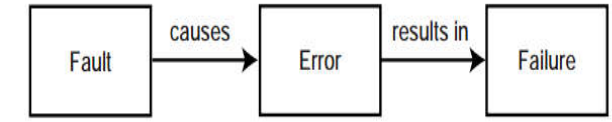
Fundamental Definitions

- Reliability
  - Probability that the system has not experienced any failures within a given time period.
- Availability
  - The probability that the system is operational at a given time *t*.

Fundamental Definitions

- Failure
  - Failure in a system can be attributed to deficiencies either in the components that make it up, or in the design
- Erroneous state
  - The transitions from this state would eventually cause a system failure
- Error
  - The part of the state which is incorrect.
- Fault
  - An error in the internal states of the components of a system or in the design of a system.

Faults to Failures

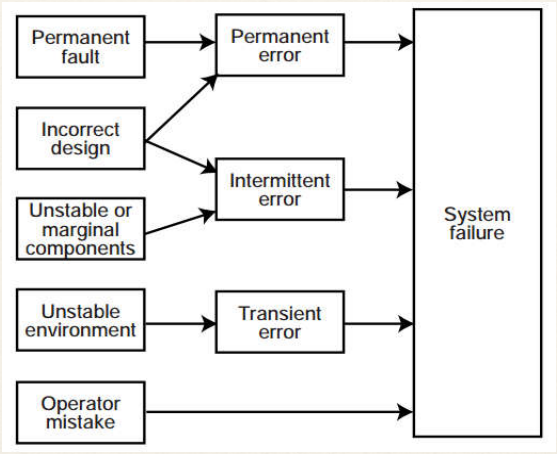


Chain of Events Leading to System Failure

Types of Faults

- Hard faults
  - Commonly called as Permanent faults
  - Permanent faults cause permanent errors that result in permanent failures
  - Recovery from such fault requires repairing the system
- Soft faults
  - Transient or intermittent faults jointly called soft faults
  - An intermittent fault refers to fault that demonstrates itself occasionally due to unstable hardware or varying hardware or software states.
  - Transient fault describes fault that results from temporary environmental conditions.

Fault Classification



Sources of System Failure

Fault-Tolerance Measures

Reliability R(t) refer to the probability that the system under consideration does not experience any failures in a given interval

The mean number of failures in time [0, t] can be computed as

$$E[k] = \sum_{k=0}^{\infty} k \frac{e^{-m(t)}[m(t)]^k}{k!} = m(t)$$

and the variance can be be computed as

$$Var[k] = E[k^2] - (E[k])^2 = m(t)$$

Thus, reliability of a single component is

$$R(t) = e^{-m(t)}$$

and of a system consisting of n non-redundant components as

$$R_{sys}(t) = \prod_{i=1}^{n} R_i(t)$$

Fault-Tolerance Measures

Availability A(t) refers to the probability that system is operational according to its specification at a given point in time t

$A(t)$ = Pr{system is operational at time t}

Assume

- Poisson failures with rate λ
- Repair time is exponentially distributed with mean $1/\mu$

Then, steady-state availability

$$A = \lim_{t \to \infty} A(t) = \frac{\mu}{\lambda + \mu}$$

Fault-Tolerance Measures

MTBF

Mean time between failures

$$\text{MTBF} = \int_0^\infty R(t)\,dt$$

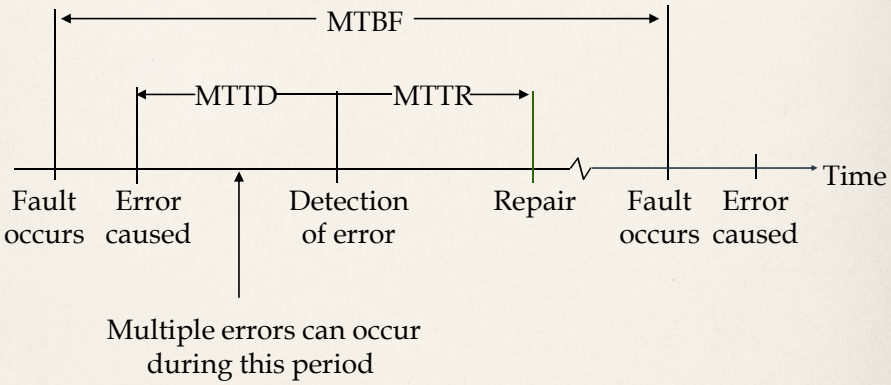
MTTR

Mean time to repair

Availability

$$\frac{\text{MTBF}}{\text{MTBF} + \text{MTTR}}$$

Failures

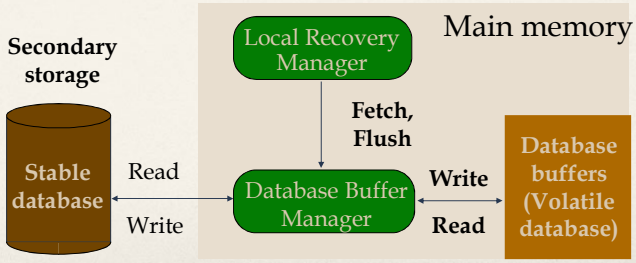


Failures in Distributed DBMS

- 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

Local Recovery Management – Architecture

- Volatile storage
  - → Consists of the main memory of the computer system (RAM).
- Stable storage
  - → Resilient to failures and loses its contents only in the presence of media failures (e.g., head crashes on disks).
  - → Implemented via a combination of hardware (non-volatile storage) and software (stable-write, stable-read, clean-up) components.



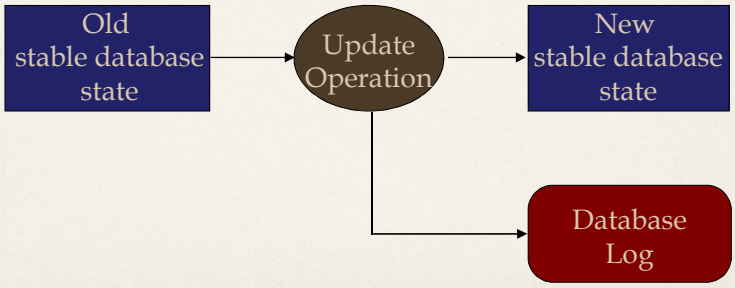
Update Strategies

- In-place update
  - → Each update causes a change in one or more data values on pages in the database buffers
- Out-of-place update
  - → Each update causes the new value(s) of data item(s) to be stored separate from the old value(s)

In-Place Update Recovery Information

Database Log

Every action of a transaction must not only perform the action, but must also write a *log* record to an append-only file.



Logging

The log contains information used by the recovery process to restore the consistency of a system. This information may include

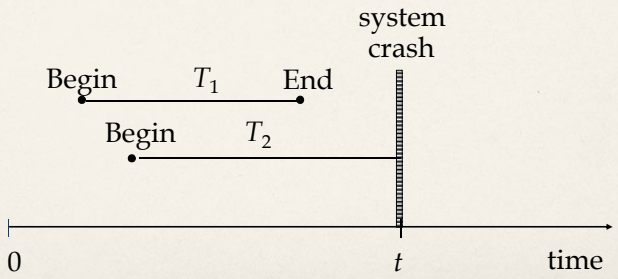
- transaction identifier
- type of operation (action)
- items accessed by the transaction to perform the action
- old value (state) of item (before image)
- new value (state) of item (after image)

…

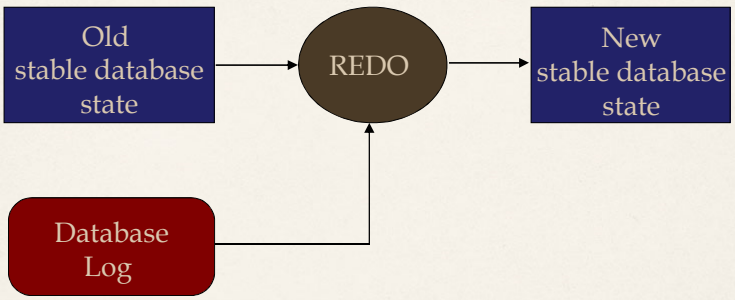
Why Logging?

Upon recovery:

- all of T_1's effects should be reflected in the database (REDO if necessary due to a failure)
- none of T_2's effects should be reflected in the database (UNDO if necessary)

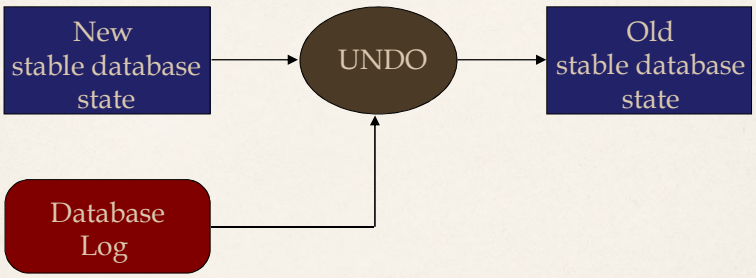


REDO Protocol



- REDO'ing an action means performing it again.
- The REDO operation uses the log information and performs the action that might have been done before, or not done due to failures.
- The REDO operation generates the new image.

UNDO Protocol



- UNDO'ing an action means to restore the object to its before image.
- The UNDO operation uses the log information and restores the old value of the object.