How to implement the software bridge in ROS

1. Install a version of ROS for the system you want to connect to the embed library using the tutorial on the official ROS website <http://wiki.ros.org/kinetic/Installation> . If you already have a ROS package and you want to add a library go to step 8.

2. Set up your source to include the “setup.bash” for example if you installed kinetic on ubuntu then it would be “source /opt/ros/kinetic/setup.bash”.

3. Create and build a ROS Workspace using catkin.

mkdir -p ~/catkin\_ws/src

cd ~/catkin\_ws/

catkin\_make

4. Set up your source to include the “setup.bash” in devel folder created in the catkin workspace. For example, “source devel/setup.bash”.

5. Create the catkin package replacing <package\_name> with the name of the package

catkin\_create\_pkg <package\_name> [depend1] [depend2] [depend3]

6. Go to the catkin\_ws directory and run “catkin\_make” to build the catkin package

7. Then go to the package.xml and added depencies. For example:

<buildtool\_depend>catkin</buildtool\_depend>

<build\_depend>roscpp</build\_depend>

<build\_depend>rospy</build\_depend>

<build\_depend>std\_msgs</build\_depend>

<exec\_depend>roscpp</exec\_depend>

<exec\_depend>rospy</exec\_depend>

<exec\_depend>std\_msgs</exec\_depend>

6. Go back to the catkin\_ws directory and run “catkin\_make” to this time build the ROS workspace

7. Create nodes in your ROS workspace create c++ files in src and python files in scripts

8. Add the “forwardsSerial.py” library which acts as the software bridge to your ROS package’s scripts folder.

9. The ROS node that you to use the “forwardsSerial.py” library should be python.

10. In the python file you want to reference the library, put in “import forwardsSerial” and “import time” at the top of the file.

11. Create an instance of the classes that you want to use for example to use the motors class you would write: “moveClass=forwardsSerial.motors()”.

12. To call a function from a forwardsSerial.py class write the instance of a class and then the function that it is in. For example, to use the forward(X) function and you called the instance “moveClass” you would write “moveClass.forward(X)” and then replace X with an integer between -1 and 1. This would write “forwardX” to the mbed commandline.

13. Put a “time.sleep(X)” in between each function that you call from the library’s classes, where X should be a minimum of 0.015, but if the function fails increase the number to as large as it needs to be.

14. For functions that return values save them as strings and then these strings can be converted and sent to other ROS nodes using publisher and subscriber mechanisms. For example, if you wanted to get the direct current voltage and publish it to a listener node by writing “dc\_voltage\_str = "DC voltage: %s" % sensorClass.get\_dc\_voltage()”, “rospy.loginfo(dc\_voltage\_str)” and “pub.publish(dc\_voltage\_str)”. This presumes that you imported rospy and created a rospy publisher declared as “pub”.

15. Edit the serialConnect function in the forwardsSerial.py to correct the serial input to fit your device.