6.033 Lecture 14
Fault Tolerance I

Americo De Filippo

June 22, 2023

References

[1] Saltzer, Jerome H. and M. Frans Kaashoek. Principles of Computer System Design: An Introduction (2009): **Section: 8.1 - 8.3**

Today we are gonna begin a new topic in the course is called fault tolerance. The idea is how to build relaiable systems.

1 Faults

A faults are the some thing which make the system do not work the way he's supposte to. There could be diverse types of fault: Bugs, Hardware, Design, Implmentation, Operation (human error).

1.1 Latent Fault and Active Fault

An example could be instead of checking A ¿ B you mistaking check B ¿ A, which will make the program run till it will read that lines. This will become an Active fault when is envoce that lines that will lead to and error that will might cause a failure.

2 How to make a system do not fail

If we want to have an unfailure system it will require too much effor and for this reason cannot be created. So the idea is to start with Unreliable components (modules) and we are gonna build reliable system out of them. The system has to be tolerant of the unreliability of the modules which make him.

2.1 How to handle failure

Let's say that we have a recurive call to a component M_4 we can hide the fault instead of make him propagate in the other modules. So M_3 has the job to handle the fault and do not make them propagate to the original caller let's say M_1.

Examples of fault tolerance

- Bad locking
- Routing (nodes that could blow up)
- Packet loss
- Congestion collapse (solution: run the system slower)
- DNS (names are replicated)

3 Systemaic Approach

1. Modalize
2. Detect errors
3. Mask errors

The goal is to make sure that the systme conform to specification. The general trick is to use some form of redundacy in order to mask errors.

Various kinds of failures

- Fail-stop
- Fail-fast (like bad sector in disk reading)
- Fail-soft (like just allowing read-only actions)
- Fail-safe (like working just a lower performance)

4 Models

A disk manufacturer could report that the error rate of undetected error is 10^{30} that means that the higher layer you have to have tequinque the integrity of you data. When you build a system you want to specify the number of # tolerated failures. The second metric is something called: **Mean Time To Failure (MTTF)**, what this said is taking a system that is running time, and at some time occures a failure that will be repaired after the system went down. So each time interval of time in which the system is working will be called TTF (Time To Failure), meanwhile the time which is down and is going to be repaired is called (TTR).

4.1 Availability

This is important to be defined in order to specify the time in which the system is up and running. It's usually defined: $\frac{\sum TTF_i}{\sum TTF_i + \sum TTR_i}$

4.2 Failure Rate

The failure rate is defined as the probability that you have a failure of a system or a component in time t + dt, saying that is working proprely at t.
h(t) = P(failure in t, t + dt — OK t).

4.3 Reliability

The reliability is the probability that the system is working at time t + Δt given that is working at time t. The function is found to be definded as $R(t) = e^{-(t/MTTF)}$. Which is an exponential decay function.