# Chapter 6. Design Patterns for Resiliency

# What is Resilient System?

Resiliency is a system’s ability to constructively deal with failures.

A resilient system detects failure and routes around it. Nonresilient systems fall down when faced with a malfunction.

# Traditional Approach

Traditional software assumes a perfect, malfunction-free world.

This leaves the hardware systems engineer with the impossible task of delivering hardware that never fails.

# The Distributed Computing Approach

Distributed computing, in contrast to the traditional approach, embraces components' failures and malfunctions.

It takes a reality-based approach that accepts malfunctions as a fact of life.

Example: Google Docs continues to let a user edit a document even if a machine fails at Google: another machine takes over and the user does not even notice the handoff.

# Resiliency through Spare Capacity

The general strategy used to gain resiliency is to have redundant units of capacity that can fail **independently** of each other.

Failures are detected and those units are removed from service.

The total capacity of the system is reduced but the system is still able to run. This means that systems must be built with spare capacity to begin with.

# Hot Spares

This strategy is to have primary and secondary replicas.

In this approach, the primary replica receives the entire workload but the secondary replica is ready to take over at any time.

This is sometimes called the **hot spare or “hot standby”** strategy since the spare is connected to the system, running (hot), and can be switched into operation instantly.

It is also known as an active– passive or master–slave pair.

# Failure Domains

A failure domain is the bounded area beyond which failure has no impact.

A failure domain may be prescriptive

A failure domain may be descriptive

# Software Failures

As long as there has been software, there have been software bugs. Long-running software can die unexpectedly. Software can hang and not respond

1. Software Crashes
2. Software Hangs
3. Query of Death

# Physical Failures

The physical devices used in a distributed system can fail on many levels.

Physical failures can range from the smallest electronic component all the way up to a country's power grid.

Providing resiliency through the use of redundancy at every level is expensive and difficult to scale. You need a strategy for providing resiliency against hardware failures without adding excessive cost.

# Types of Physical Failures

A) **Parts and Components:** RAM, Disks,power Supplies,Network Interfaces
B) **Machines:**
C) **Load Balancer: The presence of a load balancer means the system scales and is resilient automatically. This is NOT true**
D) **Racks**
E) **Datacenters**

# Overload Failures

A) Traffic Surges
   a) Dynamic Resource Allocation
   b) Load Shedding

B) DoS and DDoS Attacks:

C) Scraping Attacks:

# Quiz