



OpenPilot System Architecture Analysis

by: Team Horizon

YORK 

Agenda

1. OpenPilot Overview

5. Concurrency

2. Features of OpenPilot

6. System Evolution

3. Components of OpenPilot

7. Use Cases

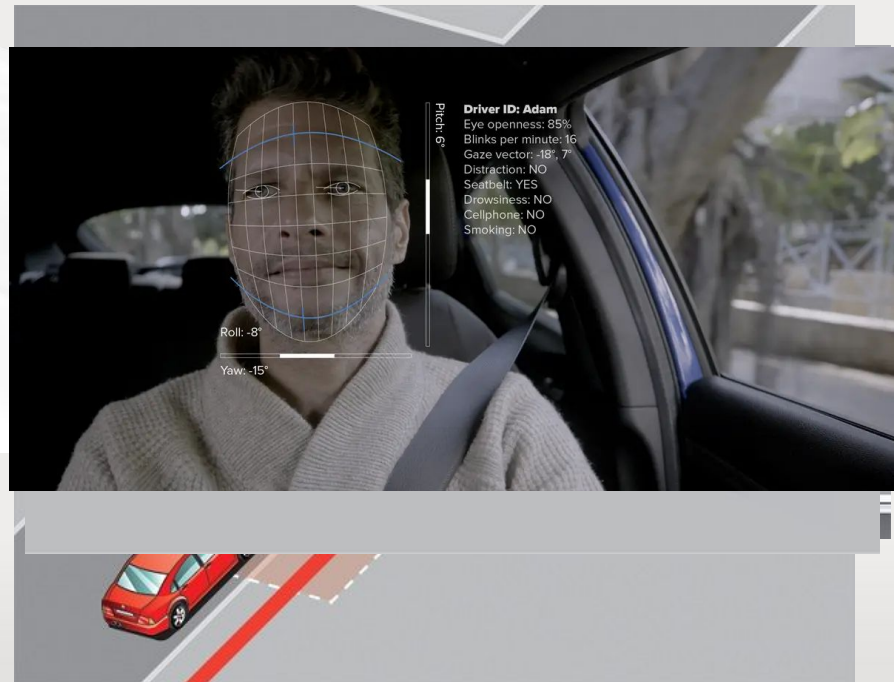
4. Architecture Styles

8. Lessons Learned



Functionality of the Top Level Systems

Overview & Features of OpenPilot



- Adaptive Cruise Control (ACC)
- Automated Lane Centering (ALC)
- Forward Collision Warning (FCW)
- Lane Departure Warning (LDW)
- Driver Monitoring (DM)

An aerial photograph of a university campus. In the background, a large, multi-story building with a central tower and many windows is visible. In the foreground, there is a large, green lawn with several trees and a few people walking. The sky is blue with some clouds.

Identifying Subsystems

Subsystems

- Sensors
- Actuators
- Neural Network Routers
- Localization
- Calibration
- Controls
- Logging
- Misc. services
- Hardware (panda/comma)

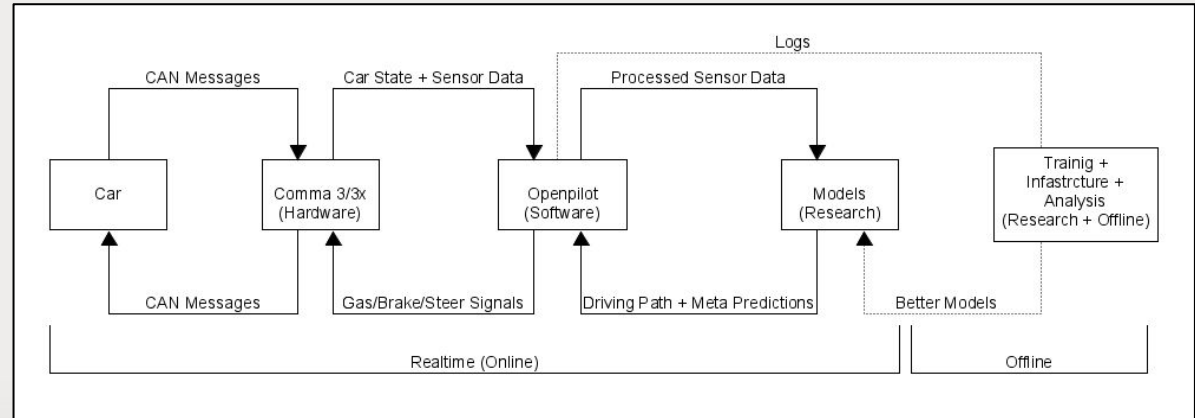


Architecture Styles

*Expected Architecture Styles Based on
Documentation Research*

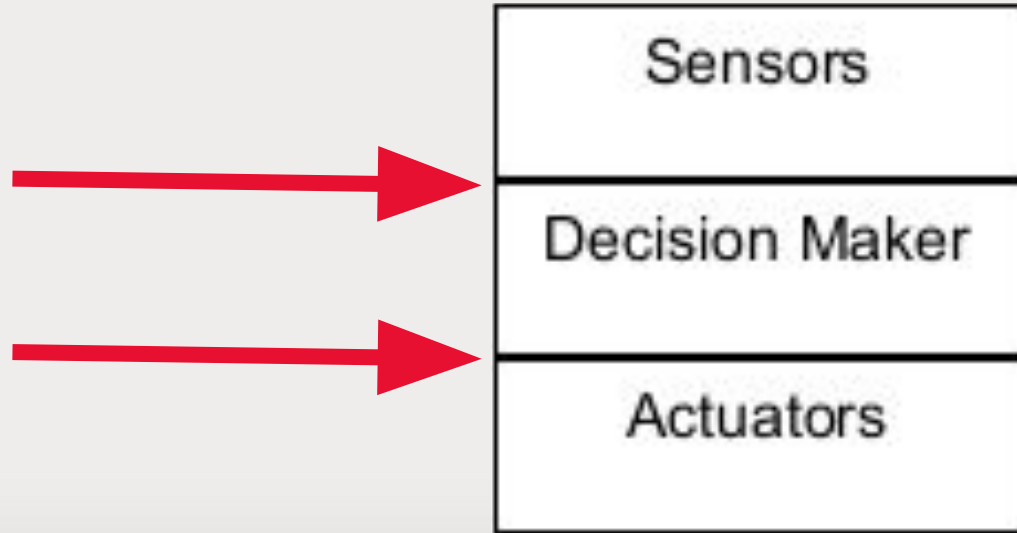
Layered

- Layers don't communicate with non-adjacent layers.
- Each layer has its own domain.
- Lowest level being vehicle hardware (OBD-II), highest being Application.



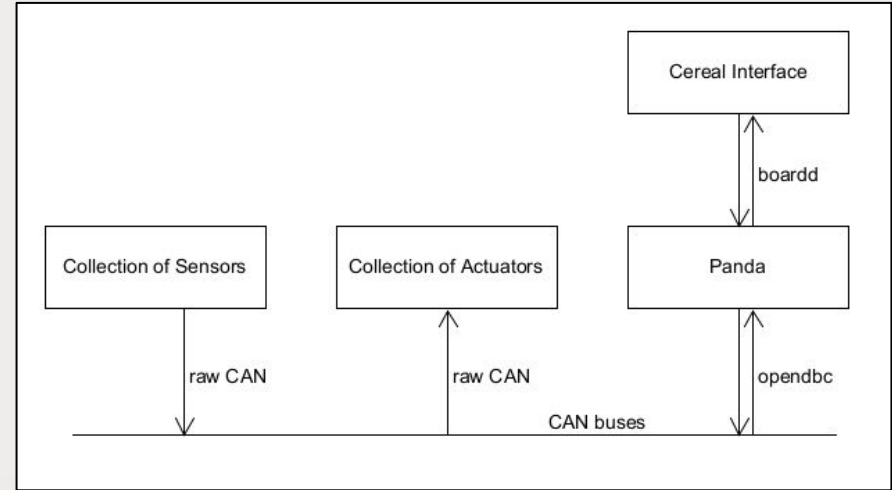
Layered - Communication

- CAN buses
- Panda
- opendbc
- boardd
- Cereal



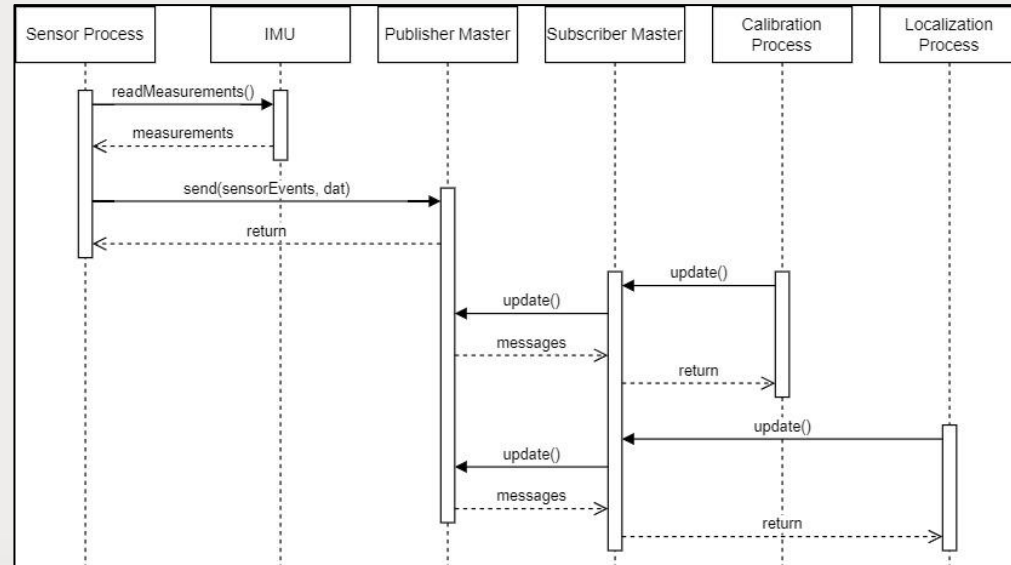
Implicit Invocation - Event Based

- Main form of communication
 - Sensors
 - Actuators
- Loosely coupled components
- Event busses



Implicit Invocation - Publish & Subscribe

- Cereal
 - Robotics systems message exchange specification
 - Interprocess communication library
 - Sensor processes read data from sensors and publish it in packets.
 - Other processes subscribe to those packets and can retrieve published data.



Process Control - Closed Loop Feedback

- Maintains a specific parameter at a desired value.
- Closed Loop: Uses current value to determine which adjustments are required to meet desired value.

OpenPilot

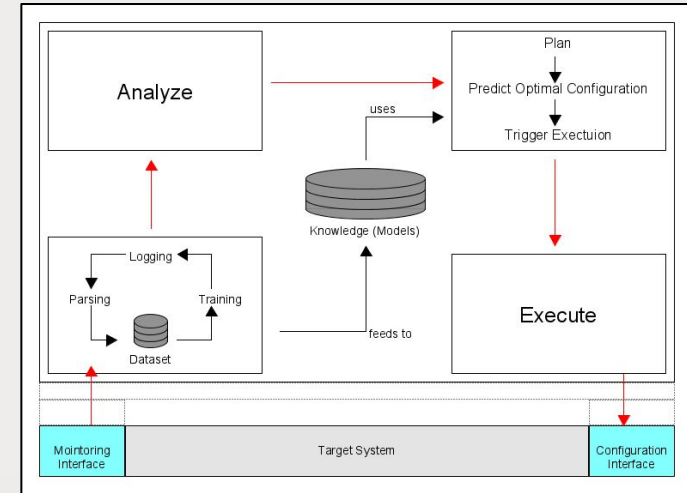
- **Adaptive Cruise Control (ACC):** Maintains desired speed. Read current speed and make adjustments via accelerator/brake.
- **Lane Keeping Assistant System (LKAS):** Maintains vehicle in desired position. Reads current position using cameras/sensor, and makes adjustments via steering.

Process Control - MAPE-K

- **Monitor-Analyze-Plan-Execute** over shared **Knowledge**
- Variation that uses a ML Model as Knowledge

OpenPilot

- **Laneless Mode:** Uses ML to predict 'where humans would normally drive'.
- Not informed about traffic laws, intersections, lanes, etc.
- Goal is to create smoother, more comfortable driving, as well as allow for unpredictable scenarios



Prediction as to how they incorporate ML, without viewing Source Code.

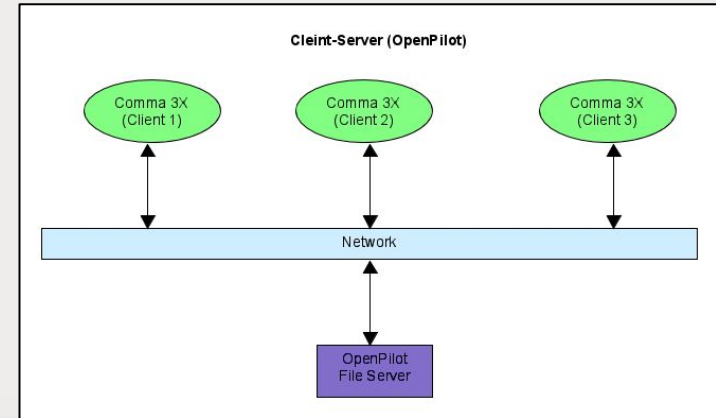
[Image Source](#)

Client-Server

- Standalone remote component (server). Server accessed by clients that make call via network.

OpenPilot

- By default, all driving data is uploaded to their servers, and used to train ML models to improve OpenPilot.
- Server accessed by Comma 3X (client) via LTE or WiFi



Other Styles

- Pipe and Filter

- Very likely to have internal pipelines to handle data, as many diagrams include ordered sequence of events. (Sensor input becomes actuator output).
- However, need to ensure they are stateless filters, unaware of up/downstream, etc. (via Source Code).

- Repository

- Must be database somewhere when data is uploaded to server and used for training models.
- Did not find any documentation about specific Repository.
- Will likely discover more for Assignment 2.

An aerial photograph of a university campus. In the foreground, a large, well-maintained green lawn is visible. To the left, there are several modern buildings with large windows. In the center, a large, multi-story building with a prominent central tower and a clock face is visible. The campus is surrounded by trees and greenery. The word "Concurrency" is overlaid in large white text on the left side of the image.

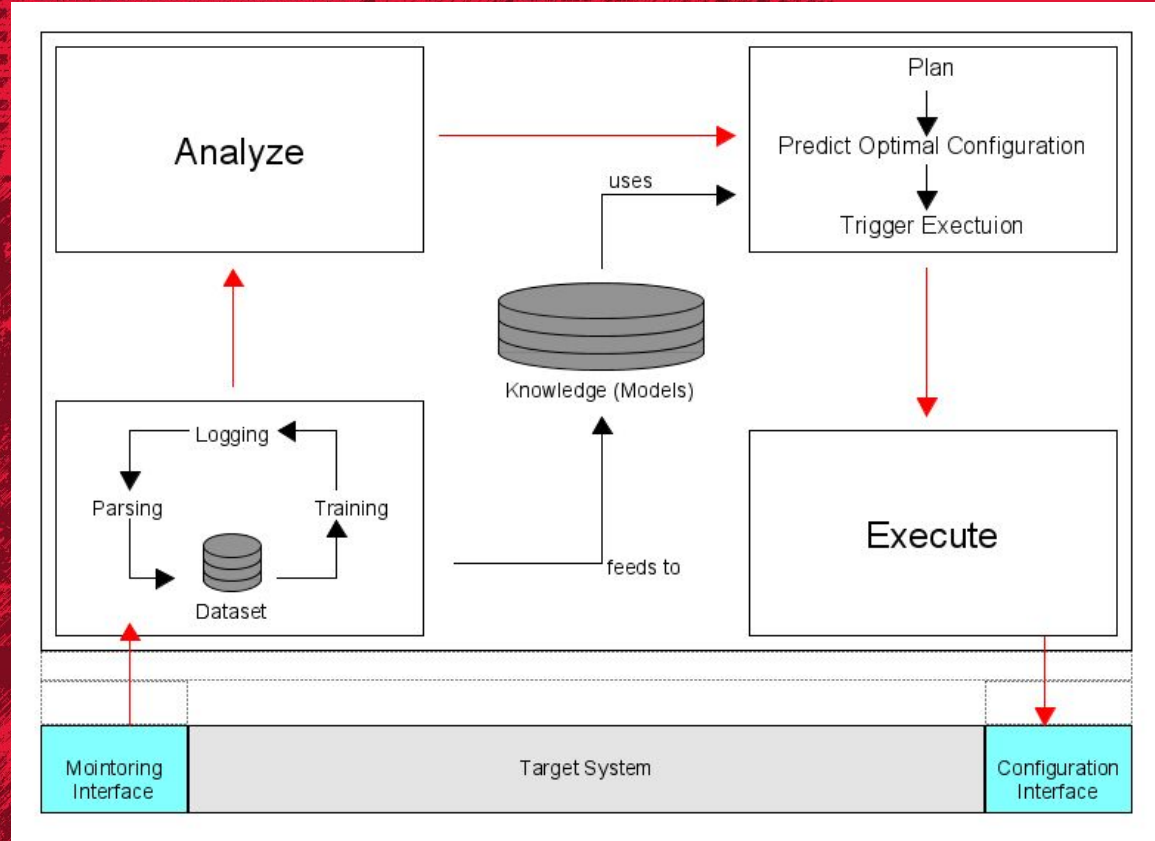
Concurrency

Concurrency

- Receives input data from cameras, radars, and vehicle sensors simultaneously
- Parallel decision making processing from different sensors
- Overlapping sensor input for diverse range of functionality
- Theoretical Example: Automated Lane Centering and Lane Departure Warning
 - Share similar input data however use separate decision making components for separate purposes

Concurrent processes occurring

Decision making model demonstrates that there are sub processes happening within the decision making process to deliver lane features.



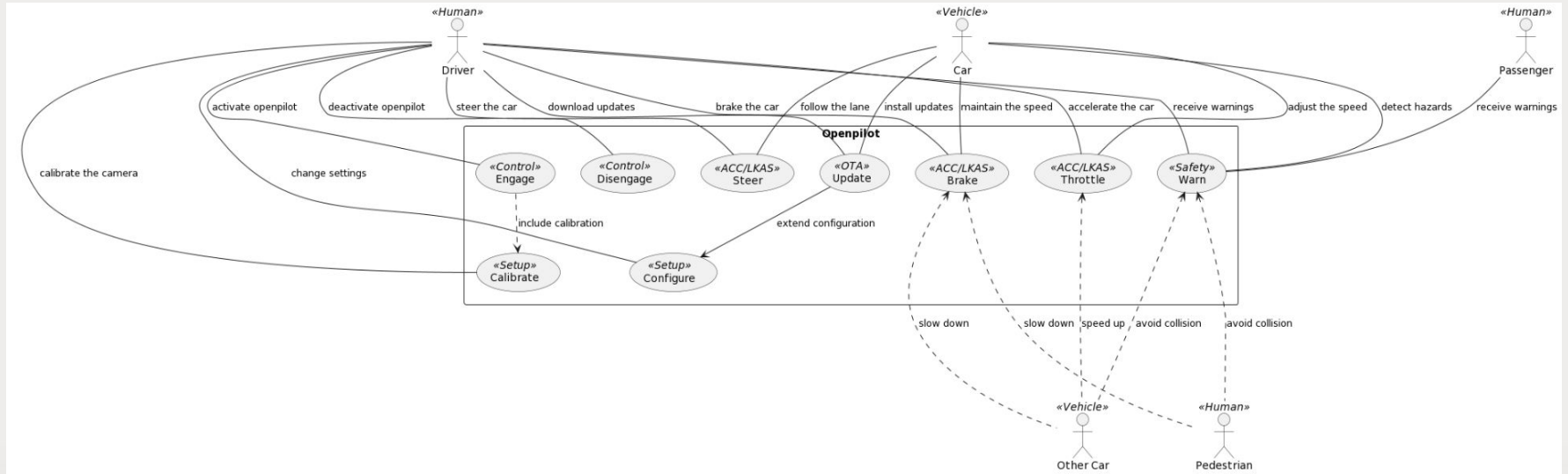


Key Findings and Discussion

System Evolution

- Structural release process
 - Multiple iterations enhancing and adding features
 - Example: 0.9.5 release
- Open Source Nature
 - Allows any developer to view, modify, or enhance the project
 - Enables continuous improvements through community contributions
- Contribution of internal employees
 - Diverse team such as Full Stack Developer, Car Interface Engineer, Production, etc
- User Involvement
 - Uploading specific data during usage
 - Leveraged by Openpilot team to improve and train better models

Use Cases



Lessons Learned

- Comma 3/3x (2 for older releases) and compatible car required
- Not fully self-drive and limits
 - Cannot check if lane change is safe - driver must do this.
 - Requires driver awareness - otherwise alerts, or eventually will slow to a stop.
- Weather conditions
 - Not verified to function as expected in low-light, rain, fog, or bright oncoming headlights.
- Legality
- Telemetry and privacy
- Updates may introduce new bugs
- Interfere with manufacturers systems



Thanks for your attention!

Any questions or suggestions?