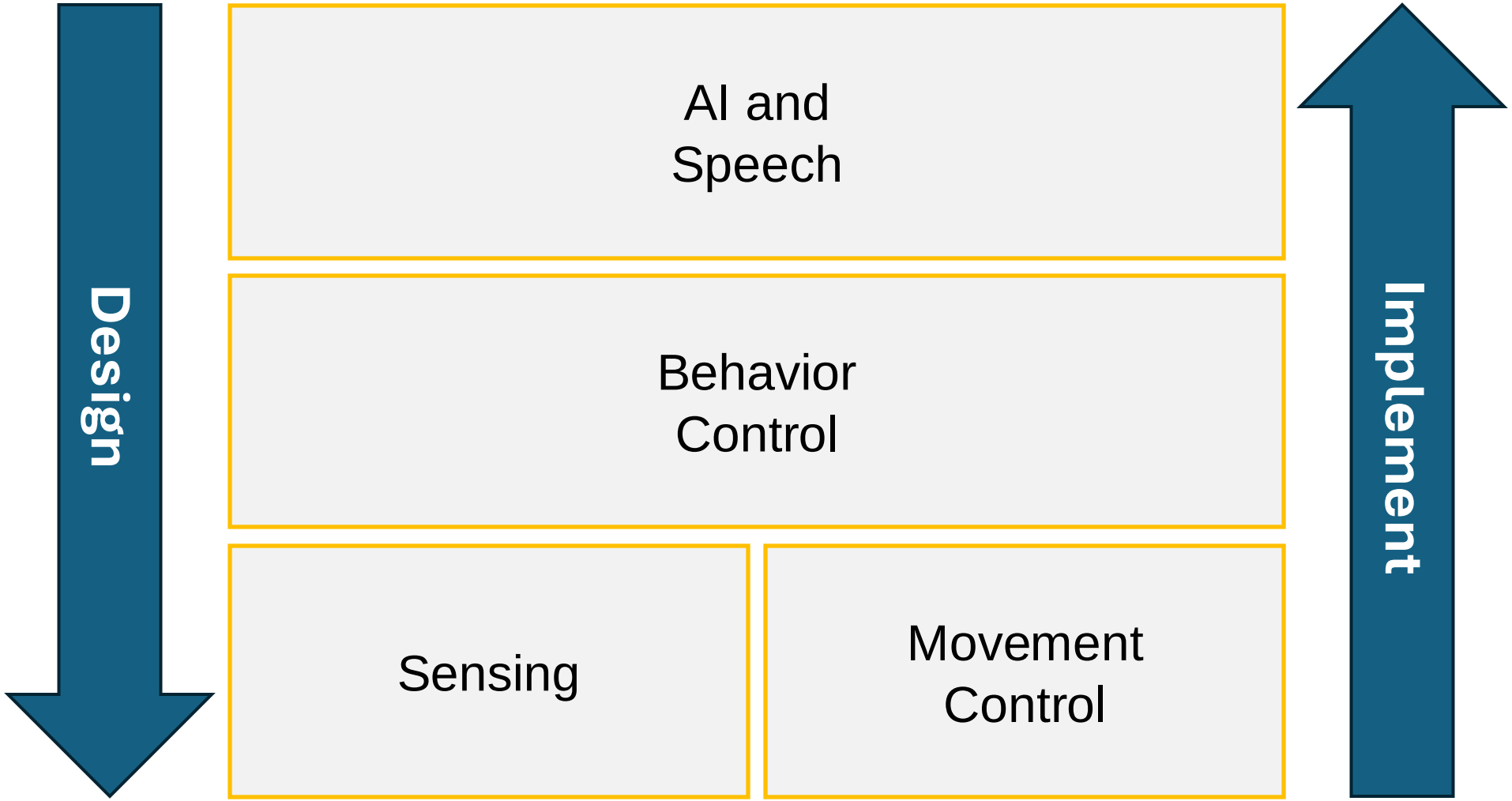
EB-6 Software

Dave Shinsel

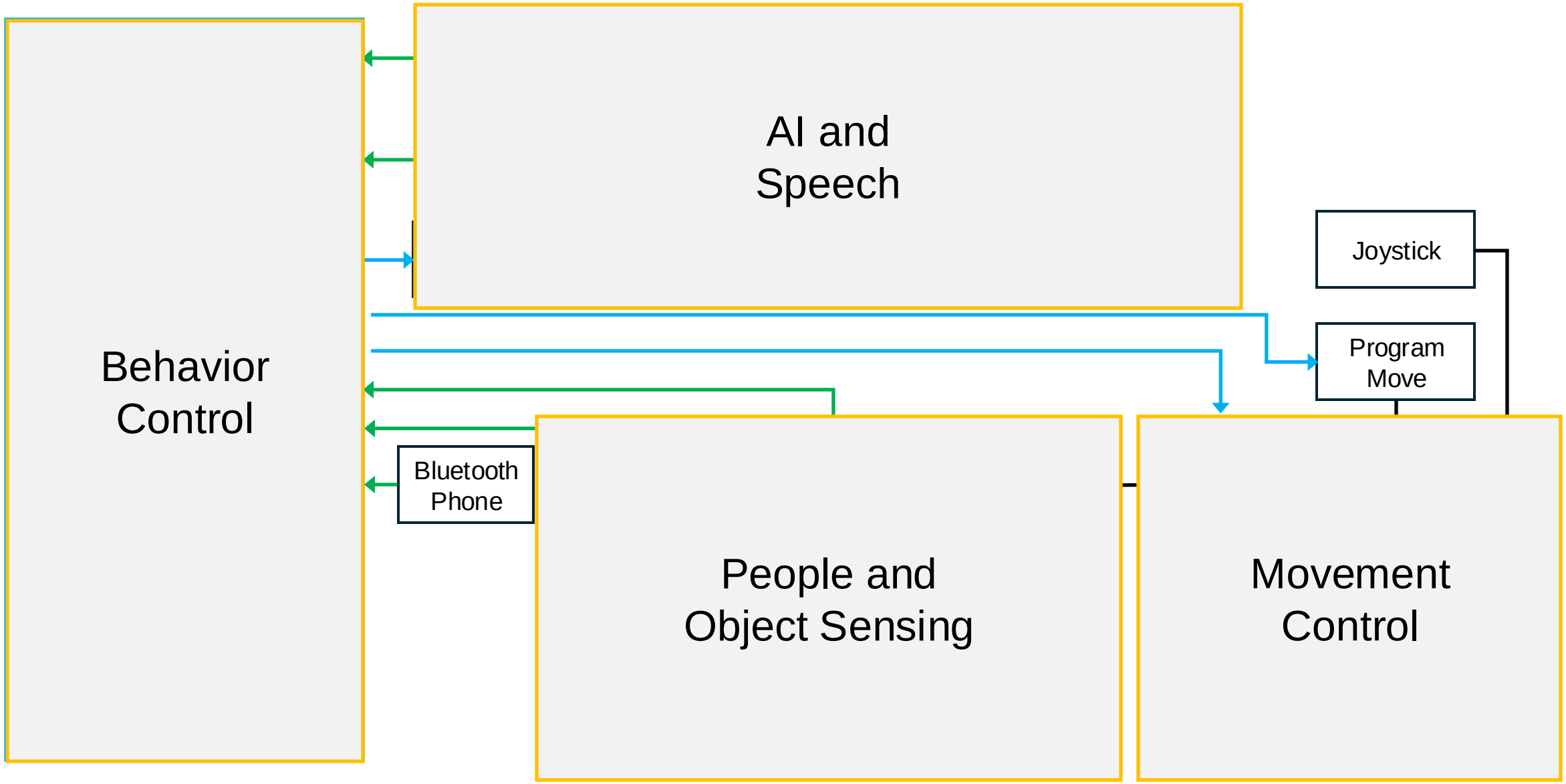
www.shinsel.com

shinsel.robots@gmail.com

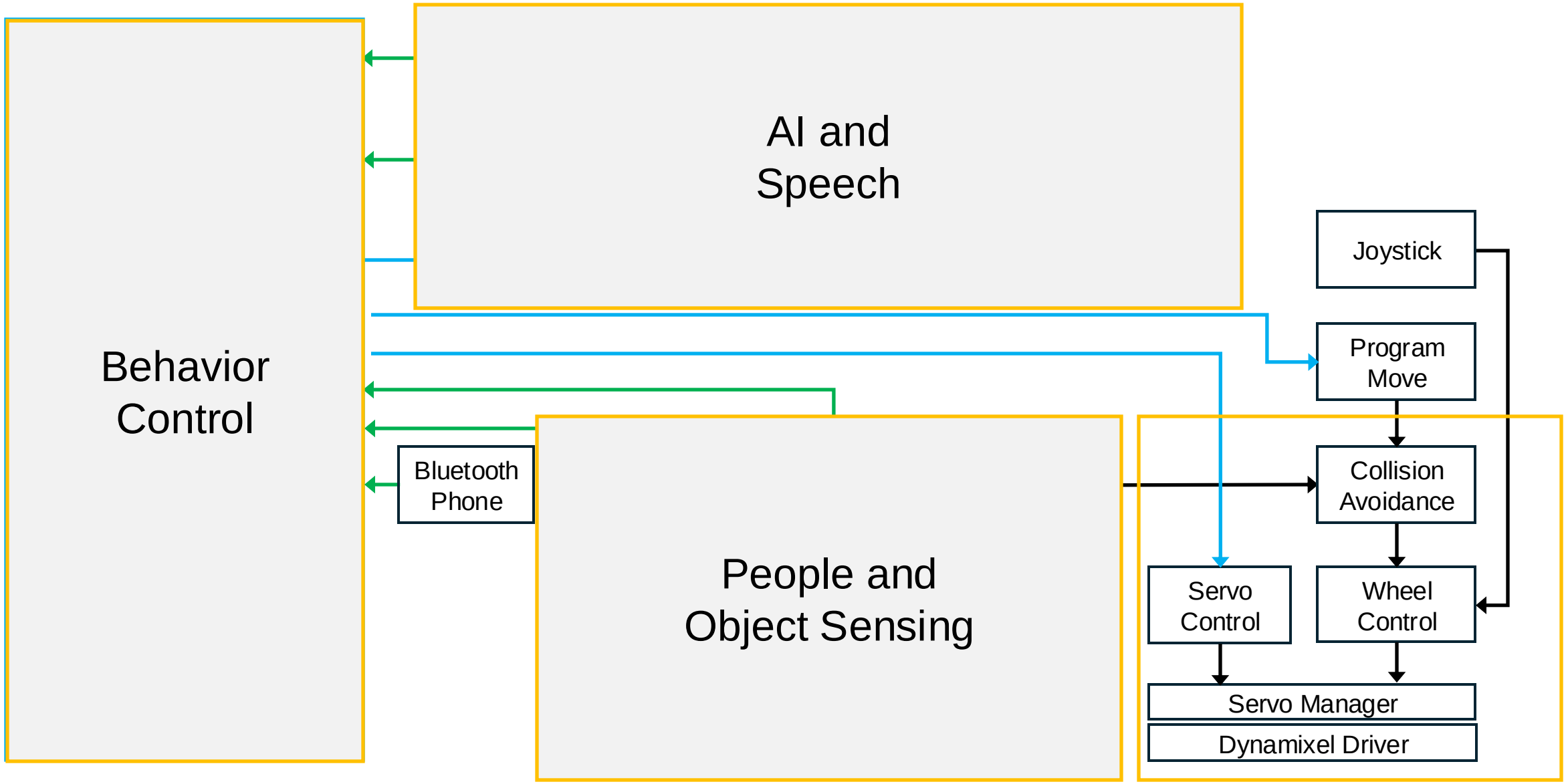
EB-6 Software Stack High level



EB-6 Software Stack High level

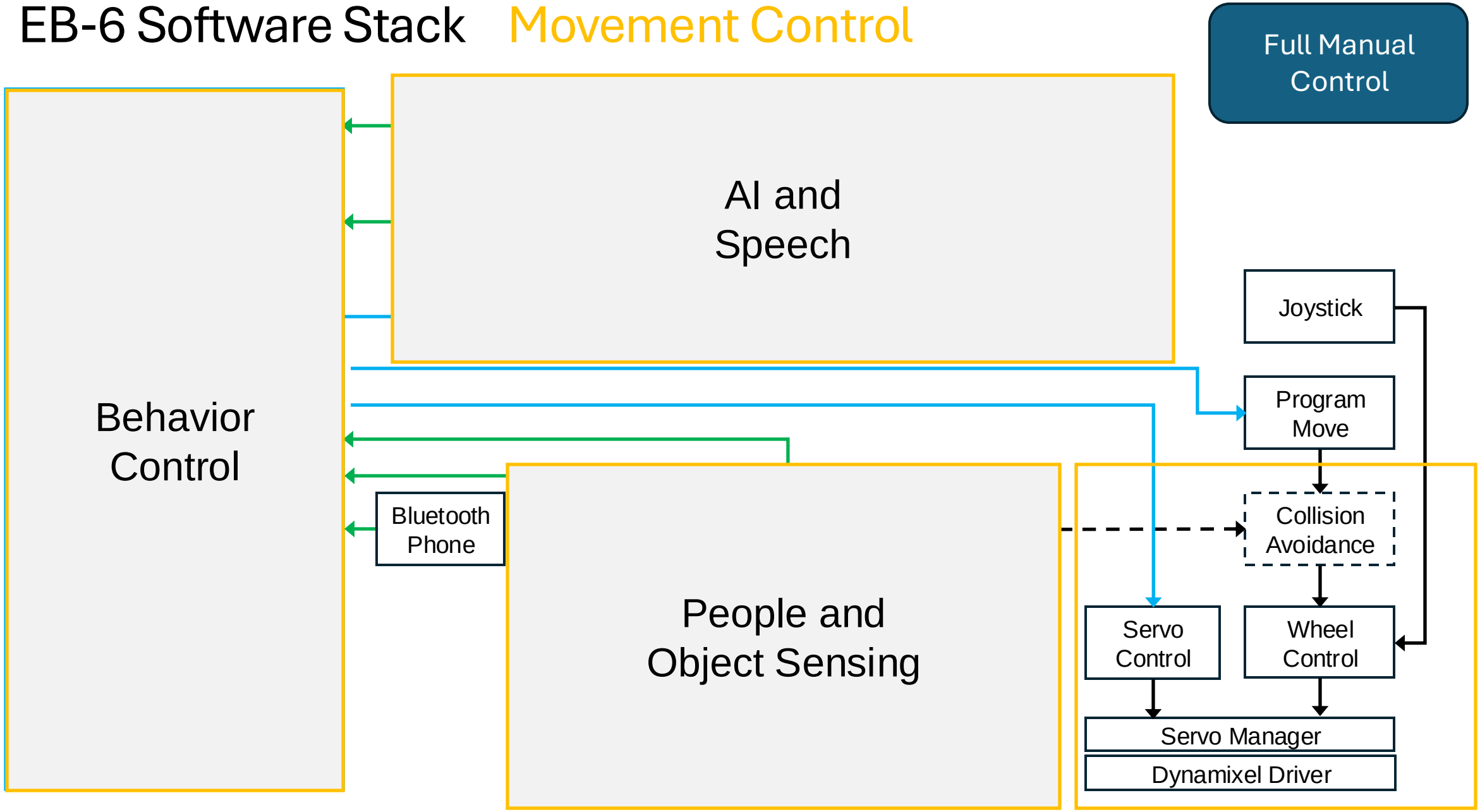


EB-6 Software Stack Movement Control

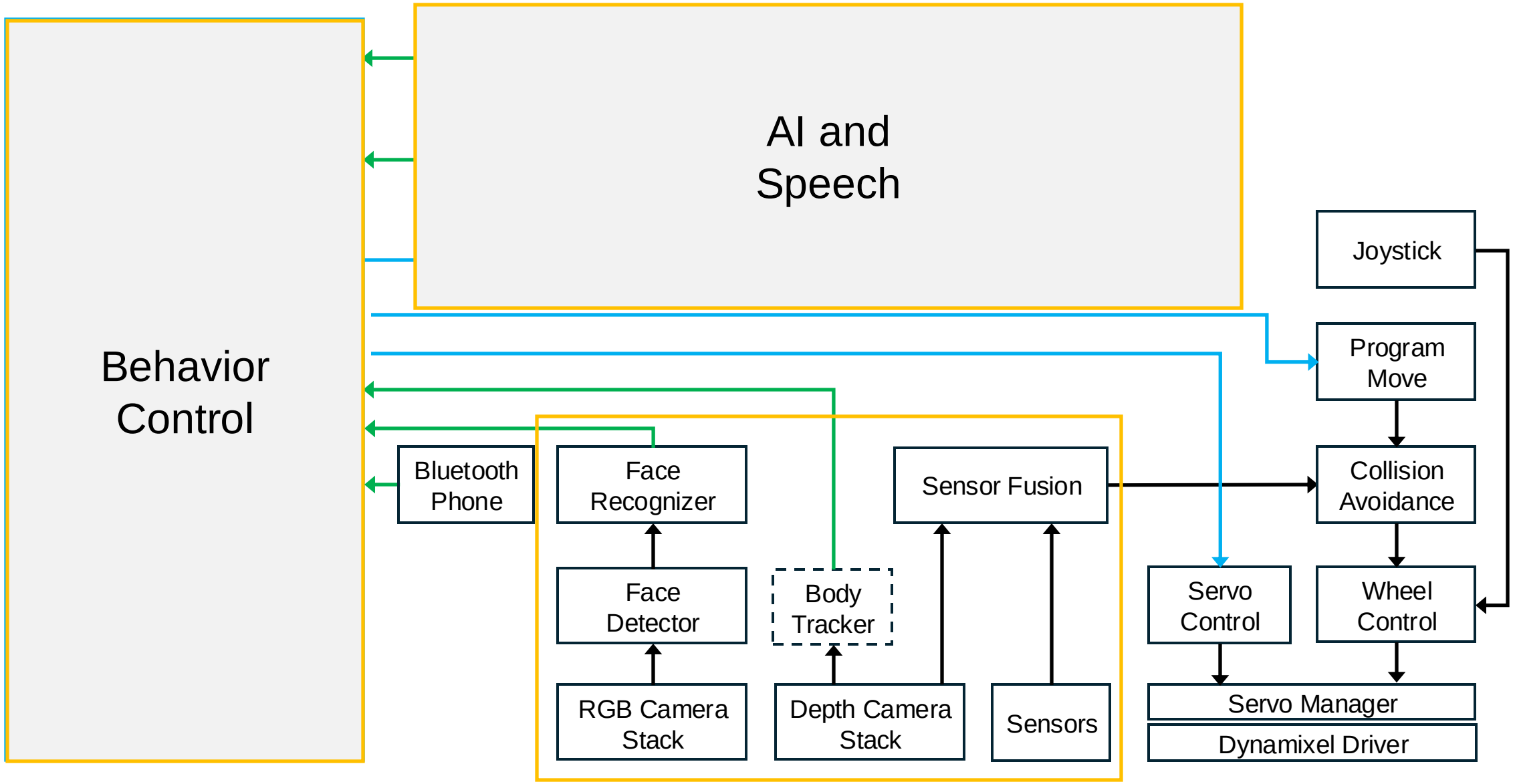


EB-6 Software Stack

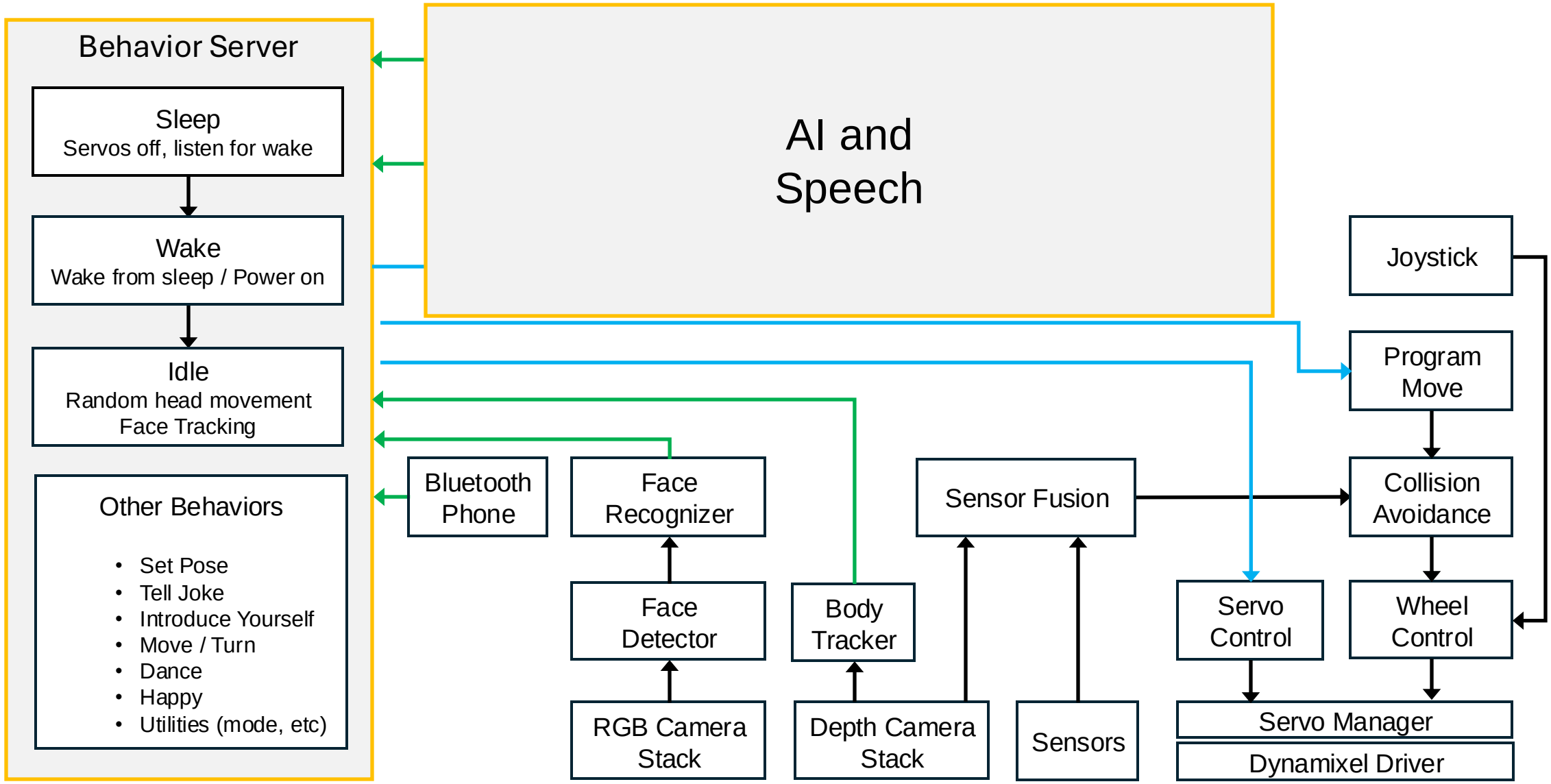
Movement Control



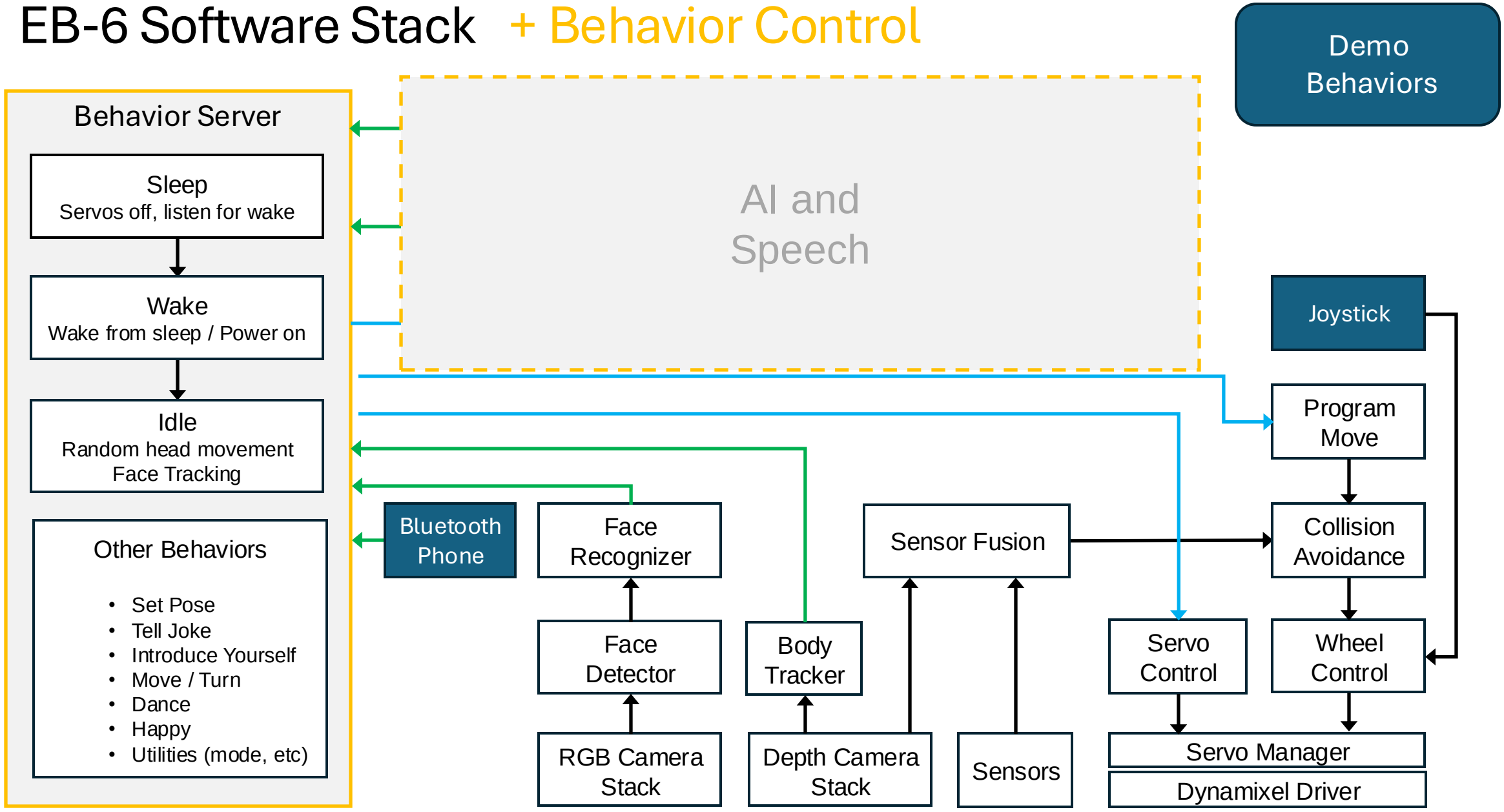
EB-6 Software Stack + People and Object Sensing



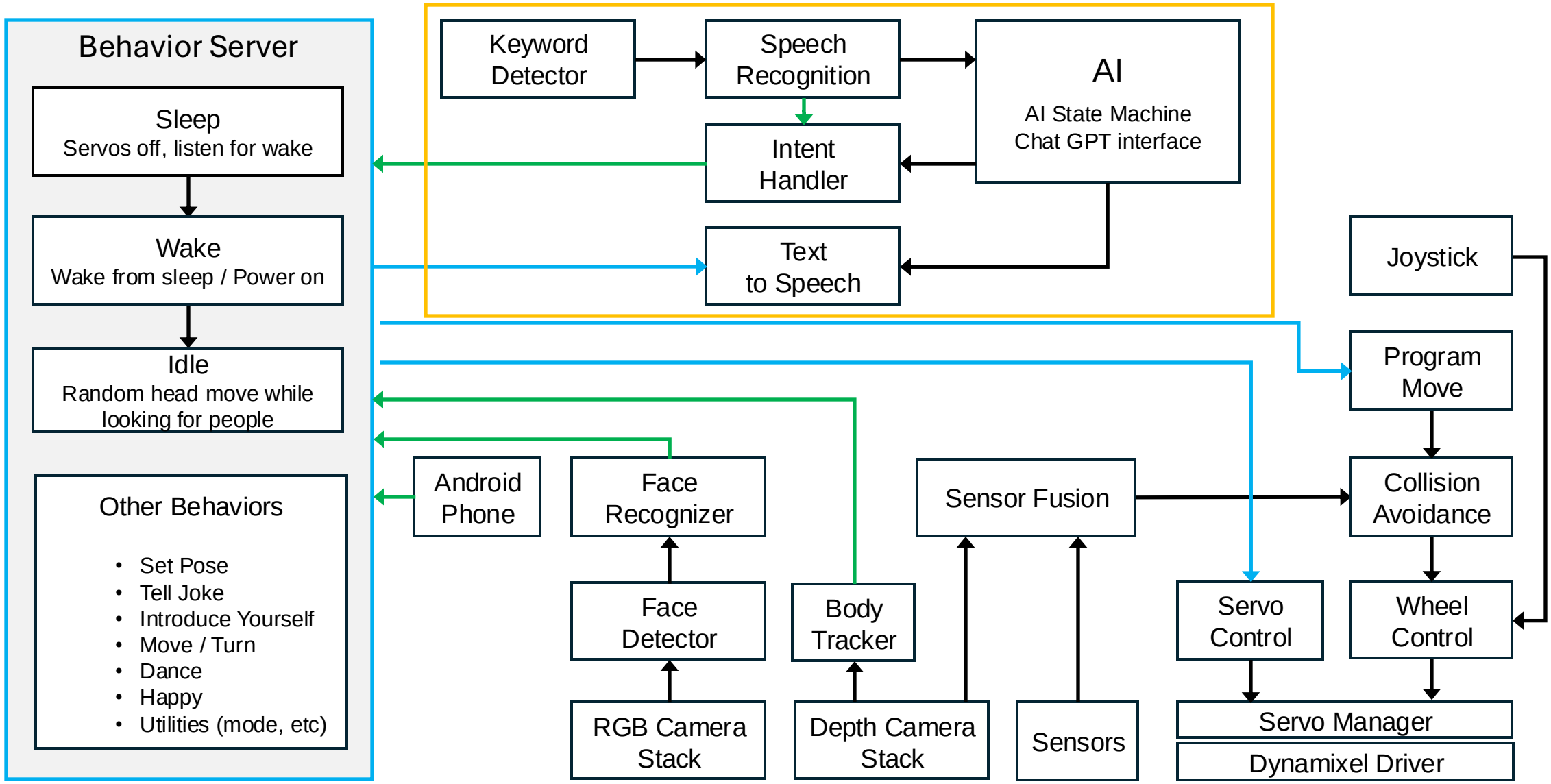
EB-6 Software Stack + Behavior Control



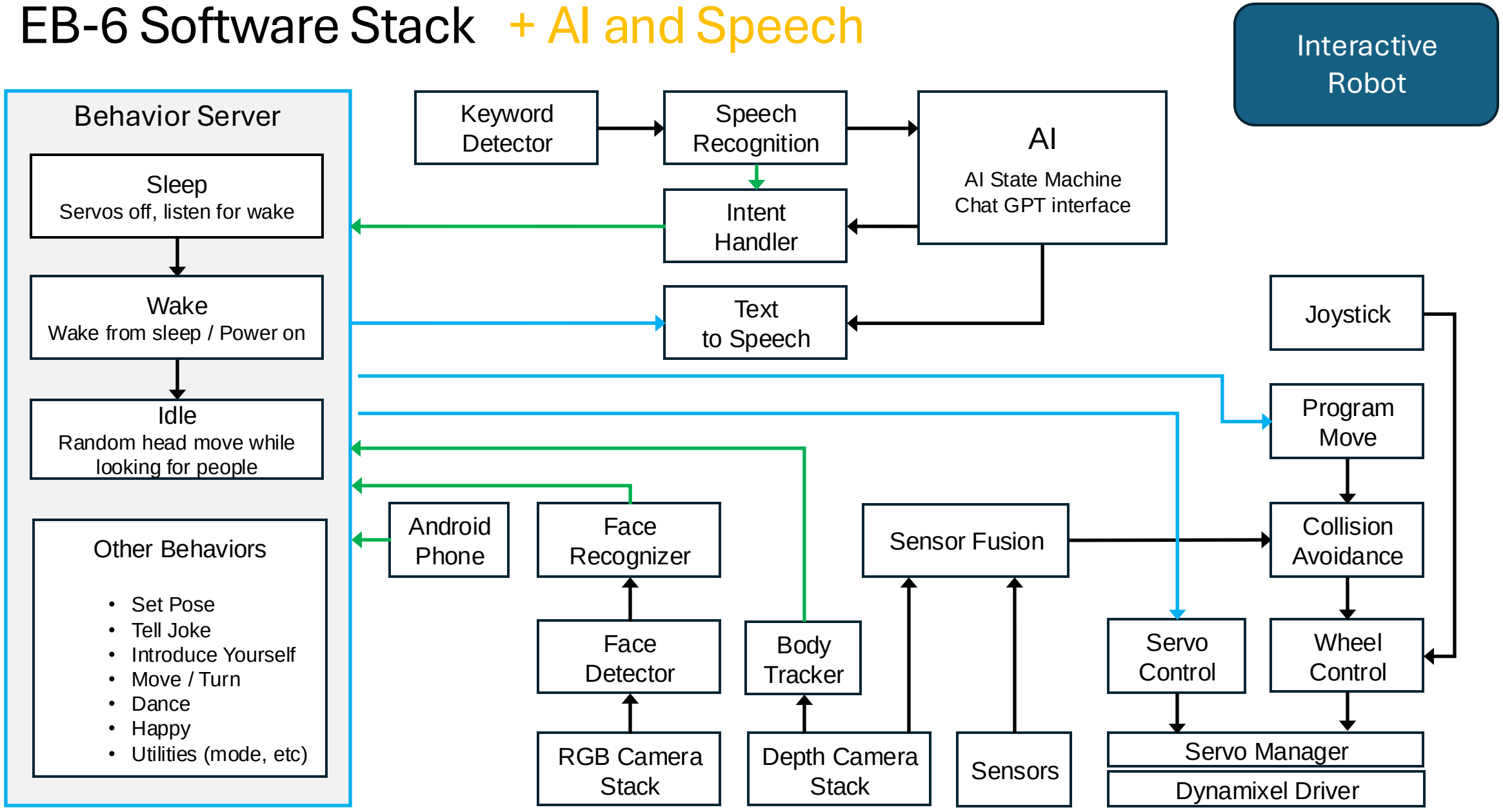
EB-6 Software Stack + Behavior Control



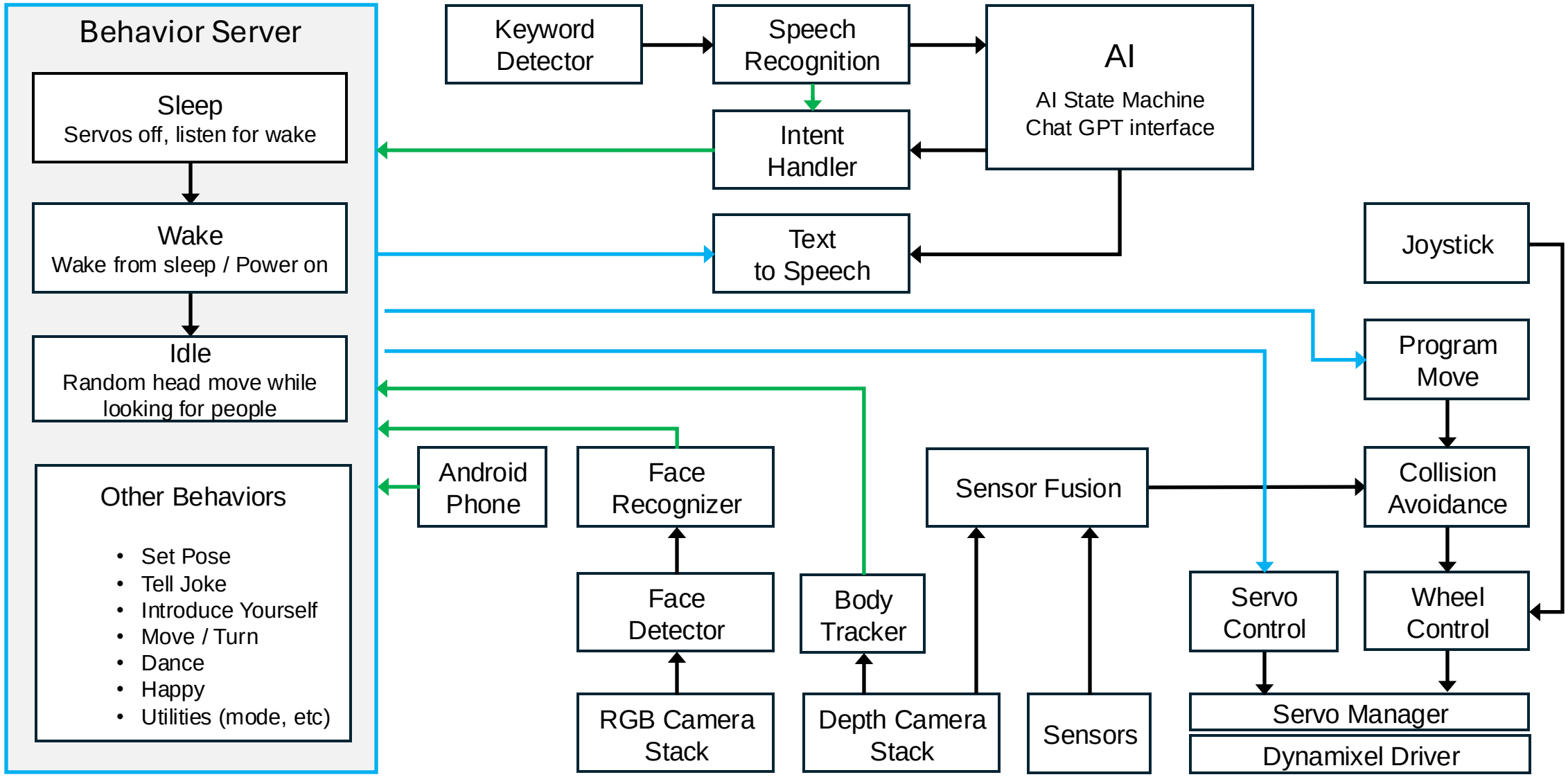
EB-6 Software Stack + AI and Speech



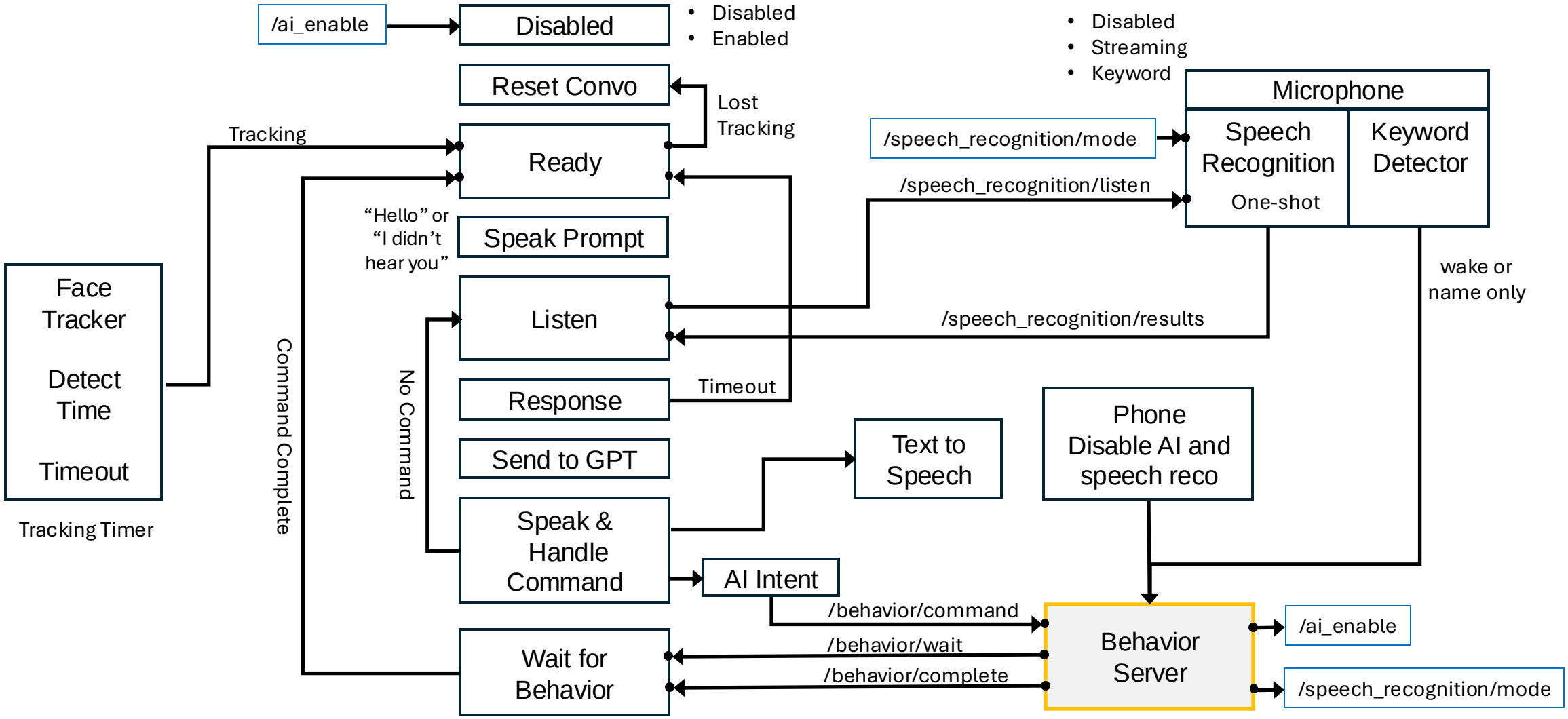
EB-6 Software Stack + AI and Speech



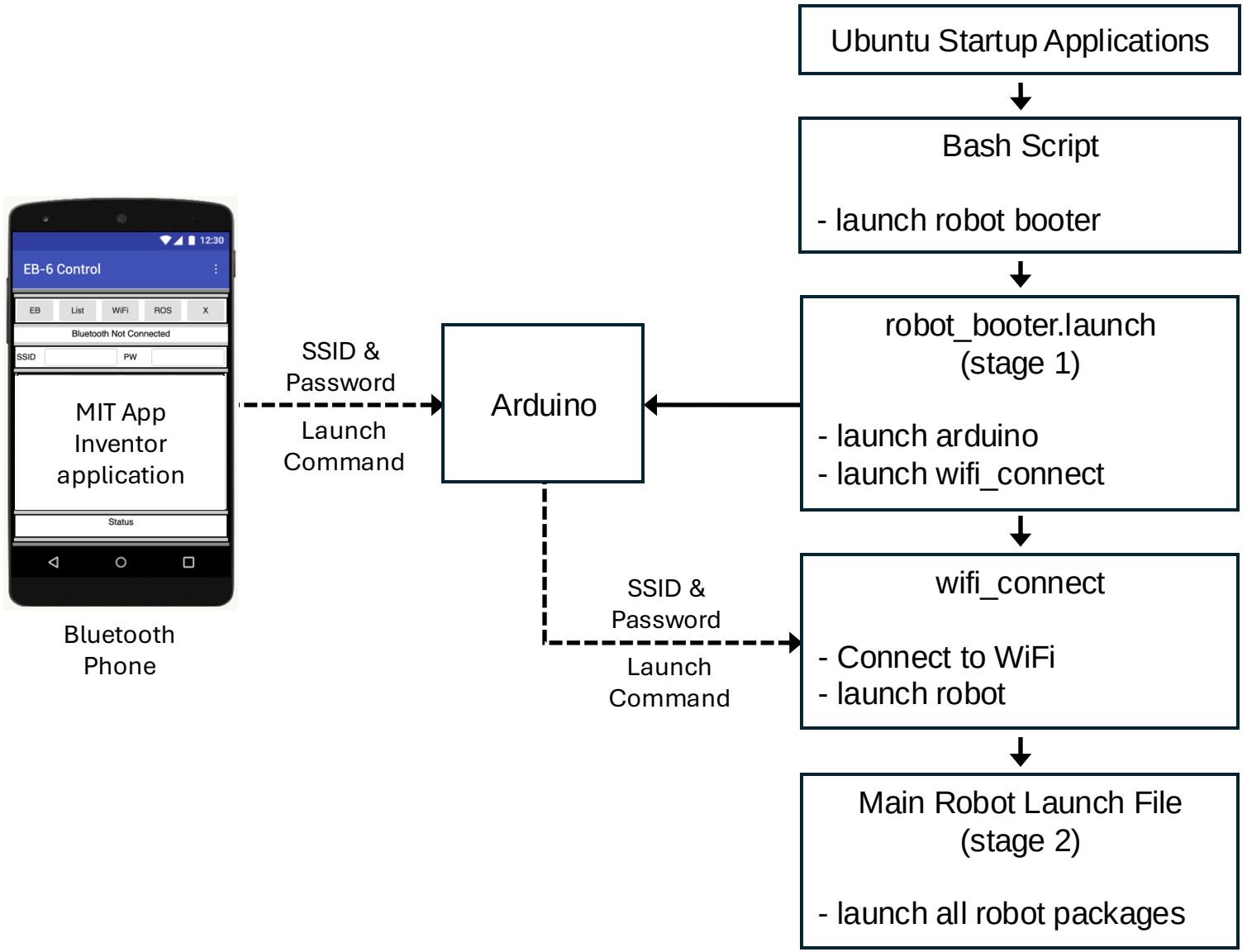
EB-6 Software Stack



EB-6 Software Stack – AI Chat State Machine

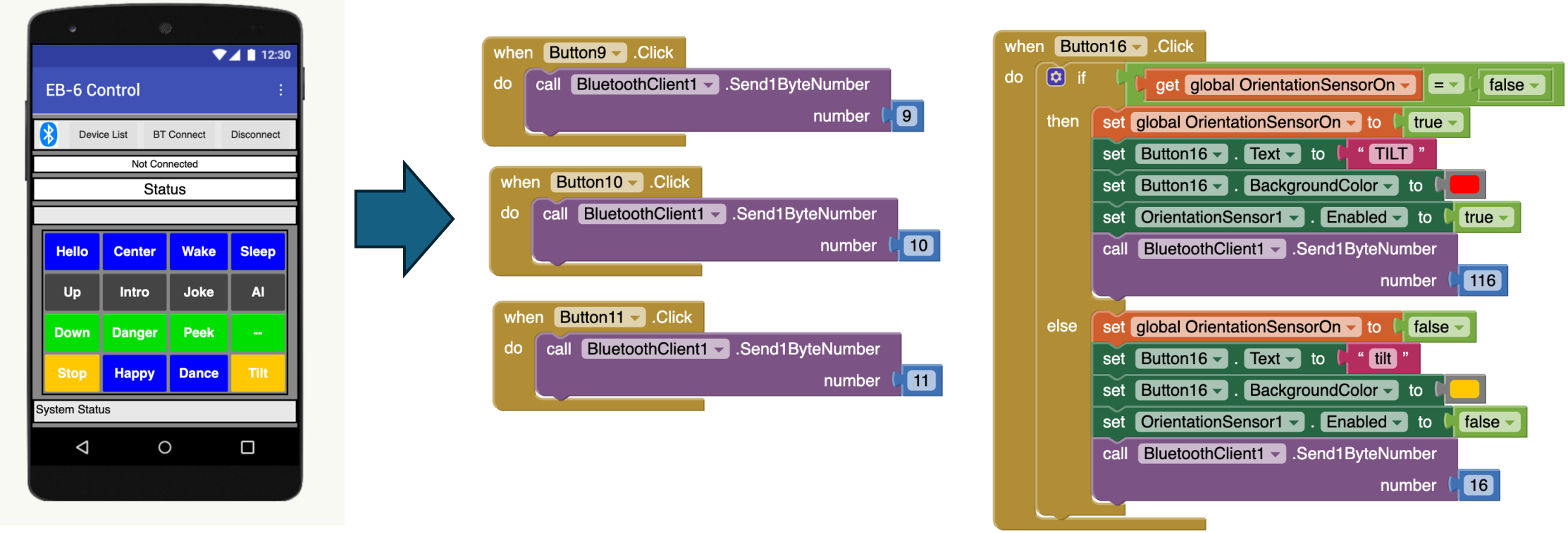


EB-6 Software Stack – WIFI Boot



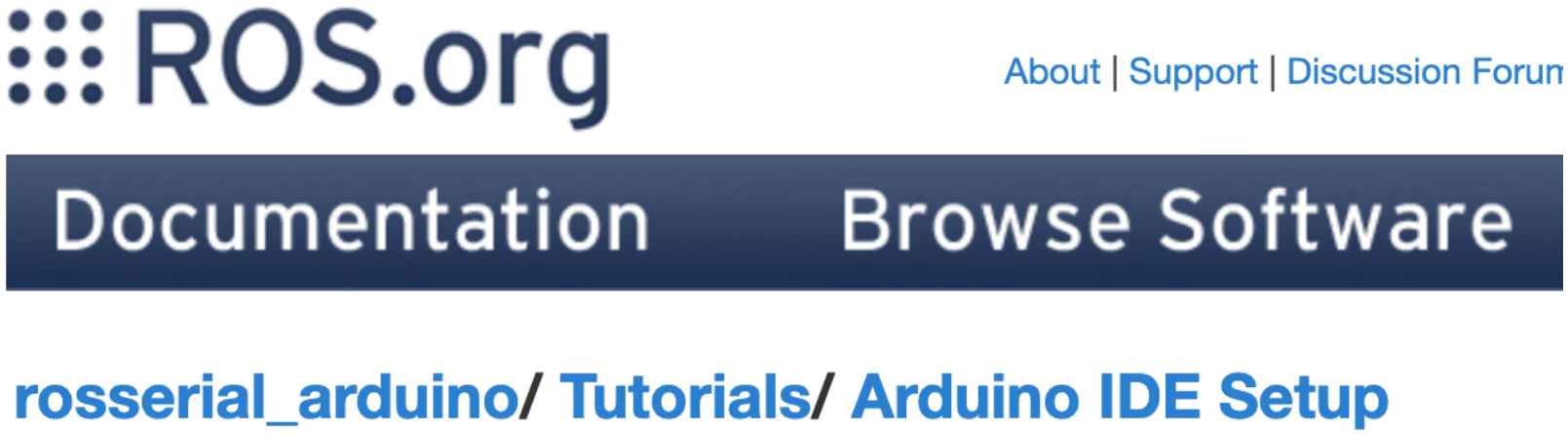
Bluetooth Phone Controller

- MIT App Inventor is great for this (Android only)
  - Free, easy, talks Bluetooth serial (to Arduino)



Arduinos in head and body

(Follow instructions on ROS webpage)



(just google “ros arduino”)

https://wiki.ros.org/rosserial_arduino/Tutorials/Arduino%20IDE%20Setup

External Packages:

1. Picovoice “Porcupine”: Keyword detector (local)
2. Cepstral Text to Speech: Robot’s voice (local)
3. Google Speech: Streaming and Batch (web)
   AI: Streaming, Keyword: batch mode
4. ChatGPT 3.1: AI Chat (web)

Utilities (not require to run the robot):

1. MIT App Inventor: Phone app creator (web)
2. NoMachine: Remote desktop
3. PyCharm: Python IDE
4. Git: Source control
5. CloneZilla: System Image backups

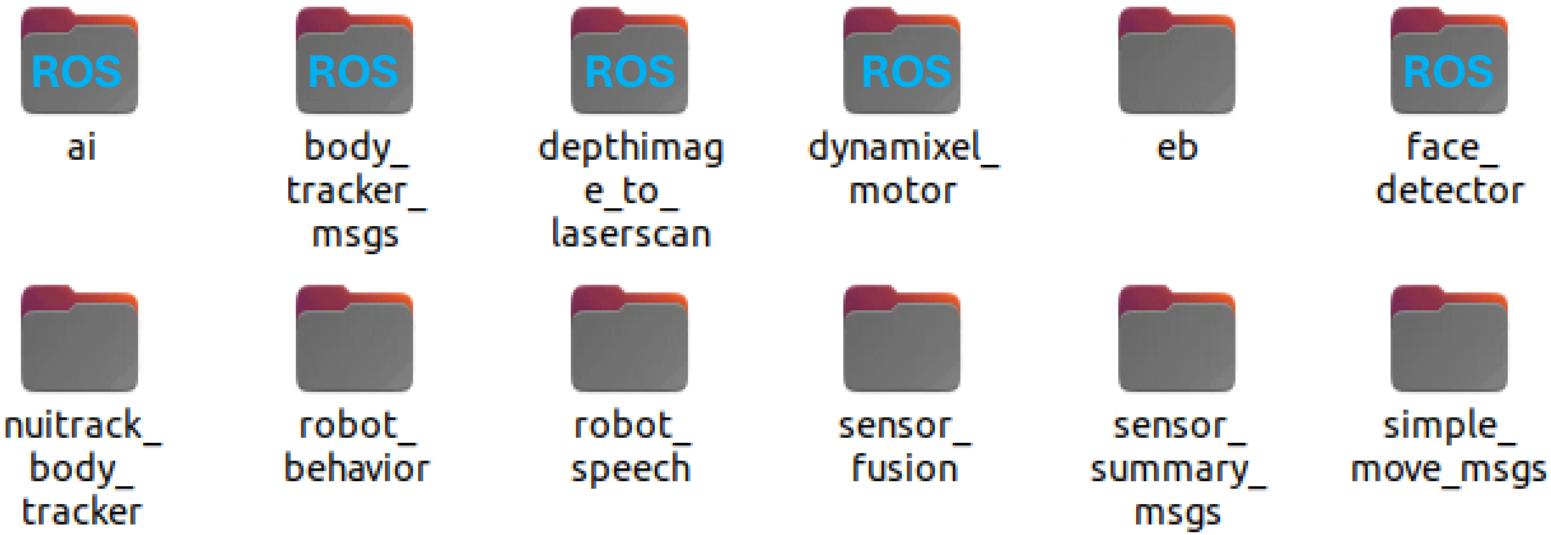
ROS (as used in this robot)

How many of you have used ROS?

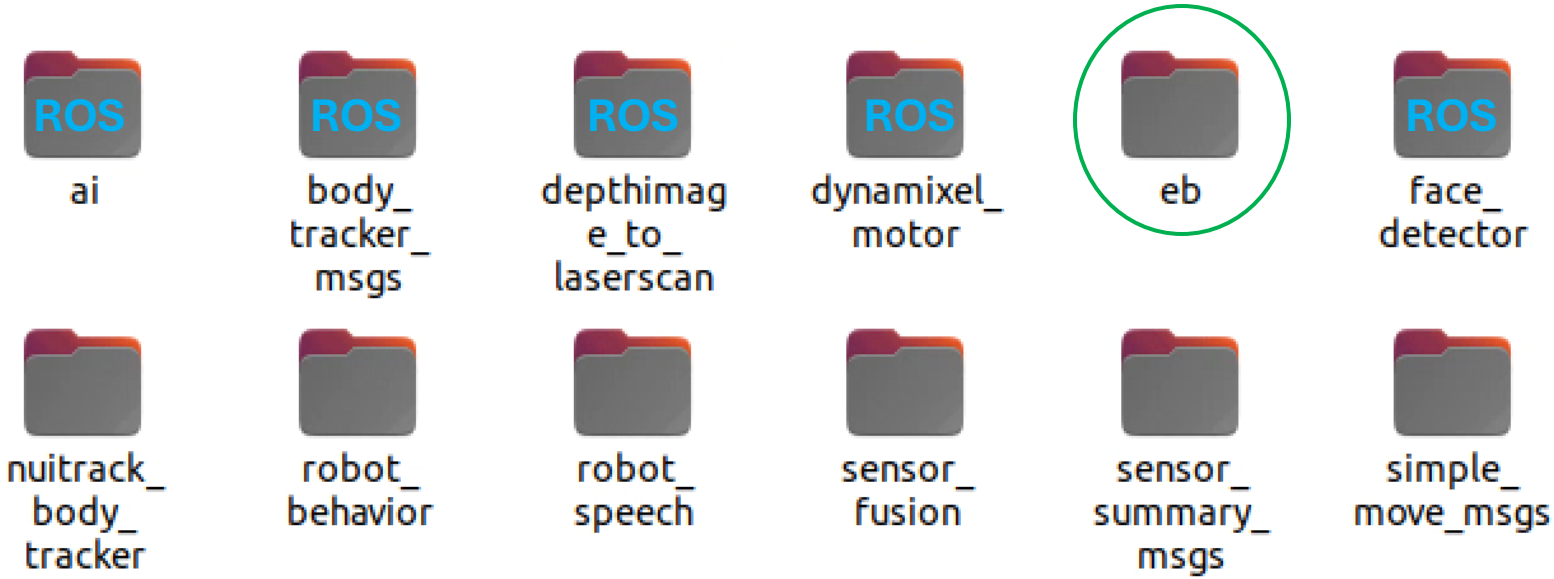
1. Framework for files / modules
2. Communication infrastructure
3. ROS / Community provided** modules:
   Dynamixel servo driver*, Face detector*,
   Arduino serial, joystick driver, Laser scan
4. Utilities for debugging

** Can be a source of bugs
* Modified

Framework: ROS directory layout
/home/catkin_robot/src:



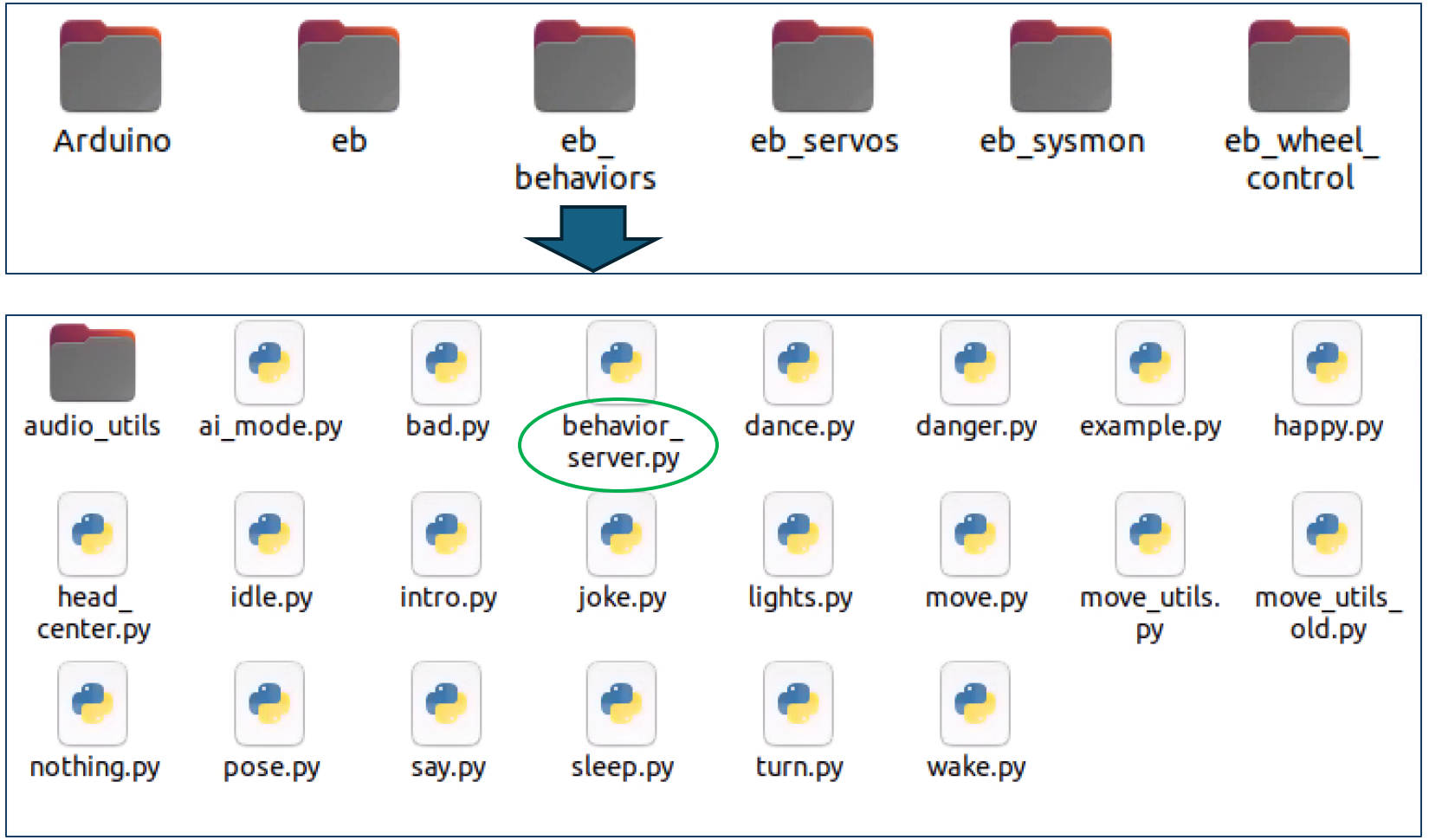
Framework: ROS directory layout → Generic Packages /home/catkin_robot/src:



Framework: ROS directory layout → Robot Specific Code /home/catkin_robot/src/eb:

Framework: ROS directory layout → Robot Specific Code

/home/catkin_robot/src/eb:



ROS Bring-up

20+ process to start…

How do you start all these programs?

ROS Launch file (first half)

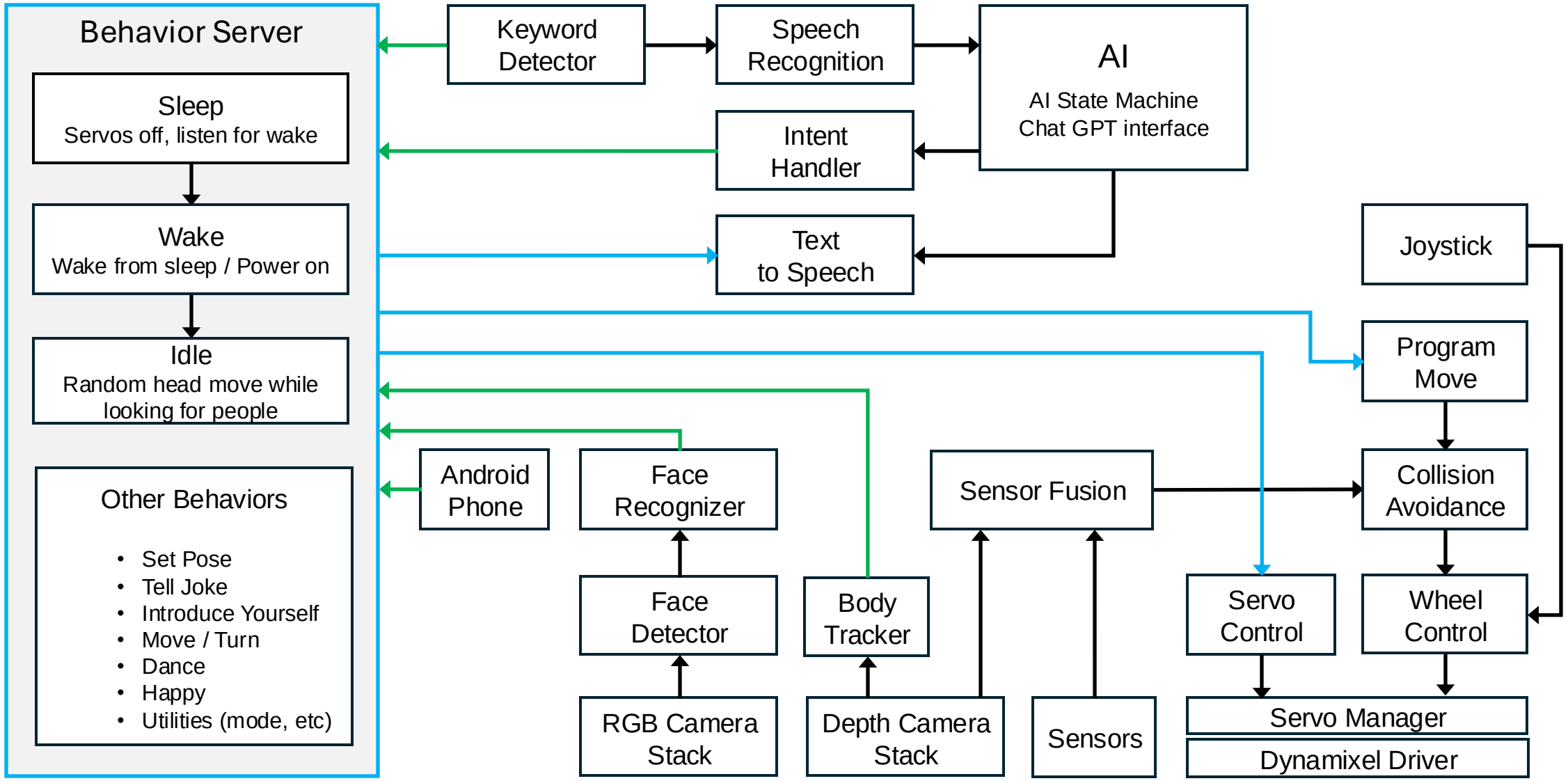
```
<!-- Head Arduino, controls eyes and ears colors -->
 <node name="head_node" pkg="rosserial_python" type="serial_node.py" output="screen">

 <!-- Body Arduino, handles bluetooth phone and publishes sensors -->
 <node name="body_node" pkg="rosserial_python" type="serial_node.py" output="screen">

 <!-- Dynamixel Servo Controllers, using modified dynamixel_motor module -->
 <include file="$(find eb_servos)/launch/servos.launch"/>

 <!-- Cameras -->
 <include file="$(find eb)/launch/include/rgb_1920_eye_camera.launch"/>
 <include file="$(find eb)/launch/include/rs_848_depth_only.launch"/>

 <!-- Sensor fusion -->
 <include file="$(find eb)/launch/include/depthimage_to_laserscan.launch"/>
 <include file="$(find sensor_fusion)/launch/sensor_fusion.launch"/>

 <!-- Wheel motor controller and collision avoidance -->
 <include file="$(find eb_wheel_control)/launch/eb_wheel_control.launch"/>
```

Launch file (second half)

```
  <!-- Text to speech (robot voice) and sound effects services  -->
  <include file="$(find robot_sounds)/launch/robot_sounds.launch"/>

  <!-- Command Intent Handler (used by command mode and AI) -->
  <include file="$(find speech_handler)/launch/speech_handler.launch"/>

  <!-- Face Detector and Person Recognizer -->
  <node name="face_detector_node" pkg="face_detector" type="face_detector.py" required="true"
output="screen">

  <!-- Behavior controller  -->
  <node name="eb_behaviors" pkg="eb_behaviors" type="behavior_server.py" required="true"
output="screen">

  <!-- AI -->
  <node name="ai_node" pkg="ai" type="ai.py" output="screen">

  <!-- Dashboard for system monitoring and control -->
  <include file="$(find eb_servos)/launch/dashboard.launch"/>
```

Backup

EB-6 Software Stack



EB-6 Hardware - full block diagram