Module 8
Fault Tolerance
Dependability

• Reliability
  ➔ A measure of success with which a system conforms to some authoritative specification of its behavior.
  ➔ Probability that the system has not experienced any failures within a given time period.
  ➔ Typically used to describe systems that cannot be repaired or where the continuous operation of the system is critical.

• Availability
  ➔ The fraction of the time that a system meets its specification.
  ➔ The probability that the system is operational at a given time $t$.

• Safety
  ➔ When the system temporarily fails to conform to its specification, nothing catastrophic occurs.

• Maintainability
  ➔ Measure of how easy it is to repair a system.
Fundamental Definitions

• Failure
  ➔ The deviation of a system from the behavior that is described in its specification.

• Error
  ➔ The part of the state which is incorrect.

• Fault
  ➔ Cause of an error.
Faults to Failures

Fault → causes → Error → results in → Failure
Types of Faults

• Hard faults
  ➔ Permanent
  ➔ Resulting failures are called hard failures

• Soft faults
  ➔ Transient or intermittent
  ➔ Resulting failures are called soft failures
Fault Classification

Permanent fault
Incorrect design
Unstable or marginal components
Unstable environment
Operator mistake

Permanent error
Intermittent error
Transient error

System Failure
## Failure Models

<table>
<thead>
<tr>
<th>Type of failure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Crash failure</td>
<td>A server halts, but is working correctly until it halts</td>
</tr>
<tr>
<td>Omission failure</td>
<td></td>
</tr>
<tr>
<td><em>Receive omission</em></td>
<td>A server fails to respond to incoming requests</td>
</tr>
<tr>
<td><em>Send omission</em></td>
<td>A server fails to receive incoming messages</td>
</tr>
<tr>
<td><em>State transition failure</em></td>
<td>A server fails to send messages</td>
</tr>
<tr>
<td>Timing failure</td>
<td>A server's response lies outside the specified time interval</td>
</tr>
<tr>
<td>Response failure</td>
<td>The server's response is incorrect</td>
</tr>
<tr>
<td><em>Value failure</em></td>
<td>The value of the response is wrong</td>
</tr>
<tr>
<td><em>State transition failure</em></td>
<td>The server deviates from the correct flow of control</td>
</tr>
<tr>
<td>Arbitrary failure</td>
<td>A server may produce arbitrary responses at arbitrary times</td>
</tr>
</tbody>
</table>
How to Improve Dependability

• Mask failures by redundancy
  ➔ Information redundancy
    ✦ E.g., add extra bits to detect and recovered data transmission errors
  ➔ Time redundancy
    ✦ Transactions; e.g., when a transaction aborts re-execute it without adverse effects.
  ➔ Physical redundancy
    ✦ Hardware redundancy
      ✓ Take a distributed system with 4 file servers, each with a 0.95 chance of being up at any instant
      ✓ The probability of all 4 being down simultaneously is $0.05^4 = 0.000006$
      ✓ So the probability of at least one being available (i.e., the reliability of the full system) is 0.999994, far better than 0.95
      ✓ If there are 2 servers, then the reliability of the system is $(1-0.05^2) = 0.9975$
    ✦ Software redundancy
      ✓ Process redundancy with similar considerations

• A design that does not require simultaneous functioning of a substantial load of critical components
Hardware Redundancy

- Two computers are employed for a single application, one acting as a standby
  ➔ Very costly, but often very effective solution
- Redundancy can be planned at a finer grain
  ➔ Individual servers can be replicated
  ➔ Redundant hardware can be used for non-critical activities when no faults are present
  ➔ Redundant routes in network
Process Redundancy

- Process groups
  - All members of a group receive a message sent to the group.
  - If one process fails, others can take over.
  - Can be dynamic; processes can have multiple memberships.
  - Flat versus hierarchical groups:
Transactional Dependability

- Problem: How to maintain
  - atomicity
  - durability
  properties of transactions
- Focus is on data: Failures that affect ACID properties of data
  - Transaction failures
    - Transaction aborts (unilaterally or due to deadlock)
    - Avg. 3% of transactions abort abnormally
  - System (site) failures
    - Failure of processor, main memory, power supply, ...
    - Main memory contents are lost, but secondary storage contents are safe
    - Partial vs. total failure
  - Media failures
    - Failure of secondary storage devices such that the stored data is lost
    - Head crash/controller failure
  - Communication failures
    - Lost/undeliverable messages
    - Network partitioning
Distributed Reliability Protocols

- Commit protocols
  - How to execute commit command for distributed transactions.
  - Issue: how to ensure atomicity and durability?
    - A transaction that executes at different sites has to complete (i.e., abort or commit) the same way everywhere

- Termination protocols
  - If a failure occurs, how can the remaining operational sites deal with it.
  - **Non-blocking**: the occurrence of failures should not force the sites to wait until the failure is repaired to terminate the transaction.

- Recovery protocols
  - When a failure occurs, how do the sites where the failure occurred deal with it.
  - **Independent**: a failed site can determine the outcome of a transaction without having to obtain remote information.
**Two-Phase Commit (2PC)**

*Phase 1*: The coordinator gets the participants ready to commit their writes

*Phase 2*: Everybody commits

- **Coordinator**: The process at the site where the transaction originates and which controls the execution
- **Participant**: The process at the other sites that participate in executing the transaction

**Global Commit Rule:**

- The coordinator aborts a transaction if and only if at least one participant votes to abort it.
- The coordinator commits a transaction if and only if all of the participants vote to commit it.
Centralized 2PC

C  P  P  P  C

C  P  P  P  C

ready? yes/no commit/abort? committed/aborted

Phase 1  Phase 2
2PC Protocol Actions

Coordinator

- INITIAL
  - write begin_commit in log
  - PREPARE
  - VOTE-ABORT
  - VOTE-COMMIT
  - Any No?
    - Yes
      - write abort in log
    - No
      - write commit in log
  - COMMIT
    - write end_of_transaction in log

Participant

- INITIAL
  - write abort in log
  - Ready to Commit?
    - No
    - Yes
      - write ready in log
  - READY
    - GLOBAL-ABORT
    - Type of msg
      - Abort
        - ABORT
        - write abort in log
      - Commit
        - COMMIT
        - write commit in log
      - ACK
    - ACK
State Transitions in 2PC

Coordinator

Participants
Problem With 2PC

• Blocking
  ➔ Ready implies that the participant waits for the coordinator
  ➔ If coordinator fails, site is blocked until recovery
  ➔ Blocking reduces availability

• Independent recovery is not possible

• However, it is known that:
  ➔ Independent recovery protocols exist only for single site failures; no independent recovery protocol exists which is resilient to multiple-site failures.

• So we search for these protocols – 3PC
Quorum Protocols for Network Partitioning

- The network partitioning problem is handled by the commit protocol.
- Every site is assigned a vote $V_i$.
- Total number of votes in the system $V$
- Abort quorum $V_a$, commit quorum $V_c$
  - $V_a + V_c > V$ where $0 \leq V_a, V_c \leq V$
  - Before a transaction commits, it must obtain a commit quorum $V_c$
  - Before a transaction aborts, it must obtain an abort quorum $V_a$
Network Partitioning & Replication

- A group of replica managers are partitioned into two or more subgroups such that members of one subgroup can communicate with each other but members of different subgroups cannot communicate with one another.

Client + front end
withdraw(B, 4)

Client + front end

Network partition

deposit(B, 3)

Replica managers
Replica Management with Network Partitioning

• Optimistic approach
  ➡️ Available copies with validation
    ✦ Let each partition perform updates freely
    ✦ Validate when partition is repaired and those that violate one-copy serializability are aborted.

• Pessimistic
  ➡️ Quorum-based
    ✦ Reduce availability (even when there is no partitioning)
    ✦ Updates can only occur in the partition that has a majority of the replica managers