

Concept of Fault, Failure & Recovery

- Definition:

- Fault tolerance is the ability of a system to perform its function correctly even in the presence of internal faults.

- The concept of fault, failure & Recovery is described by using following points.

- Fault Classification
- Failure model in distributed system
- Recovery in Distributed system.

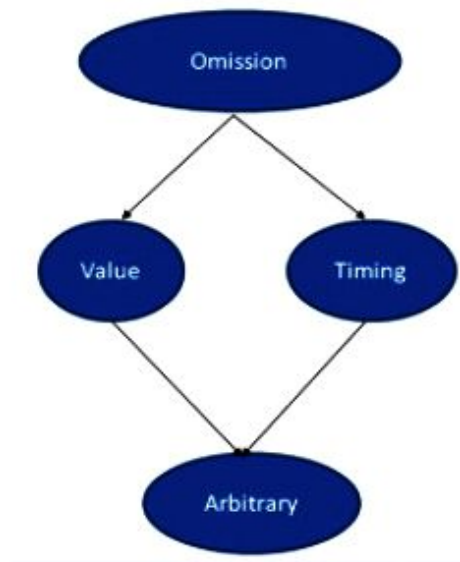
Fault Classification

- Fault can be classified as transient or permanent.
- A transient fault will eventually disappear without any apparent intervention.
- A permanent one will remain unless it is removed by some external agency.
- Classification of faults
 - Crash Fault
 - Omission fault
 - Timing Fault
 - Byzantine fault

Failure Model in Distributed System

- Definition

- It define a precise specification of the faults that can be exhibited by processes and communication channels.
- **Omission Failure**
- **Timing Failure**
- **Arbitrary Failure**



Recovery In Distributed System

- Definition

- A transaction processing system will go through a recovery of the database to cope when the system fails, it involves the backup, journals, checkpoint and recovery manager.
- Types of Recovery Techniques:
 - **Journal**: A journal maintain an audit trail of transaction and database changes.
 - **Checkpoint**: It is to provide a snapshot of the data within the database.
 - **Recovery Manager**: A recovery manager is a program which restores the database to a correct condition which can restart the transaction processing.

Byzantine Faults

- Definition:
 - A Byzantine fault is an arbitrary fault that occur during execution of an algorithm by a distributed system.
- It encompasses both **omission failures** and **commission failure**.
- When a byzantine fault has occurred the system may respond in any **unpredictable way**, unless it is designed to have byzantine fault tolerance.

Adversaries

- Definition:
 - An **adversary is a demon** that has a great deal of control over the environment in which your protocol executes and will try to make your protocol fail.
- The adversary has the power to examine the global state and to schedule the execution of the protocol.
- Adversary is the **ability to destroy or modify message** and change the protocol that some of the processors execute.

Byzantine Agreement

- Definition:
 - It refers to distributed agreement algorithm in which a failed processor can send arbitrary message into the network.
 - A failed processor can try to prevent the nonfailed processor from reaching an agreement.

Impossibility Of Consensus

- Consensus is the task of getting all processes in a group to agree on some specific value based on the votes of each processes.
- The impossibility result depend completely on the fact that the distributed system is asynchronous. While processor can fail the system is otherwise reliable.

Randomized Distributed

Agreement

- A randomized algorithm is one which contains an assignment to a variable based on the outcome of tossing a fair coin or a random number generator.
- The randomized distributed agreement can be described by using some point that is as follows.
 - Exponential Time Shared memory Consensus.
 - Polynomial Time Shared memory Consensus.
 - Message Passing Randomized Consensus.

Randomized Distributed Agreement(Cont..)

- Exponential Time Shared memory Consensus:
 - It is simple-minded approach to writing a consensus protocol in a shared memory system.
 - We initialize by choosing a value to prefer, V_p we look at the value preferred by every other processors $\{V_j\}$. If all other processors have the same value $V_i = V_p$, we decide on value V_p . Otherwise we flip a coin and set V_p to the value of the coin Flip.

Polynomial Time Shared memory

Consensus

- This protocol make it that every processor obtains the same value on its flip() on a given round r .
- If every processor obtain the same value for flip() on an arbitrary round r will probability at least P_{same} , the SM_Consensus protocol will require $O(1/P_{\text{same}})$ rounds.
- If P_{same} is independent of N , only an expected constant number of round are required. So it is worthwhile to spend a fair amount of effort to get every processor to flip the same coin.

Message Passing Randomized

Consensus

- It is similar to the shared memory consensus protocol except shared variable is the V array.
- The element $V[i]$ is written to by processor i only and is read by every other processor. So array V makes shared memory look like a broadcast medium.
- A naive translation of the SM_consensus protocol to the MP_consensus protocol is o broadcast all updates.

Memory Management Algorithm

- Definition

- In multiprogramming system, to share the processor with a number of processes that must be kept in memory. These memory management is achieved through memory management algorithms.

- Methods of Memory Management:

- Fixed Partitioning
- Variable Partitioning
- Paging
- Segmentation

Memory Management Algorithm(Cont..)

- Fixed Partitioning

- In this method, memory is divided into partitions whose sizes are fixed. Operating system is placed into the lowest bytes of the memory. Processes are classified on entry of the system according to their memory they requirements.

	Operating System
(nkB)	Small
(3nkB)	Medium
(6nkB)	Large

Memory Management Algorithm(Cont..)

- Variable Partitioning
 - In it, the memory size may vary dynamically.
 - In it, we keep a table(linked list) indicating used/free areas in memory.
- There are three algorithms for searching the list of free blocks for a specific amount of memory.
 - First Fit: Allocate the first free block that is large enough for the new process. This is the fast algorithm.
 - Best Fit
 - Worst Fit

Memory Management Algorithm(Cont..)

- Paging
 - Paging permit a program to allocate noncontiguous blocks of memory. The OS divide the program into pages which are blocks of small and fixed size. Then it divide the physical memory into frames which are block of size equals to page size.
- The OS uses a page table to map program pages to memory frames.

Replicated Data Management

- Definition

- Replication is the process of copying & maintaining database objects in multiple database that make up a distributed database system.
- Replication can improve the performance & protect the availability of application because alternate data access option exist.

- The replicated data management is described by using some point.

- Concept & Issues
- Database Techniques
- Atomic Multicast
- Update Propagation

Replicated Data Management(Cont..)

- Concept & Issues:

- Basic replication environment support application that require read-only access to the table data that originates from a primary site.

- The concept of basic replication environments

- Use of basic replication
 - ⊠ Information distribution, off loading & Information transport.
- Read-only table snapshots
 - ⊠ It is a local copy of table data that originates from one or more remote master tables.
- Snapshots Refreshes
 - ⊠ A table snapshot is a transaction- consistent reflection of its master data as that data existed at a specific point in time.

Replicated Data Management(Cont..)

- Database Techniques:
 - It is a technique that are used for managing replicated data, that is called database techniques.
- It will examine the fundamentals algorithms for maintaining data consistency in a distributed database.
- Types of Database techniques:
 - Database logging & Recovery
 - Two-phase Commit
 - Three-phase Commit
 - Replicated-Data- Management
 - Dynamic Quorum Changes

Atomic Multicast

- Definition:
 - If one processor on the delivery list receives a reliable multicast messages, every processor on the delivery list receives the message.
- Some point to describe the Atomic multicast.
 - Virtual Synchrony
 - Ordered Multicasts
 - Reliable & Casual multicast
 - Group Membership
 - Atomic group multicasts.

Update Propagation

- The use of update propagation is to replicate the database it manages.
- Updates need to be propagated to all sites.
- The update propagation can be described by using some point that is as follows.
 - Epidemic Algorithms
 - Antientropy
 - Update Logs

Update Propagation(Cont..)

- Epidemic Algorithms
 - When a server updates a data items d , it should start the process of distributing the update.
- Antientropy:
 - A distributing updates is to have one site contact another to exchange recent updates, that is called Antientropy.
- Update Logs
 - Every processor p keeps a log L of the updates that it has processed. The log is an listing of event records, where each event corresponding to an update.

Case Study: CORBA

- Definition

- CORBA is a middleware design that allows application programs to communicate with one another irrespective of their programming language, their hardware and software platforms, the networks they communicate over and their implementers.
- Applications are built from CORBA objects, which implement interfaces defined in CORBA's interface definition language.

Case Study: CORBA(Cont..)

- The Common object Request Broker architecture is described as follows:
 - Architecture
 - CORBA RMI
 - CORBA Services

Case Study: CORBA(Cont..)

- Architecture of CORBA:
 - It designed to support the role of an object request broker that enables clients to invoke methods in remote objects, where both clients and server can be implemented in a variety of programming language.

Case Study: CORBA(Cont..)

- In the CORBA architecture contains three additional components:
 - The object adapter
 - The implementation Repository
 - The interface Repository
- CORBA provides for both static & Dynamic invocations.
 - Static invocation are used when the remote interface of the CORBA object is known at compile time, enabling client stub & Server skeleton to be used.
 - If the remote interface is not known at compile time, dynamic invocation must be used.

Case Study: CORBA(Cont..)

- Object Adapter:

- The role of an object adapter is to bridge the gap between CORBA objects with IDL interfaces and the programming language interface of the corresponding servant classes.

- Skeleton:

- Skeleton classes are generated in the language of the server by an IDL compiler. The skeleton unmarshals the arguments in request messages and marshals exception and result in reply messages.

Case Study: CORBA(Cont..)

- Client Stub:

- In the client language, a set of stub procedures is generated from an IDL interfaces by an IDL compiler for the client language.

- Implementation Repository:

- It is responsible for activating registered servers on demand and for locating servers that are currently running.
- The object adapter name is used to refer to servers when registering and activating them.

- Interface Repository

- It is to provide information about registered IDL interfaces to client and servers that requires it.

Case Study: CORBA(Cont..)

- Client Stub:
 - In the client language, a set of stub procedures is generated from an IDL interfaces by an IDL compiler for the client language.
- Implementation Repository:
 - It is responsible for activating registered servers on demand and for locating servers that are currently running.
 - The object adapter name is used to refer to servers when registering and activating them.
- Interface Repository
 - It is to provide information about registered IDL interfaces to client and servers that requires it.

Case Study: CORBA(Cont..)

- CORBA RMI:
 - CORBA RMI requires more of the programmer than programming in a single language RMI system such as Java RMI.
- CORBA Services:
 - CORBA naming Services
 - CORBA Event Services
 - CORBA notification Services
 - CORBA Security Services.