Formal Technical Review

05/02/2021

Yegor Kozhevnikov - *Review Leader and Recorder*	Reject
Christopher Bowden - *Reviewer*	Reject
Nolan Vernon - *Reviewer*	Reject

Security

- System is never connected to a network
    - Difficult/tiresome to update/extract data from (*Defect*)
    - Impossible to breach from outside of the train, must also know valid credentials
- Credentials
    - Stored in an unencrypted file, with instructions how to forge an entry (*Defect*)
    - File can be viewed from the terminal if the IoT program is forcefully quit
- Program can be forcefully quit via SIGINT keyboard shortcuts (*Defect*)

Performance

- Very little resources are required to run the program
    - Very little graphical fidelity
    - No assets to be loaded on startup
- Approach allows for many more types of sensors with very little downtime
    - Further improvements can be made to eliminate this problem altogether
    - Sensor data can potentially be immediately identified which type of sensor it is meant for
- Default refresh rate for router update is set to 1.5 seconds
    - Going lower is possible at the expense of the user having to read very quickly (*Defect*)
    - Ruins human interaction
    - A way to improve this is to have the system update only every N amount of router cycles

Reliability

- Physical sensor reliability is unknown
- Error handling
    - Every command that can throw an error is within a try-catch block
    - Some blocks handle errors better than others (*Defect*)
- Logging
    - Every event is recorded to a dedicated log file
    - If none is detected, one is made

    - If the file is removed or renamed during operation, logging events are halted (*Defect*)

General Code
- Some classes use float while others use double without a specific reason
- Some Sensor classes have getter methods but are never used by any program
- Parameters are unmodifiable without having access to the source code, and recompiling the entire program. (*Defect*)

Sensor.java
- *no critique found*

GPSSensor.java
- If GPS misses an update and doesn't refresh the current coordinates, the IoT will report that the train did not move in the past cycle (*Defect*)
- Will cause IoT to calculate the speed of the train to be 0
- Will guarantee to generate a false Major Wheel Slippage Alert

DopplerRadarSensor.java
- Assumes that the installed sensor will use EM waves without the ability to use acoustic waves (speed of the wave is hardcoded to be the speed of light)

IceFormationSensor.java
- Assumes ice formation possible on exclusively two parameters
- Ice formation ranges are unmodifiable without recompilation

CrossingGateSensor.java
- Very simplistic
- Assumes the function of the sensor to be more complex than other sensors

WheelSensor.java
- No way to change the units in which wheel diameter is measured in (conversion may cause inaccuracies that will lead to incorrect speeds)

LoggingSystem.java
- Does not account for the cases if the log file cannot be found during the trip (*Defect*)
- Makes no attempt to make a new file and will keep throwing errors

RouterSim.java

- Not a real version of what will be used to take in sensor data
- Good simulation in the fact that it considers a script file as an input pipeline that a real router will have
- Heavily relies on the fact that the input sensor data is properly formatted and does not attempt to check. (*Defect*)

IoTSimulator.java

- Coding structure is complicated and difficult to work with
- Relies on Thread.sleep() calls, which if they fail the whole system will become desynchronized (*Defect*)
- Coding structure directly follows the UML State Diagram