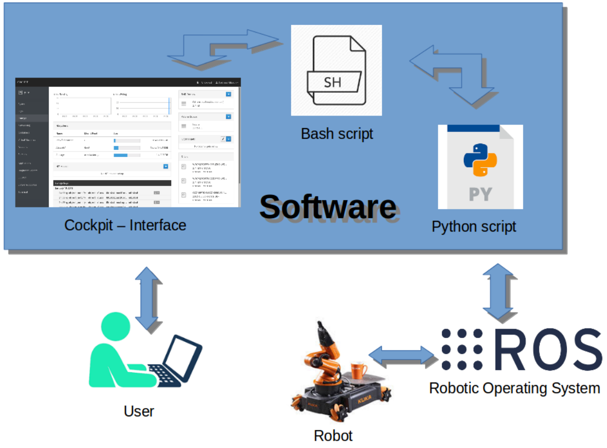

Figure 1: Working of Software

Dashboard Layout

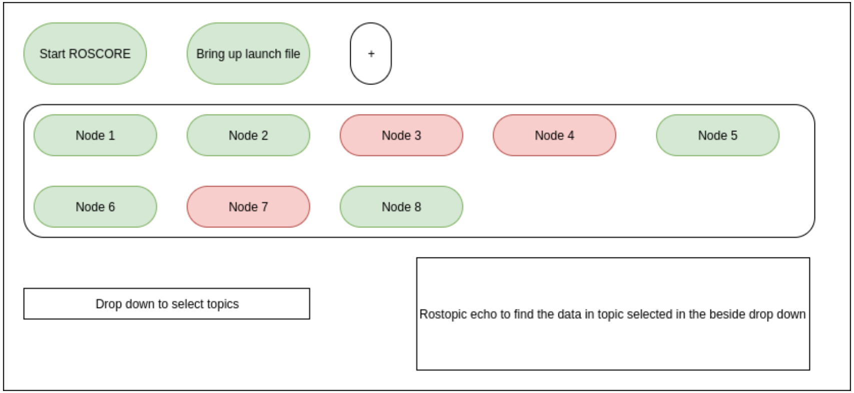
Start ROSCORE
Bring up launch file
+
Node 1
Node 2
Node 3
Node 4
Node 5
Node 6
Node 7
Node 8
Drop down to select topics
Rostopic echo to find the data in topic selected in the beside drop down

Figure 2: Layout of Dashboard

Module I

Start and Kill Rosmaster. Selecting Launch file Description.

Module II

Getting the node status from launcher and display.

Module III

Getting the rostopics and diplaying details.

Problems occured during development

- **Creating packages inside ROS.**
- **Connecting Cockpit to ROS.**
- **Starting and Killing ROS master from Dashboard.**
- **Getting Node status of launch file inside cockpit and display.**
- **Getting Topics as list and display.**

Development Status - Capabilities

- Easy installation of Cockpit and Dashboard inside system.
- The Dashboard can start and kill ROS master.
- Display the status of ROS nodes inside selected launch file.
- Display information about the ROS topics.

Development Status - Limitations

- Certain GUI based nodes not launched from Dashboard.
- The feature for starting and killing nodes not developed.

Development Status - Wish List

- **Small wish - Incorporating the feature for starting and killing nodes for better usage.**
- **Big Dream - Dashboard able to control all aspects of ROS.**

Demonstration

A brief demonstration of the system

Summary

- Implememted Software-development practices for developing a Dashboard for monitoring ROS-based system via web services.
- Completed the basic package for the client's initial requirements.
- The software can be extended and improved through further work.