Fault Tolerance in Distributed Systems

Submitted by

Sumit Jain

Distributed Systems(CSE-510)

Topics

1. Introduction
2. Basics Terminologies
3. Phases in the fault Tolerance.
4. Fault Tolerance Techniques
5. Limitations

1.Introduction

- In the early days of computing, Centralized systems were in use.

- Due to the gradual development in the field of networks and micro-electronics, centralized systems lost their ways to Interconnected-multiprocessor systems.

- Interconnected multiprocessor systems

1.Parallel Processing systems-Single system wide memory.

2.Distributed Systems-No shared memory.

1.1 What is a Distributed System

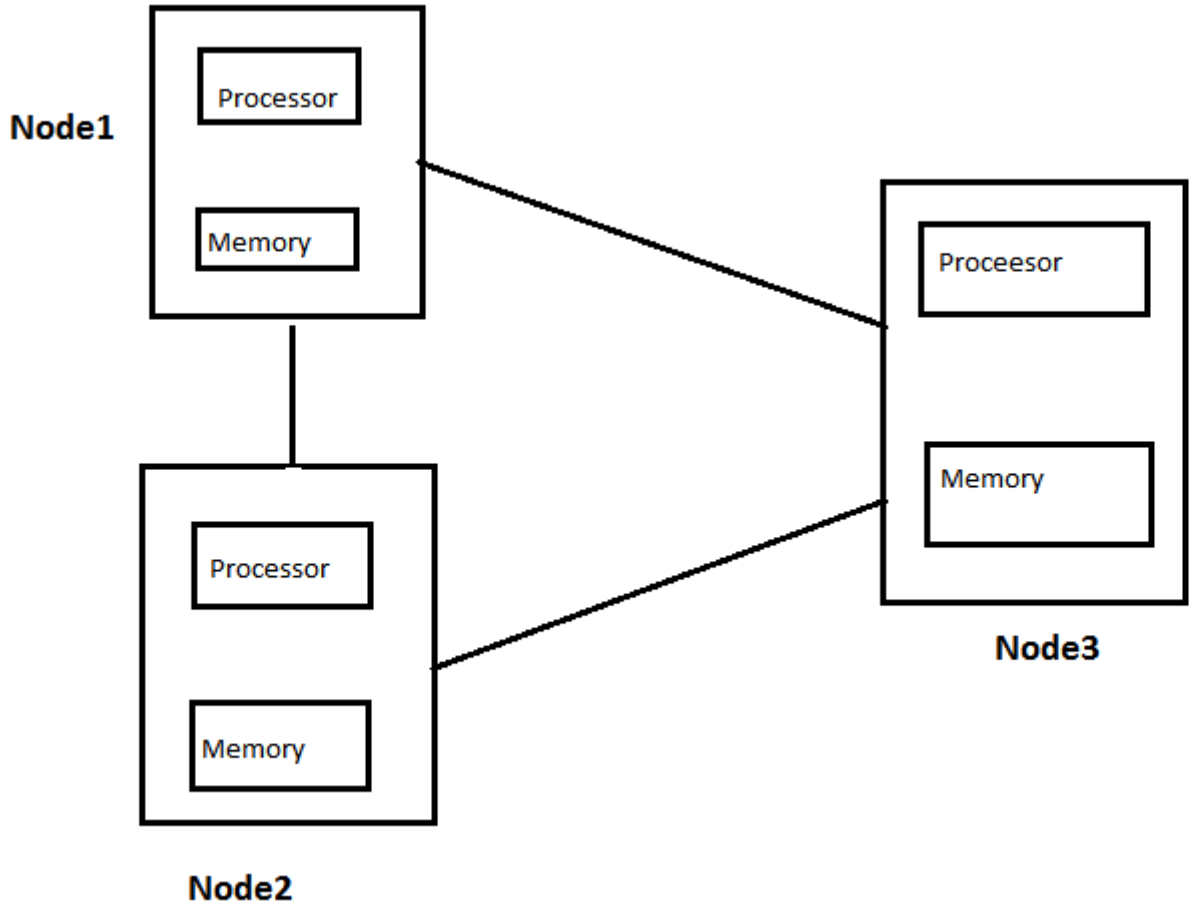
- Collection of independent computers that appear to its users as a single coherent system.
- Every system has its own memory and its own set of resources and they can share some common peripheral devices such as a printer.
- Systems are organized in such a manner so as to hide their existence from the end user.
- Transparency is to be maintained-ISO(8 kind of transperanacy)
- Message passing or RPC technique through communication technology such as TCP/IP.

1.2

- Designing a distributed system is a complex task because of the presence of a large number of components which can be located at distance from each other. some of the major challenges that designers have to face are listed below

- Fault tolerance
- Communication primitives
- Flexibility
- Transparency

1.3Block Diagram of Distributed Systems



2.Faults, Errors and Failures.

- In any distributed system, three kinds of problems can occur.

  1) Faults

  2)Errors(System enters into an unexpected state)

  3)Failures

- All these are inter related.
- It is quite fair to say that fault is the root cause, where a problems starts, error is the result of fault and failure is the final out come.

2.1Types of Faults

Transient Faults	Permanent Faults
Occur for a very short duration.	Permanent
Hard to locate	Easy to be identified
Do not affect the system to a great extent.	Can cause severe damage to the entire system
Network fault, processor fault, Media Fault are some of the examples	Node Level Faults-when an Entire node is unavailable.

2.2 Types of Failures

Crash Failure	**Occurs when a server crashes or any other hardware related problem occurs.**
Omission Failure	**Occurs when a server does not receive incoming requests from client or fails to send messages in response to clients request.**
Timing Failure	**Occurs when a server fails to respond with in a particular amount of time.**
Response failure	**Occurs when a server sends incorrect message in response to the client's message.**
Arbitrary Failure	**Occurs when a server sends any arbitrary message.**

2.3 What is Fault Tolerance?

- Ability of a system to continue functioning in the event of a partial failure.
- Though the system continues to function but overall performance may get affected.
- Distributed systems are made up of a large number of components, developing a system which is hundred percent fault tolerant is practically very challenging.
- Two main reasons for the occurrence of a fault

  1)Node failure -Hardware or software failure.

  2)Malicious Error-Caused by unauthorized Access.

2.4 Why do we need fault tolerance?

- Fault Tolerance is needed in order to provide 3 main feature to distributed systems.

  1)Reliability-Focuses on a continuous service with out any interruptions.

  2)Availability - Concerned with read readiness of the system.

  3)Security-Prevents any unauthorized access.

- examples-Patient Monitoring systems, flight control systems, Banking Services etc.

3.Phases In The Fault Tolerance

- Implementation of a fault tolerance technique depends on the design , configuration and application of a distributed system.

- In general designers have suggested some general principles which have been followed.

  1)Fault Detection

  2)Fault Diagnosis

  3)Evidence Generation

  4)Assessment

  5)Recovery

3.1 Phases In The Fault Tolerance

Fault Detection	• Constantly monitoring the performance and comparing it with expected outcome. •Fault is reported if there is a deviation from expected outcome.
Fault Diagnosis	•Done to understand the nature of the fault and possible root cause.
Evidence Generation	•Report generated based on the outcome of the fault diagnosis.
Assessment	• Understanding the extent of the damage caused by the faulty component. •Done by examining the flow of information that has passed out from the faulty component to the rest of the system. •A virtual Boundary is created.
Recovery	Making the system fault free and restoring it to a consistent state- Forward recovery and Backward recovery.

4.Fault Tolerance Techniques

Replication

- Creating multiple copies or replica of data items and storing them at different sites
- Main idea is to increase the availability so that if a node fails at one site, so data can be accessed from a different site.
- Has its limitation too such as data consistency and degree of replica.

Check Pointing

- Saving the state of a system when they are in a consistent state and storing it on a stable storage.
- Each such instance when a system is in the stable state is called a check point.
- In case of a failure, system is restored to its previous consistent state.
- Saves useful computation.

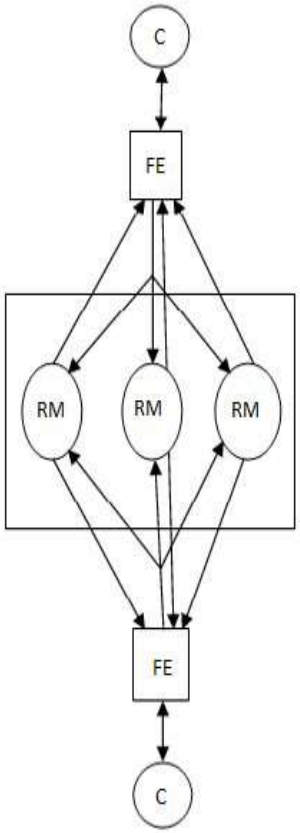
4.1 Replication

- Can be used only for deterministic processes.
- Main idea is to create multiple copies of data and storing them at multiple locations.
- If one site goes down, data is accessed from other so that performance in not affected.
- An effective approach but have limitations also
    1. Data consistency(Data should match)
    2. Degree of Replica(Exact Numbers of Replica).

4.2 Types of Replication

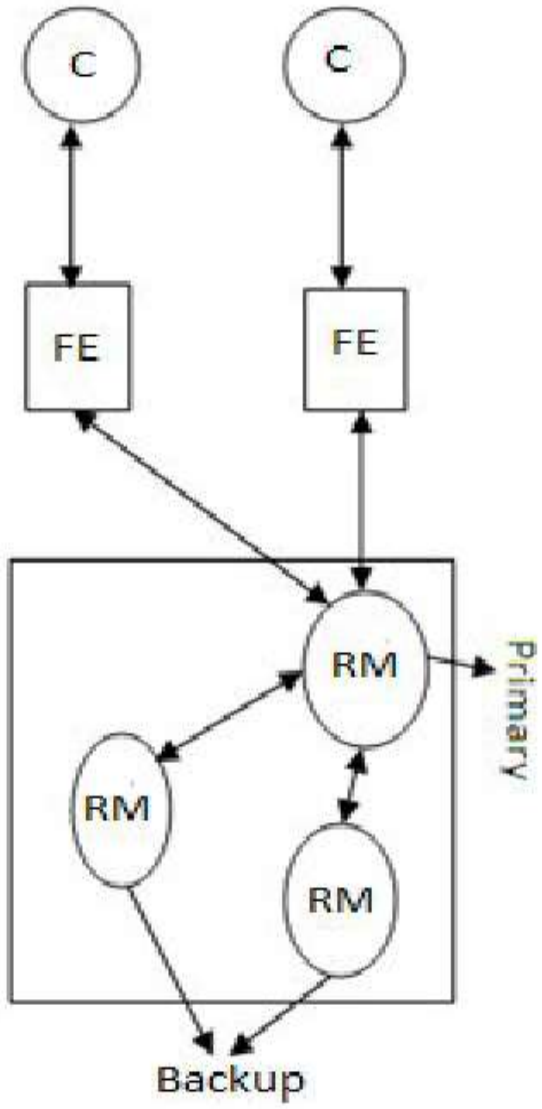
Active Replication

- Can be used only for deterministic processes.
- Client's request is processed by all the servers.
- Requires an atomic broadcast protocol ,to forward the requests to all the servers in the same order.



Passive Replication

- Can be used for non deterministic processes also.
- There is only one server that processes client's request known as primary server .
- Other servers act as back up servers.
- Response time is high as there is only one server which process many client's request.



4.3 Check Pointing

- Every system has some information associated with it which defines its state at a particular moment.
- This information include process state, environment, value of the active registers and variable.
- All this information are collected and stored and each such instance is called a check point.
- In the event of a failure, system is restored to a previously stored check point rather than starting it from the beginning.
- Check pointing is useful but time consuming.

4.4 Types of Check Pointing

User triggered	•Requires User Intervention. •Can be useful only when users have understanding of the computation. •It is not easy to identify when a check point should be created
Uncoordinated Check pointing	• Also known as Independent check pointing. •Processes do not communicate with each other and creates their own check points. •No communication overhead
Coordinated Check points	•Processes communicate with each other. •firstly, temporary check points are created and then made permanent. •Recovery time is high, due to communication.
Message Based Check pointing	Suitable when communication is through message passing only. The state of the processes is stored in the form of message. if one process goes down , other takes its place and acquire its state with the help of messages.

4.5 Limitations

Replication

- Difficult to manage as the no. replica or copies increases.
- Consistency and degree of replica is a major issue.
- Active replication can be used only for deterministic processes.
- Passive replication can be used for non deterministic but its recovery time is high.

Check Pointing

- Lost of computation
- Check point length and check point frequency and storage is a major issue.
- User level check pointing requires users involvement, but not every user is capable of doing it.
- Coordinated check pointing includes communication over head work.

Thank You.