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

October 31, 2019

Team Members

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

Currently all members are working as developers.

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 Dashboard Features

Client: Deebul Nair

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

Features offered by the Dashboard:

1. Effortless visualization of ROS Nodes.
2. Smooth monitoring of ROS and Robot's system metrics.
3. Start and kill the ROS nodes.(if possible)

Main Components

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

- Currently the project management is being carried ouut through Github.
- Each member has a separate branch for development.
- Master branch is having the reviewed components merged from branches.
- Issues, Sprints and progress also being placed.
- Link to Github Repository.
- Alternatives to Github such as Jira being considered.

Current stage of work

Description of terms

- rosbridge - package providing JSON API to ROS functionality for non-ROS programs.
- websocket - protocol to establish stable connection between Client and Server.
- roslib - base dependencies and support libraries for ROS.

Current stage of work

Connecting Html to ROS

1. Json files are used to establish a connection with ROS through web browser via websockets.
2. Created a ROS node and connected it to local host port 9091.
3. Added a listner, publisher and displayed the result on html page.
4. Used roslib objects and functions for subscribing and listening to a topic.
5. Successful in connecting html with ros using rosbrige and roscore.
6. *More detailed work has to be carried out.*

Current stage of work

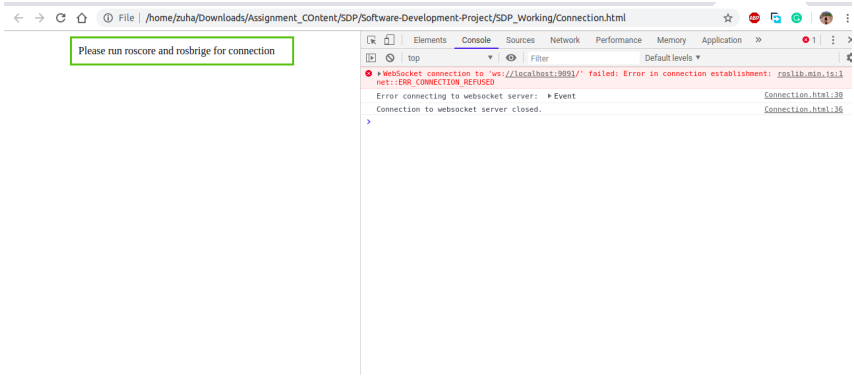

Figure 1: Not connected

Current stage of work

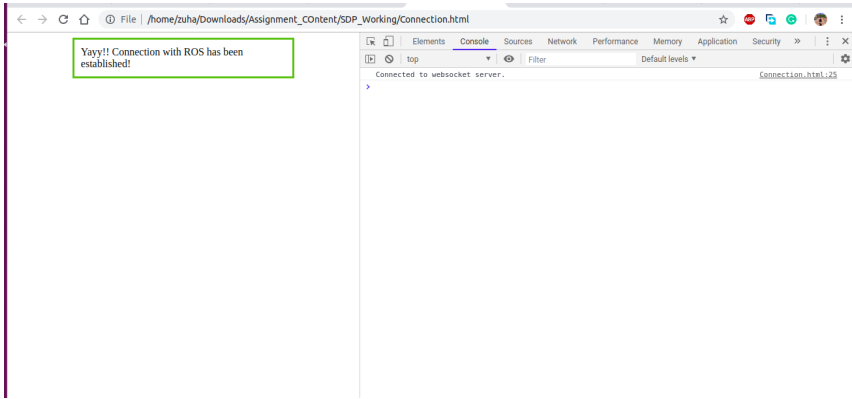

Figure 2: Successful connection

Course of action for upcoming week

- Start Sprint 1 (2-3 weeks)
- Assign Userstories and work on them