TTK4145 – Real-time Programming

Sondre Bø Kongsgård
sondrebk@stud.ntnu.no

Innhold

1 Fault model and software fault masking

This chapter ...

Fault model The one used in this chapter involves three entities: *processes, messages and storage.*

Unexpected faults Faults that are not tolerated by the design. Two categorizations:

- *Dense faults*: The algorithms will be n-fault tolerant. If there are more than n faults within a repair period, the service may be interrupted (in this case the system should be designed to stop).
- *Byzantine faults*: The fault model postulates certain behavior – for example it may postulate that programs are failfast. Faults in which the system does not conform to the model behavior are called Byzantine.

Underlying progression

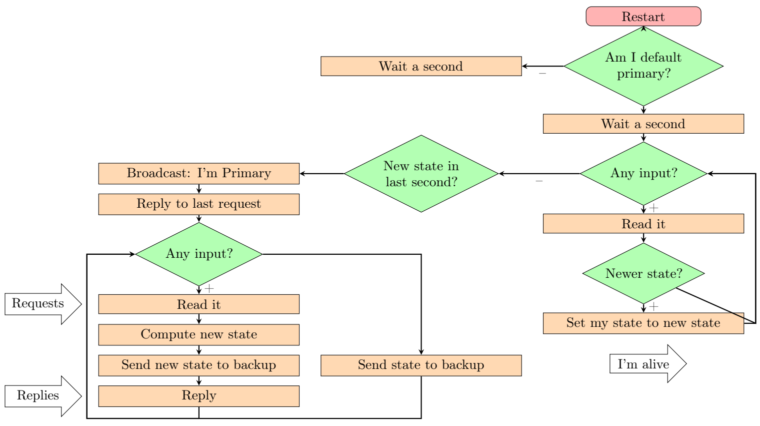
- *Failfast*: They either execute the next step, or they fail and reset to the null state.
- *Available*: Failfast + repairability
- *Reliable*: Continuous operation

Checkpoint-Restart Write state to storage after each acceptance test, but before each event. This approach can be somewhat slow (hours/days).

Process pairs OS generates a backup process for each new primary process. The primary sends *I'm Alive* messages to the backup on a regular basis.

The backup can take over in three different ways:

- *Checkpoint-restart.* The primary records its state on a duplexed storage module. At takeover, the backup starts by reading these duplexed storage pages. (quick repair)
- *Checkpoint message.* The primary sends its state changes as messages to the backup. At takeover, the backup gets its current state from the most recent message. (basic pairs must checkpoint)
- *Persistent.* The backup restarts in the null state and lets the transaction mechanism clean up (undo) any recent uncommitted state changes. (simple)


Restart
Am I default primary?
Wait a second
−
Wait a second
Any input?
−
New state in last second?
Broadcast: I'm Primary
Reply to last request
Any input?
+
Requests
Read it
Compute new state
Send new state to backup
Replies
Reply
Send state to backup
+
Read it
Newer state?
+
Set my state to new state
I'm alive