PS8: ARIAC box unloader

In this problem set, you will write code that interacts with ARIAC sensors and sends goals to a conveyor action server and a robot behavior server and invokes functions of a BoxInspector object. Your objective to to detect and remove a defective part from a box, then to unload the rest of the parts from the box.

In your ros_ws/src directory, get our team code with:
https://github.com/cwru-robotics/cwru_ariac_2018.git

Make sure this compiles cleanly. There may be packages that interfere with "learning_ros". If so, remove the duplicate "learning_ros" packages. However, you will still need "learning_ros_external_pkgs_kinetic" in your src directory.

Start up Gazebo simulating ARIAC in a specific test mode with:
roslaunch cwru_ariac_launch sample_environment.launch fill_demo_shipment:=true

Make sure this launches correctly. If you get complaints about gzserver not found, this is likely because you do not have the external-packages repo in your directory.

Enable the "competition' manually by entering the following line in a terminal:
rosservice call /ariac/start_competition

Open three terminals and start the following nodes:

Start up robot move action server:
rosrun kuka_move_as kuka_behavior_as2

Start up the conveyor action server:
rosrun conveyor_as conveyor_as

Start up the new box-unloader node:
rosrun shipment_filler unload_box

You should see the robot move to an initial pose and the conveyor advance a box to center under box_camera_1. The box should contain 3 parts. One of these parts is defective, as reported by the quality sensor camera quality_control_sensor_1.

The "unload_box" node will suspend here, waiting for your input.

Modify the unload_box, writing code to identify the type (name) of the defective part. (This is important for parts of different heights to be able to infer the appropriate pick-up height for the robot). To do so, you will need to compare coordinates from the box inspector vs coordinates from the quality inspector. Once you have found the name of the defective part, update the "current_part" object to contain the correct part name.

Next, modify the code to remove the defective part. (See example, line 138, for how to command the robot action server to do so).

Finally, (and pointlessly), have the robot remove the rest of the parts from the box and discard them.

Submit a report describing your approach, describing your results, and including any interesting observations.   Include a link to your unload-box code on github. Upload a movie of your solution.