Fault Tolerance

1- Definition of Fault Tolerance

Fault Tolerance is the property that enables a system to continue operating properly in the event of the failure of (or one or more faults within) some of its components. If its operating quality decreases at all, the decrease is proportional to the severity of the failure, as compared to a naively designed system, in which even a small failure can cause total breakdown.

Fault tolerance is particularly important in high-availability or life-critical systems. The ability of maintaining functionality when portions of a system break down is referred to as **Graceful Degradation**.

Fault-tolerant Computing: is the art and science of building computing systems that continue to operate satisfactorily in the presence of faults.

Fault-tolerance describes a computer system or component designed so that, in the event that a component fails, a backup component or procedure can immediately take its place **with no loss of service**. Fault tolerance can be provided with software, or embedded in hardware, or provided by some combination.

Fault-tolerant Design enables a system to continue its intended operation, possibly at a reduced level, rather than failing completely, when some part of the system fails.

This term is most commonly used to describe computer systems designed to continue more or less fully operational with, perhaps, a reduction in throughput or an increase in response time in the event of some partial failure. That is, the

system as a whole is not stopped due to problems either in the hardware or the software.

An example in another field is a motor vehicle designed so it will continue to be drivable if one of the tires is punctured. A structure is able to retain its integrity in the presence of damage due to causes such as fatigue, corrosion, manufacturing flaws, or impact.

Highly Fault-tolerant System is a system that might continue at the same level of performance even though one or more components have failed.

The designer of a highly fault-tolerant system can begin by designing a system that tolerates only benign failures and then use some methods to automatically convert it into one that tolerates more severe failures. This simplifies the task of designing a highly fault-tolerant system to that of designing a much simpler system.

For example, a building with a backup electrical generator will provide the same voltage to wall outlets even if the grid power fails.

A fault-tolerant system may be able to tolerate one or more fault-types:

1. Transient, intermittent or permanent hardware faults,
2. Software and hardware design errors,
3. Operator errors, or
4. Externally induced upsets or physical damage.

2- Reliability and Availability with Fault-tolerant System

Reliability and availability have become increasingly important in today's computer dependent world. In many applications where computers are used, outages or malfunction can be expensive, or even disastrous. What if the computer system in a nuclear plant malfunctioning, or the computer systems in

a space shuttle **boots** just when the shuttle is about to land...? These are the more exotic examples. Closer to everyday life are the telecommunications switching systems and the bank transaction systems.

To achieve the needed reliability and availability, we need *fault-tolerant computers*. They have the ability to tolerate faults by detecting failures, and isolate defect modules so that the rest of the system can operate correctly.

Reliability techniques have also become of increasing interest to general-purpose computer systems. **Four trends contribute to the importance of Reliability techniques in fault tolerance**:

- **First** is that, computers now have to operate in *harsher environments*. Earlier, computers operated in clean computer rooms, with stable climate and clean air. Now the computers have moved out to industrial environments, with temperatures over a wide range, dust, humidity and unstable power supply. All these factors alone could make a computer fail.

- **Second**, *the users* have changed. Earlier, computer operators were trained personnel. Now, with an increasing number of users, the typical user knows less about proper operation of the system. The consequence is that computers have to be able to tolerate more. Haven't we all seen users swearing over a disappeared document in a text editor (Backup? What is that?), or heard about people that accidently have poured coffee into the computer?

- **Third**, the *service costs* increase relative to hardware costs. Earlier the average machine was a very expensive, big monster. At that time, it was common with one or several dedicated operators to keep the system up and running. Today, a computer is cheap, and the user has the job of being the "operator". The user can not afford frequent calls for field service.

- **Fourth** and last trend is *larger systems*. As systems become larger, there are more components that can fail. This means, to keep the reliability at an acceptable level, designs have to tolerate faults resulting from component failures.

What can cause outages of equipment, making fault-tolerance techniques necessary? They can be split into outages caused by:

- *Environment*: This is facilities failures, e.g. dust, fire in the machine room, problems with the cooling, earthquakes or sabotage.
- *Operations*: Procedures and activities of normal system administration, system configuration and system operation. This can be installation of a new operating system (requires booting of the machine), or installation of new application programs (which requires exit and restart of programs in use).
- *Maintenance*: This does not include software maintenance, but could be hardware upgrading.
- *Hardware*: Hardware device faults.
- *Software*: Faults in the software.
- *Process*: Outages due to something else, e.g. a strike.

In a modern system, fault-tolerance masks most hardware faults, and the percentage of outages caused by hardware faults are decreasing. On the other side, outages caused by software faults are increasing. According to a study on Tandem systems, the percentage of outages caused by hardware faults was 30% in 1985, but had decreased to 10% in 1989. Outages caused by software faults increased in the same period, from 43% to over 60%!