Acta Scientific COMPUTER SCIENCES

Volume 4 Issue 1 Janauary 2022

Literature Review

Fault Tolerance in Distributed System: A Review

Gajendra Sharma* and Santosh Sah

Department of Computer Science and Engineering Kathmandu University, Dhulikhel, Kathmandu, Nepal

***Corresponding Author:** Gajendra Sharma, Department of Computer Science and Engineering Kathmandu University, Dhulikhel, Kathmandu, Nepal.

Received: October 19, 2021

Published: December 17, 2021

Abstract

Distributed systems consist of several hardware and software components connected together which may fail eventually. Therefore, the system should be designed with the proper fault tolerance technique, so that in case of fault the system would be able to recover from the failure without any loss of service. Fault may occur due to various reasons like communication failure, resources or hardware failure, failure due to fault in process, software errors etc. Any of these faults may result the system into faulty environment. The system in faulty environment may not perform the task as expected and will result in faulty output or no output. This paper attempts to introduce fault, fault tolerance and fault tolerance techniques in detail with the help of previous research in the field of fault tolerance in distributed system.

Keywords: Fault; Fault-Tolerance; Distributed System; Fault Tolerance Techniques

Introduction

Distributed Computing Systems consists of variety of hardware and software components. Failure of any of these components can lead to unanticipated, potentially disruptive behavior and service unavailability [1]. In the event of failure in the system or any component of the system, the system must be capable of operating as normal condition. This quality of system to operate normally in case of failure is the fault tolerance of the system. High-availability of the system is guaranteed by the fault tolerance of the system. Incorrect resultor unpredictable service of the system cannot be accepted in real time distributed system. Some examples are the online transaction, process control and computer based communication system. A system's fault tolerant capability guarantees to minimize the loss that may be caused due to the unpredictable system behavior. The demand for fault tolerance in a system is increasing day by day.

Common characteristics of a Distributed System are Resource Sharing, Openness, Scalability, Transparency, and most importantly is Fault Tolerance. In distributed system the individual workstations communicate each other by passing messages. There are always chances of fault to occurs which may be due communication failure, hardware failure, shortage of memory, software bugs etc.

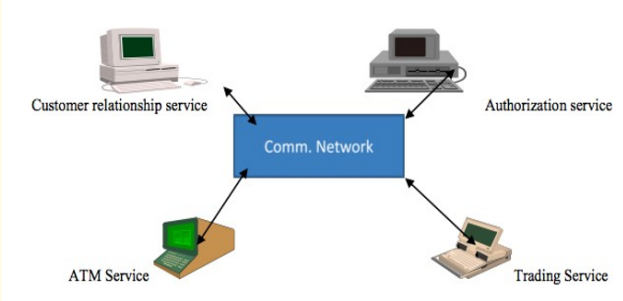

Figure 1: Distributed system [2].

The above figure (Figure 1) shows the example of real time system in distributed environment. Real time system is highly dependable on deadline. The given task to the system must be completed with the allocated amount of time. Resultobtained after the given period of time is of no use in case of real time system. Some examples of real time system are Nuclear System, Robotic Controls, Medical equipment, defense system etc. Distributed systems are presented to the user as a single system image and has a feature of easily expanding the resources based the load to the system. Additional component can be added to the system, this additional component can be used in case of fault in the systemhelping to reduce the fault and for better output.

Below are few terminologies that are related to fault tolerance in distributed system.

Fault - At the lowest level of abstraction fault can be termed as "defect". It canlead to inaccurate system state. Fault in the system can be categorized based ontime as below:

- **Transient**: This type of fault occurs once and disappear
- **Intermittent**: This type of fault occurs many time in an irregular way
- **Permanent**: This is the fault that is permanent and brings system to halt.

Error – May be defined as state of the system, which is undesirable and maylead to failure of the system.

Failure – May be defined as faults due to unintentional intrusion. Differenttypes of failure are as below:

- **Crash Failure**: A server halts, but is working correctly until it halts
- **Omission Failure**: A server fails to respond incoming requests
  - **Receive omission**: A server fails to respond incoming message
  - **Send Omission**: A server fails to send message
- **Timing Failure**: A server's response lies outside the specified timeinterval
- **Response Failure**: The server's response in incorrect
  - **Value failure:** The value of the response is wrong
  - **State transition failure:** The server deviates from the correct flowof control
- **Arbitrary (Byzanthine) Failure**: A server may produce arbitraryresponses at arbitrary times

Fault Tolerance – Ability of system to behave in a well-defined manner upon occurrence of faults.

Recovery – Recovery is a passive approach in which the state of the system ismaintained and is used to roll back the execution to a predefined checkpoint.

Redundancy – With respect to fault tolerance it is replication of hardware,software components or computation.

Security – Robustness of the system characterized by secrecy, integrity,availability, reliability and safety during its operation.

It has been found that the need of fault tolerance is for the system Availability, Reliability, Safety, Maintainability and Security.

- **Availability**: The system must be usable immediately at any time.
- **Reliability**: The system must work for a long period of time withouterror.
- **Safety**: There should be no catastrophic consequences of temporalfailure.
- **Maintainability**: The system must be able to repair and fix the faultquickly and easily.
- **Safety**: The system should be able to resist the attacks against itsintegrity.

Failure masking by redundancy

- **Information redundancy**: Extra bits are added (e.g. CRC)
- **Time redundancy**: Action may be redone (e.g. transaction after abort)
- **Physical redundancy**: Hardware and software component may bemultiplied (e.g. adding extra disk, replicating the database), TMR.

Triple modular redundancy (TMR)

It uses the principle of building a majority of opinion. Each device is replicated 3 times, signal pass all 3 devicesIf one device fails, a voter can reproduce the correct value based on 2 correct signals. In this case it is assumed that at every stage 1 device and 1 voter may fail.

Literature Review

There is a lot of research that has already been performed and is ongoing in the field of fault tolerance in distributed system. Research and experimentation efforts began in earnest in the 1970s and continued through 1990s, with focused interest peaking in the late 1980s.

A number of distributed operating system were introduced during these period; however, very few of these implementations achieved even modest commercialsuccess.

Different authors have reviewed the concept of fault tolerance computing system, like Ramamoorthy [1967], Short [1968], Avizienes [1971], Khul and Reddy [1980, 1981], Bagchi and Hakimi [1991]. The SAPO Computer built in Prague, Czechoslovakia was probably the first Fault- Tolerant Computer built in 1950-1954 under the supervision of Antonin Svoboda, using relays and a magnetic drum and was operated in 1957-1960.

According to Leslie Lamport [3],Time should be used instead of timeout to increase the fault tolerance. A general method is described for implementing a distributed system with any desired degree of fault- tolerance. Instead of relying upon explicit timeouts, processes execute a simple clock-driven algorithm. For Byzantine problem solution author has assumed reliable clock synchronization.

According to Paval., *et al*. [4], the fault resilience techniques can be broadly classified into three categories as below:

- Hardware Resilience
- Resilient System Software
- Application Based Resilience.

According to Diego Zuquim Guimarães Garcia., *et al.* [5], the web service architecture still lacks the facilities to support fault tolerance. The author has provided an architecture that provides mediation and monitoring for web service.

According to Arvind Kumar., *et al.* [7], types of faults occurring in the system, fault detection and recovery techniques are discussed. A system after failure can be in one the three below:

- Fail Stop System
- Byzantine System
- Fail-Fast System

They have mentioned some approaches for fault tolerance in Real Time distributed system. They are as following:

- Replication
    - Job Replication
    - Component Replication
    - Data Replication
- Check pointing
- Scheduling/Redundancy
    - Space Scheduling/Redundancy
    - Time Scheduling/ Redundancy
    - Hybrid Redundancy

A common way to handle crashes involves two steps: (1) Detect the failure; and (2) Recover, by restarting or failing over the crashed component. Failure recovery has received a lot more attention than Failure detection. Joshua B. Leners., *et al.* [9], have given a fault detector mechanism named as Falcon. According to the authors Falcon achieves these features by coordinating a network of spies, each monitoring a layer of the system.

Padmakumari (2015) [10] has provided the idea for diverse fault tolerance and monitoring mechanism to enhance the reliability in cloud computing environment. In has given the data about various techniques and methods which are used for fault tolerance and also focused on future research direction in cloud fault tolerance. Joshi (2014) [11] has given the concept of virtual data centres (VDC) which is based on the migration technique. In this methodology if a virtual machine is overloaded then some of its resources are migrated to another virtual machine to handle the server failure. TT-based designs have proved to be a viable solution in the scope of adaptive systems and recent works in this area show that there is an on-going interest in continuing improving the RT-related features of the FTT protocol [12].

Fault tolerance model

Fault model Fault model describes which faults and associated rate of occurrence are assumed by the system being designed. According to different viewpoints the faults can be: system boundaries, internal or external; phenomenological cause natural or human-made; intent-deliberate or non-deliberate; capacity accidental or incompetence; persistence permanent or transient. Two examples of the faults considered by the presented fault model are

physical deterioration and physical interference as seen in figure 2. These faults are caused by processes such as radiation, power transients, noisy input lines, etc.

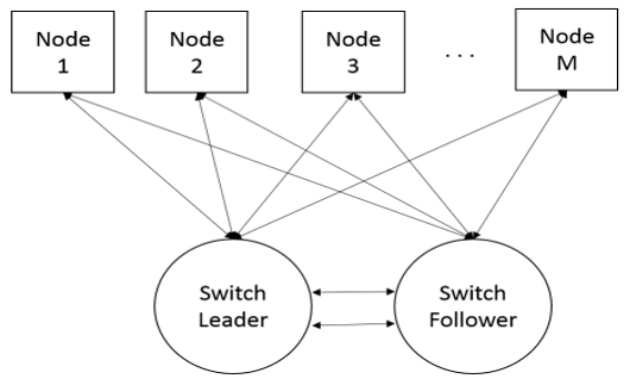

Figure 2: Fault tolerance model.

Summary and Conclusion

When a fault occurs in a system, then the system requires the fault tolerance method to detect the fault and recover the system to its normal state. Fault tolerance techniques are required to predict these failures and take appropriate action before these faults actually occurs. Fault detection is equally importantas failure recovery for having a better fault tolerance mechanism in a system.

All the activities executed by fault-tolerant system have to be synchronized, e.g. node replicas have to first execute certain application tasks in order to produce the results, then exchange the produced results by transmission/reception of messages using the network protocol, and execute application tasks that vote on the locally produced result and the results received through the channel.

After going through previous works in the field of fault tolerance, it is found that several fault tolerance models are present. Transient link faults may affect the capacity of a node for transmitting/receiving, but they are transparently tolerated by using the pro-active retransmission mechanism. Replication of Hardware and Software technique are the most common technique used for fault tolerance. Faults may lead a node replica to become desynchronized at the communication and/or the application level beyond the error recovery capacity. Thus, it was realized that it was necessary to propose more sophisticated recovery mechanisms to prevent undesirable fault attrition. These replication techniques may not be economical. There are many fault tolerance techniques proposed earlier but none of the single fault tolerance mechanism can fulfill all aspects of fault tolerance. The model can be used with necessary customization according to system that is being designed.

Bibliography

1. Flavin Cristian. "Understaning Fault-Tolerant Distributed Systems". Computer Science and Engineering, University of California, San Diego (1993).
2. Lakshmi PS. "Distributed Fault Tolerance Sytem in Real Time Environment". Kundal Kr. Medhi, *International Journal of Advance Research in Computer Science and Software Engineering* (2013).
3. Leslie Lamport. "Using Time instead of Timout for Fault Tolerant Distributed System". SRI International (2017).
4. Pavan B., *et al*. "Fault Tolerance Techniques for Scalable Computing". Mathematics and Computer Science Division, Argonne National Laboratory (2014).
5. Diego Z., *et al*. "A Fault Tolerant Web Service Architecture". Institute of Computing University of Campinas, São Paulo, Brazil (2016).
6. Avizienis A. "Fault Tolerance Computing-An overview". *IEEE Computer* 3 (2011).
7. Arvind Kumar., *et al*. "Fault Tolerance in Real Time Distributed System". *International Journal on Computer Science and Engineering* 3 (2011): 933-939.
8. Mitvin S. "Fault Tolerant Distributed System". Department of Computer Science and Engineering, University of Texas at Arlington (2019).
9. Joshua B L., *et al*. "Detecting Failure in Distributed Systems with FALCON spy network". The University of Texas at Austin, Microsoft Research Silicon Valley (2011).
10. Padmakumari P. "Methodical Review on Various Fault Tolerant and Monitoring Mechanisms to improve Reliability on Cloud Environment". *Indian Journal of Science and Technology* 8 (2015).
11. Joshi SC and Sivalingam KM. "Fault tolerance mechanisms for virtual data center architectures". *Photonic Network Communications* 28 (2014): 154-164.

12. Garibay-Martínez R. "Improved Holistic Analysis for Fork–Join Distributed Real-Time Tasks Supported by the FTT-SE Protocol". In: IEEE Transactions on Industrial Informatics 12.5 (2016): 1865-1876.