EXERCISE 3 - CRITICAL SYSTEMS

https://github.com/AleksiJoo/exercise-3-critical-systems

Author
Aleksi Jarva
Kim Kuparinen
Ville Heikkinen
Jannatul Mawa

Contents

1 Specification analysis

The original specification was deemed untenable, as a 60 day mean time to failure (MTTF) would require a frankly ridiculous number of ruggedized ESPs, at least with the provided data.

By our calculations, placing all ESPs in the position with the lowest average failure rate and ignoring all possible mass failure (rock hitting back of robot destroying all ESPs mounted there), the smallest amount of ESPs required to fulfill the 60 day requirement would be 12. Consider the following equations for calculating the MTTF:

$$R_b(t) = (1 - 0.05)^t$$
$$R_s(t, n) = 1 - (1 - R_b(t))^n$$
$$MTTF(n) = \int_0^\infty R_s(t, n)dt$$

Here, n is the number of ESPs mounted at the back position, R_b is the probability that one ESP is operational on day t, R_s is the probability that the whole system is operational on day t. Iterating over $n = [0, 12]$ we get the following:

$MTTF(1)$	19.496
$MTTF(2)$	29.244
$MTTF(3)$	35.742
$MTTF(4)$	40.616
$MTTF(5)$	44.515
$MTTF(6)$	47.765
$MTTF(7)$	50.550
$MTTF(8)$	52.987
$MTTF(9)$	55.153
$MTTF(10)$	57.102
$MTTF(11)$	58.875
$MTTF(12)$	60.499

Placing 12 ESPs in the same location seems ludicrous to us, especially once we consider that a falling rock that hits the spot where all ESPs are mounted is likely to make the whole system inoperable, most likely meaning that the actual MTTF is lower than the calculated value.

We also have some doubts about the validity of the provided data, the MTTF of 20 the customer is reporting with one ESP seems to somewhat closely match the calculated MTTF(1), although rounded in the wrong direction. However, we're unsure how the values were calculated and the extent of the value, for example does the failure rate take into account physical damage due to falling rocks or is it only component failure due to hot gases in the mine?

2 Suggested architecture

Due to the concerns raised above, we have come up with a suggestion that does not meet the requirements, but that can be a good starting point.

2.1 Hardware

A single drilling robot will be mounted with four ruggerized ESP microcontrollers. One for each position (rear, top, front left, front right). Refer figure 1 for illustration. ESP's will be configured as parallel units. Four microcontroller setup is balance between availabilty, price and reliability of the measurements.

There are several reasons to justify this hardware configuration. Physical separation of the units prevent the failure of all units at the same time due to falling rocks or other physical hazards. When the units are distributed across the drilling robot there is a higher possibility that local temperature anomalies affect only one sensor, meaning that the anomaly can potentially be recorded as an anomaly rather than just another blind data point. Additionally, the new configuration allows easier fault detection compared to the old system, where if a sensor gets damaged such that it starts sending garbage data, it can be compared to the other sensors and discarded as false, whereas the old system didn't have any kind of support for this.

The approximate MTTF with this configuration and given reliability data is

$$
\begin{aligned}
R_b(t) &= (1-0.05)^t \\
R_t(t) &= (1-0.07)^t \\
R_{fr}(t) &= (1-0.09)^t \\
R_{fl}(t) &= (1-0.09)^t \\
R_s(t) &= 1-(\\
&\quad (R_b(t)(1-R_t(t))(1-R_{fr}(t))(1-R_{fl}(t)) \\
&\quad + R_t(t)(1-R_b(t))(1-R_{fr}(t))(1-R_{fl}(t)) \\
&\quad + R_{fr}(t)(1-R_b(t))(1-R_t(t))(1-R_{fl}(t)) \\
&\quad + R_{fl}(t)(1-R_b(t))(1-R_t(t))(1-R_{fr}(t))) \\
&\quad + (1-R_b(t))(1-R_t(t))(1-R_{fr}(t))(1-R_{fl}(t))) \\
MTTF &= \int_0^\infty R_s(t)dt \approx 14.368
\end{aligned}
$$

Since we're operating under a 2-of-4 configuration (more info in the Software section), if two of the sensors are still operable, the system as a whole is still operable. The calculations reflect this by first calculating the probability of one sensor still being alive and the probability of all sensors being dead, both being prerequisites for the whole system to be dead. This is then inverted to represent the probability of the system being alive at time t.

It is worth noting that while the calculation may seem to imply we're lowering the whole system's operational time, the system is now more resilient to one-time-events that would've previously completely disabled the robot. Also by using multiple sensors and majority voting the measurements are more reliable to represent the physical atmosphere around drilling robot. Depending on how common these one-time-events are and how they affect the lifetime data from the customer, it is still possible that the robot has an overall longer lifetime. Unfortunately, the data given is not detailed enough for this kind of analysis.

We would still recommend trying to improve the lifetime of the sensors as (likely) the cheapest way to improve robot lifetimes, for example building cages around the sensors to protect them from falling debris and/or improving air circulation around the humidity sensor to avoid it getting waterlogged.

2.2 Software

Software architecture used is majority voting architecture. Majority voting will take place in host platform (outside of mining cave). The ESPs mounted in drilling robots will be responsible for measuring temperature and humidity and sending data to the host via TTE mining communication system. Each ESP can be thought of as being connected to the host in parallel. Software in the system can be considered as single point of failure since the same software is duplicated to all ESP measurement units. As per failure rates for customer it seems obvious that the hardware is more likely to fail than software so we didn't consider this SPoF as big issue.

2.2.1 Majority voting

Majority voting system in host application will receive two measurement signals (temperature and humidity) from each of ESPs installed on drilling robot. Majority voter will analyze measurements and reject signals that differs too much from the majority of signals. By detecting outliers in measurements the majority voting system will be able to tell which sensors are not functioning properly and report this to maintenance. It is up to operator to recall drilling robot back for maintenance when considered necessary.

Results of majority voter are most reliable when at least 3 out of 4 measurements are in accepted tolerance. If 2 out of 4 sensors are malfunctioning the majority voter will still give reliable results, assuming 2 malfunctioning sensors are not displaying similar incorrect data. We deemed this situation very unlikely.

In case of 1 out of 4 situation where only 1 sensor is working there is no way to tell if the measurement is reliable or correct. Operator is informed that no measurement data is available and robot should be recalled for maintenance.

A three voter system would be more typical, but we agreed that two sensors breaking seemed to be a common enough occurrence (approx. once per day in the whole fleet) that we didn't want to risk the incorrect value to be in the majority, in case both sensors broke in the same way and reported the same incorrect value. The case is, as previously stated, probably very unlikely, but at least in the four sensor architecture we chose this situation is still detectable, though requires immediate repair.

2.2.2 Proof of concept

As part of the evaluation proof of concept software was developed based on previous software from the customer. It consists of client and host software. Client software is running on ruggerized ESPs. Temperature and humidity is measured and send to MQTT broker hosted on test server (test.mosquitto.org). Host software subscribes to correct data topics to retrieve data from broker. The proof of concept host software doesn't implement the actual majority voting system since it is up to customer preferences and domain knowledge. It however presents the measurement data via terminal interface (figure 2).

Proof of concept software was tested with two physical ESP devices. Host software simulates two more ESPs by adding random noise to the actual measurements by two physical ESPs.

2.3 System safety

As per customer requirements the system is not considered as life critical but mission critical. Measurement software running in ruggerized ESP can be developed using low SIL (Safety Integrity Level) methods to reduce costs. Measurement software is also quite simple and manageable. Host software and majority voter are also mission critical but more engineering rigor can be applied since failure of host software will ground the whole fleet so more development costs are justified.

2.3.1 Risk matrix

	Negligible	Minor	Moderate	Significant	Severe
Very Likely	Low med	Medium	Med high	High	High
Likely	Low	Low med	Medium	Medium high	High
Possible	Low	Low med	Medium	Medium high	Medium high
Unlikely	Low	Low med	Low med	Medium	Medium high
Very Unlikely	Low	Low	Low med	Medium	Medium

2.3.2 Risk analysis

Hazard	**Possibility**	**Severity**	**Risk**
Rock falls to drilling robot	Very Likely	Moderate	Medium high
Ruggerized ESP fails	Likely	Significant	Medium high
Humidity sensor gets stuck	Possible	Significant	Medium high
Connection is lost to drilling robots	Possible	Severe	Medium high
Local temperature anomalies affect measurements	Very Likely	Moderate	Medium high
ESPs take physical damage from cave walls	Possible	Significant	Medium high
Cave gets flooded with water	Very Unlikely	Severe	Medium

From most obvious hazards identified for the system the hardware and software architecture defined in this document tries to mitigate the risks. Redundancy and physical separation of sensor units tries to mitigate falling rocks, physical damage, ESP failures, humidity sensor getting stuck and local anomalies. The quality of service of communication system is upon operator.

2.4 Other architectures considered

Technically the highest theoretical MTTF would be achieved by stacking a large number of sensors on the back of the robot, but this was seen as infeasible both due to cost and risk of one-time-events destroying the entire sensor cluster.

3 Cost analysis

Since we don't know how much a ruggedized ESP costs, nor how much the company can expect downtime to cost, we can't provide any kind of direct cost analysis, but we can calculate some approximations for how many ESPs are likely to break per day.

We're using the data provided by the customer despite our qualms with it, since we don't have anything more accurate at the moment.

3.1 ESP stockpiles

Since we have approximately 100 robots in the fleet, and each one has one ESP in each position, we can calculate how many ESPs are likely to break per day for each position with the following:

$$\begin{aligned} Q_b &= 0.05 \\ Q_t &= 0.07 \\ Q_{fr} &= 0.09 \\ Q_{fl} &= 0.09 \\ N_b &= 100Q_b \approx 5 \\ N_t &= 100Q_t \approx 7 \\ N_{fr} &= 100Q_{fr} \approx 9 \\ N_{fl} &= 100Q_{fl} \approx 9 \end{aligned}$$

The sum of these is $N_b + N_t + N_{fr} + N_{fl} = 30$, i.e. we expect each day 30 out of the 400 ESPs in active use will break per day.

We assume each position will be replaced eventually, so we don't take into account cumulative usage. In practice a broken ESP that is part of a still working robot can't break again, meaning day after day the number of ESPs breaking will likely settle down to some value slightly smaller than the one calculated here. The customer is responsible for keeping stock of ESPs.

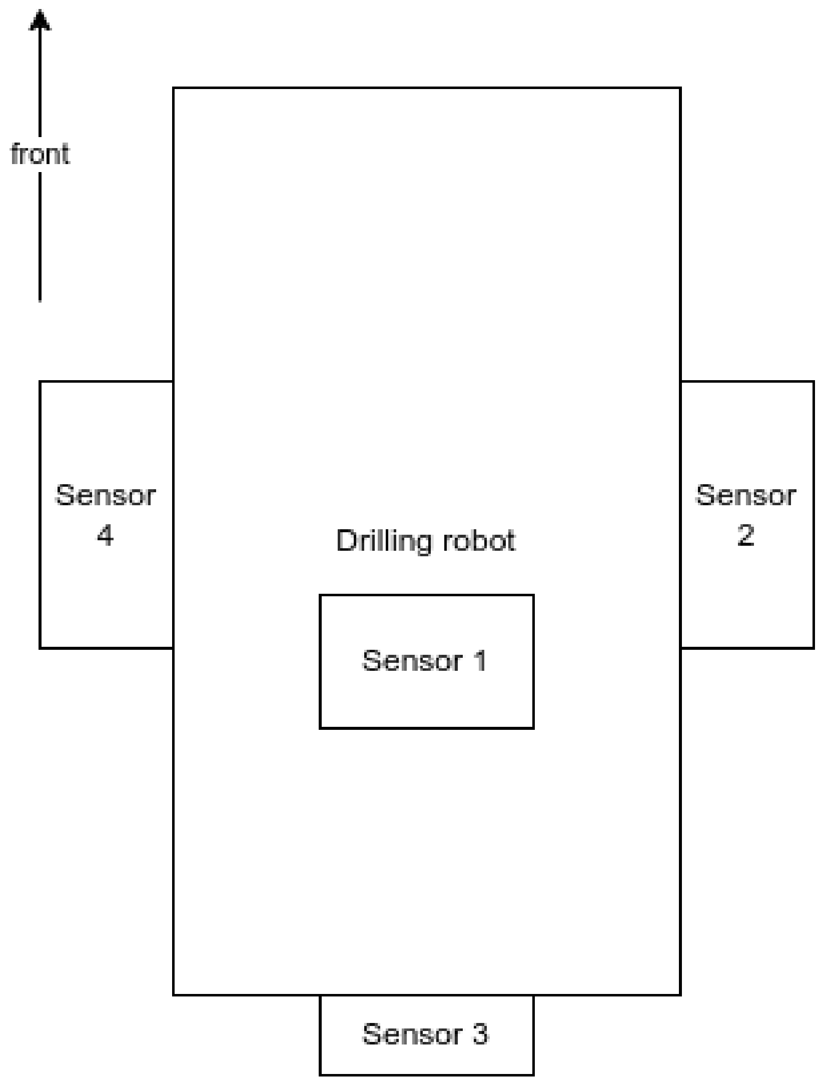
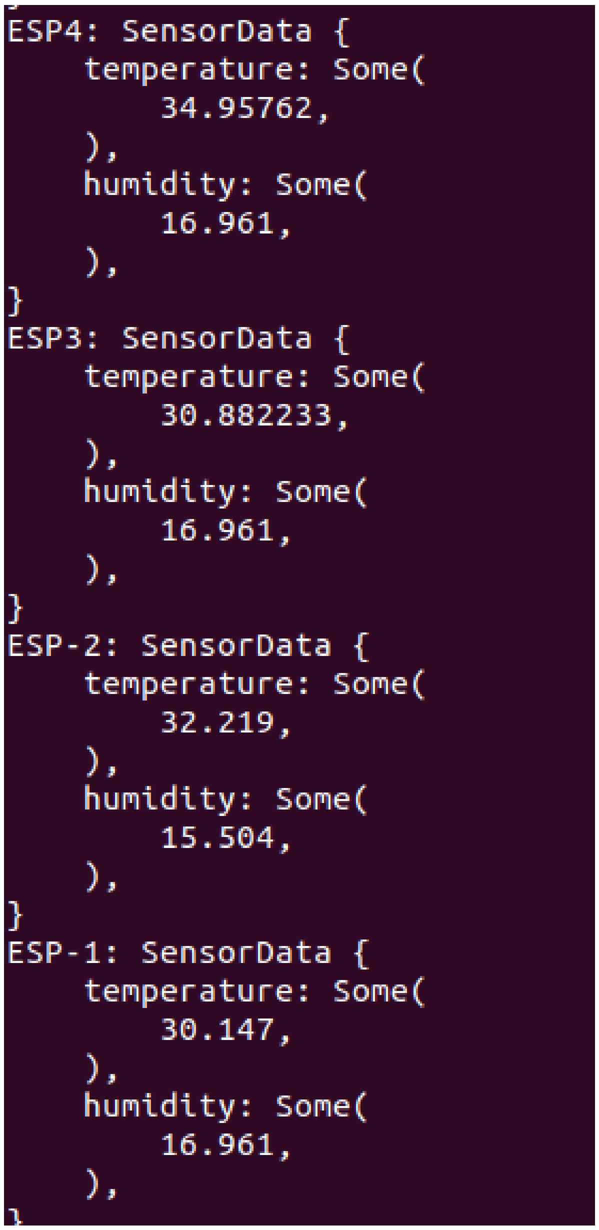


Figure 1: Physical hardware configuration



```
ESP4: SensorData {
    temperature: Some(
        34.95762,
    ),
    humidity: Some(
        16.961,
    ),
}
ESP3: SensorData {
    temperature: Some(
        30.882233,
    ),
    humidity: Some(
        16.961,
    ),
}
ESP-2: SensorData {
    temperature: Some(
        32.219,
    ),
    humidity: Some(
        15.504,
    ),
}
ESP-1: SensorData {
    temperature: Some(
        30.147,
    ),
    humidity: Some(
        16.961,
    ),
```

Figure 2: Host software in operation