Diagnostics, fault and change management

Dr. G. Usha Devi
Associate Professor
School of Information Technology and Engineering
VIT University
Vellore

Principle 43 (Predictable failure). *Systems should fail predictably so that they can be recovered quickly. Predictability is encouraged by adopting standardized (or well-understood) protocols and procedures for quality assurance in design and maintenance.*

Fault tolerance and propagation

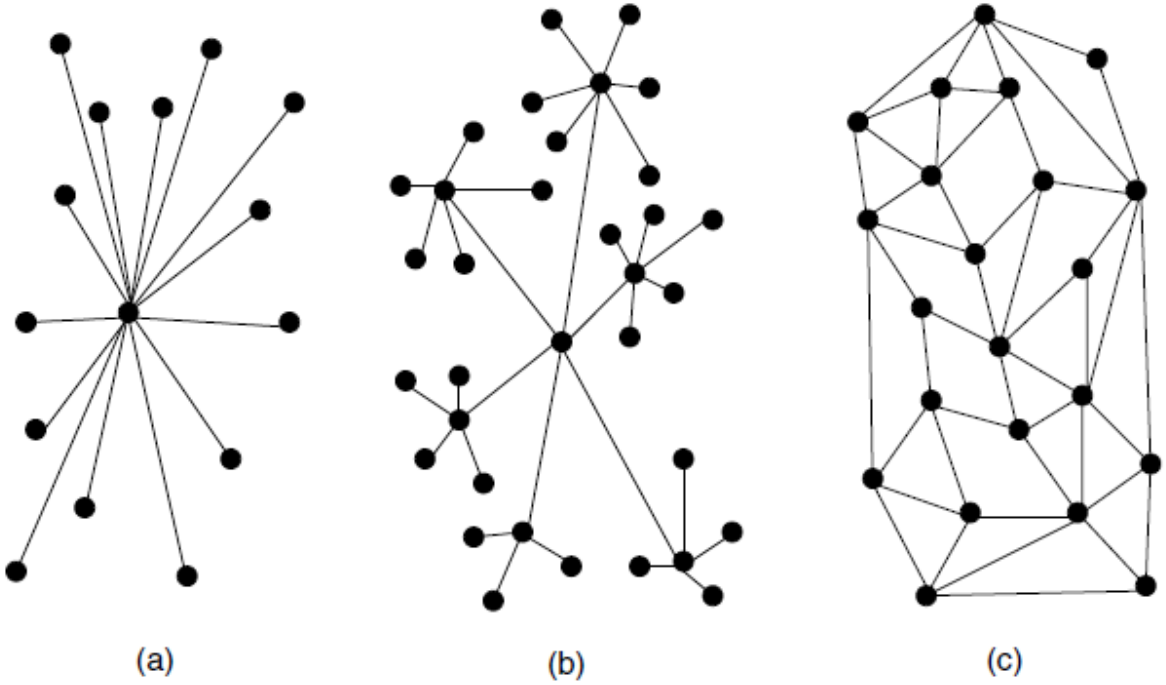

Figure 8.1: Network topologies: (a) centralized, (b) decentralized or hierarchical, and (c) distributed mesh.

If a system is tolerant to faults and security breaches, then we can look at it in one of two complementary ways:

- The access network that allows problems to propagate is poorly connected; i.e. connections (security breaches) between nodes (resources) are absent.
- The resource network is well connected and is resilient to removal of nodes(resources) and connections (supply channels).

- A *tolerant network* is robust to node removal and connection removal. Node removal is usually more serious

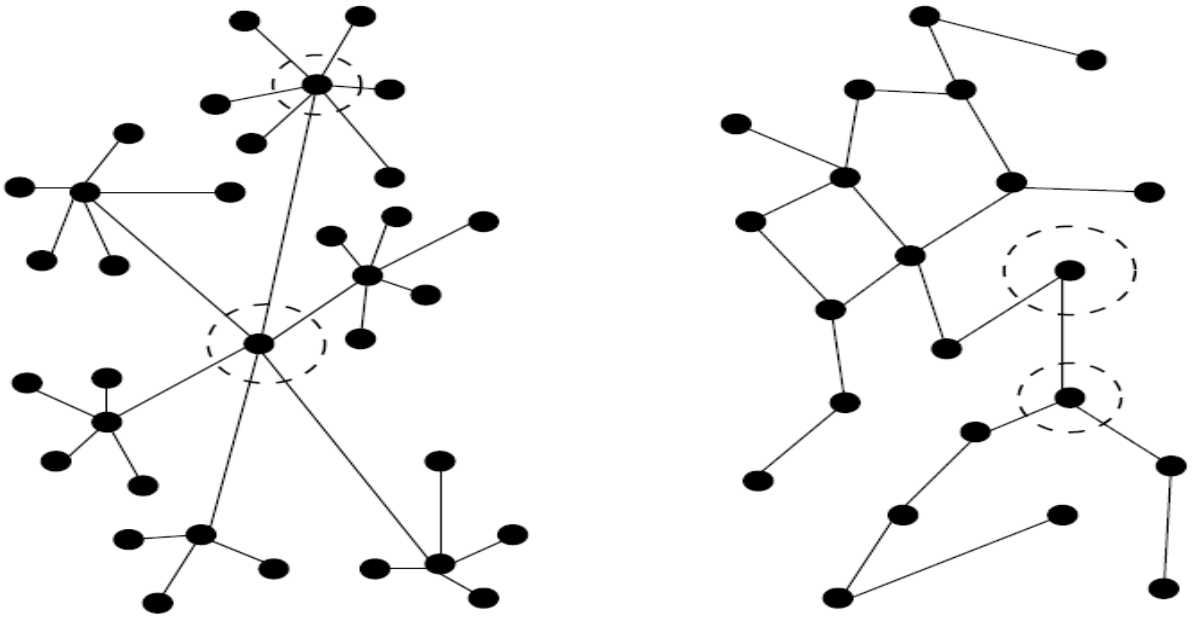

Figure 8.2: Network tolerance to node removal: nodes are more important than connectors.

- **Definition 6 (Small-world network).** *There is a class of highly clustered graphs that behave like random graphs. These are called small-world networks*
- To summarize, the reason why networks are important to human–computer systems is this: the ease with which information flows through a network depends on how well it is connected; this affects
  - The propagation of faults
  - Security breaches
  - The likelihood of emergent properties (bugs).

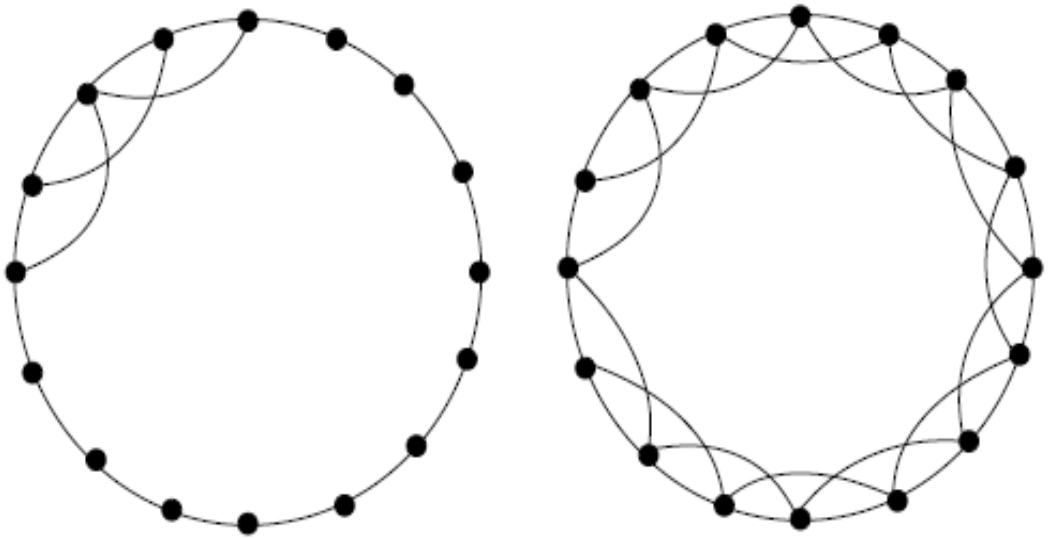

Figure 8.3: A network is built up by adding connections between neighbors. As more distant neighbors become connected, small, local clusters become connected over longer distances.

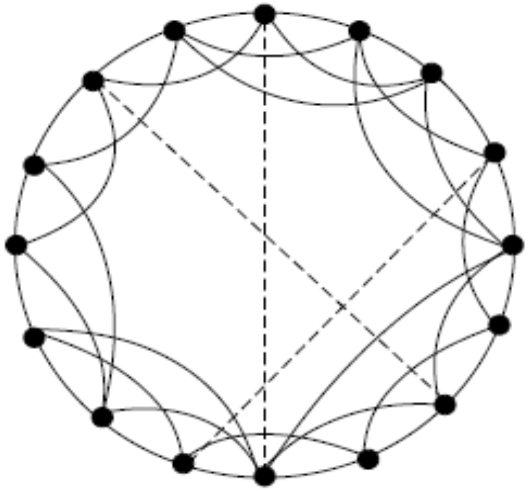

Figure 8.4: In a small-world network, weak links to distant neighbors provide important short-cuts that dramatically reduce the distance between random individuals.

Causality and dependency

- **Principle 44 (Causality).** *Every change or effect happens in response to a cause that precedes it.*
- Causality is a mapping from cause to effect

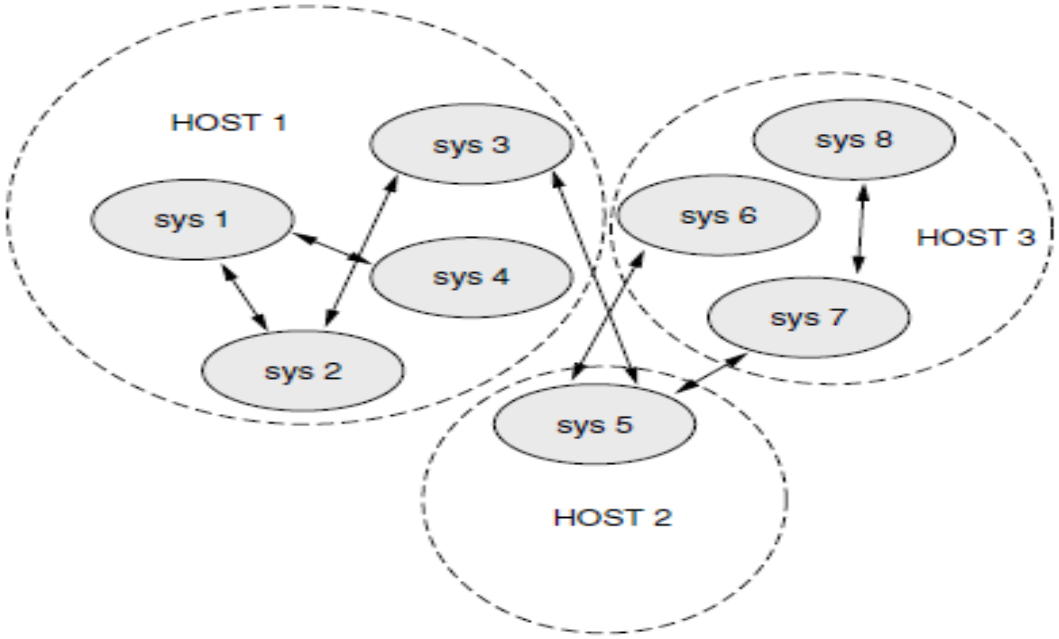

Figure 8.5: A complex system is a causal web or network of intercommunicating parts. It is only possible to truly isolate a subsystem if we can remove a piece of the network from the rest without cutting a connection. If we think of the total system as $S(x_1 \ldots x_n)$, and the individual subsystems as $s_1(x_1 \ldots x_p), s_2(x_p \ldots x_n)$ etc, then we can analyze a subsystem as an open system if the subsystems share any variables, or as a closed system if there are no shared variables.

Faults

The IEEE classification of software anomalies is

- Operating system crash
- Program hang-up
- Program crash
- Input problem
- Output problem
- Failed required performance
- Perceived total failure
- System error message
- Service degraded
- Wrong output
- No output.

Another source of error is found at the human edge of the system:

- Management error
- Miscommunication

Forgetfulness

- Misunderstanding/miscommunication
- Misidentification
- Confusion/stress/intoxication
- Ignorance
- Carelessness
- Slowness of response
- Random procedural errors
- Systematic procedural errors
- Inability to deal with complexity
- Inability to cooperate with others

How are faults corrected?

The basic primitives are:

- Examining files
- Creating files
- Aliasing files
- Replacing files
- Renaming files
- Removing files
- Editing files
- Changing access rights on files
- Starting and stopping processes or threads
- Signaling processes or threads
- Examining and configuring hardware devices.

Fault report and diagnosis

Error reporting

Cause trees

- Charting cause trees is a systematic method used in fault diagnosis.
- The idea is to begin by building lists of possible causes, then causes of those causes, and so on, until one has covered an appropriate level of detail.
- Once a cause tree has been constructed for a system, it becomes a road-map for fault finding for the future also.
- The use of cause trees is sometimes called *Root Cause Analysis* (RCA).
- A related method called *Event Tree Analysis* (ETA) maps out every single eventuality as a true/false binary tree, where every possibility is documented but only certain pathways actually occur

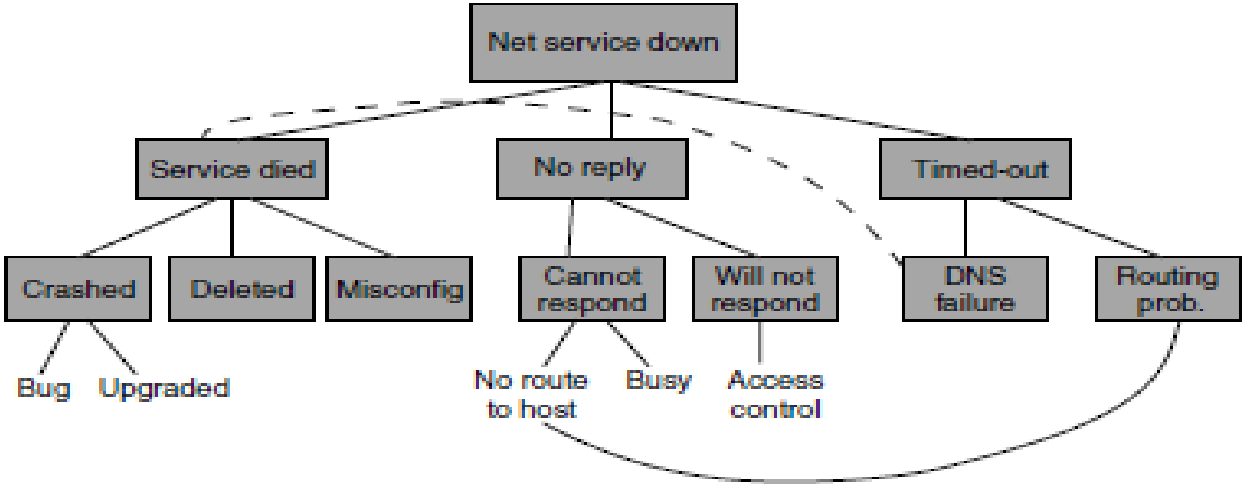

Figure 8.7: Attempt at cause tree for a missing network service.

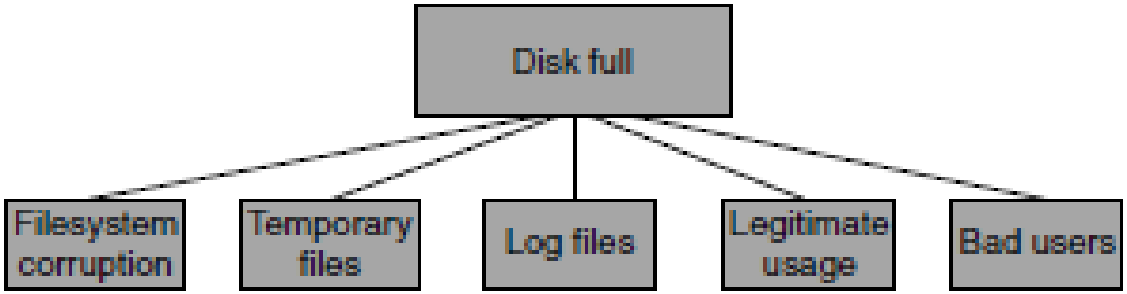

Figure 8.8: Attempt at cause tree for a full disk.

Cause analysis can be used at different levels.

At human management level, the heuristic roles are:

- Inadequate procedures
- Inadequate training
- Quality control
- Miscommunication
- Poor management
- Social/human engineering
- Supervision error
- Preventative maintenance lacking.

Probabilistic fault trees

Faults

For the purposes of modeling, fault tree analysis distinguishes between:

- *Failures*: abnormal occurrences that do not prevent the system from functioning.
- *Faults*: systemic breakdowns within the system.

An important subset of faults is formed by *component faults*.

Component faults fall into three categories:

- *Primary faults*: occur when a component is working within its design limits, e.g. a web server that is rated at 50 transactions per second fails when it reaches 30 transactions per second.
- *Secondary faults*: occur when a fault is operating outside its design specification, e.g. a web server that is rated at 50 transactions per second fails when it reaches 90 transactions per second.
- *Command faults*: are faults that occur when a system performs its specified function, but at the wrong time or place, e.g. a Web server that begins querying a database persistently when no request is being made by an external agent.