Carrier-Grade, High-Availability Computing Platforms for Voice and Data Networks

Definition

Communication service providers face the seemingly impossible challenge of maintaining service continuity while regularly making changes and upgrades to network equipment, computing platforms, and application software. To remain competitive in a world that operates 24 hours a day, seven days a week, service providers must protect against unexpected failures while managing the evolution of service logic, platform software, and system hardware. Thus, solutions such as the HP OpenCall computing platform have been designed to tolerate hardware failures and reduce the risk of service outage as a result of software failures.

Overview

This tutorial discusses the challenge of providing continuous service availability and presents an overview of the technologies that can be used in ensuring mission-critical service continuity for today's computer-based communications network.

Topics

1. Introduction
2. Voice and Data—The History
3. Leveraging the Legacy Infrastructure
4. Moving Forward: Converged Switching Architecture
5. Opportunities for Carriers
6. The Next-Generation Carrier
Self-Test
Correct Answers
Glossary

1. Ensuring Continuous Availability of Communication Services

The Need for Software Fault-Tolerance

Services in today's communications networks must be highly robust. With the nearest competitor only one or two mouse clicks away, unplanned service downtime causes revenue loss and, in some cases, contractual penalties.

The computer industry has historically been concerned about hardware failures. Today, however, with amazingly high hardware reliability, the emphasis is shifting to software. Hardware reliability has increased because of process improvements in manufacturing and testing, and through the use of redundancy of those components most likely to fail. Redundant array of inexpensive disks (RAID), redundant fans, and redundant power supplies are examples of design improvements in today’s midrange computer systems.

The increasing complexity of software causes growing concern about the robustness of system software components. This is a direct consequence of the complexity that this software represents. Even when exceptional caution is taken, simple logic errors can result in catastrophic failures. The midair explosion of the first Arian-5 rocket and the massive failure of the North American signaling system 7 (SS7) network are recent examples.

Hardware fault tolerance does not protect against software failure; indeed, it may even replicate the failure. Thus, protection from software failure becomes the central challenge. It is practically impossible to prevent software failures completely, and design approaches that virtually eliminate software failures (typically used only when human life is at risk) can more than triple development costs and still not guarantee that failures will not occur.

This implies that solutions must be designed to tolerate hardware failures (e.g., through simple replication of hardware resources) while using software architectures that reduce the risk of service outage as a result of software failures.

Benefits of Software Replication

To achieve hardware fault tolerance, it is necessary to replicate all critical hardware components. Not surprisingly, the same principle applies for software fault tolerance. In order to protect a software application from software faults, the application’s logic and data must be replicated. Typically, the application’s logic and data must be distributed on a cluster of computer systems to ensure that it can tolerate any single hardware or software fault within the cluster.

As well as providing a higher level of service availability, software replication can also bring other benefits. It can be used to address scalability requirements, through load-sharing (n+1) architectures. It can be used as a basis for on-line upgrade procedures, at both the platform (hardware + operating system) and application level. It allows the use of standard off-the-shelf computer platforms, enabling service providers to benefit from the latest hardware technology advances and increased processor speeds. This typically ensures a better price-to-performance ratio when compared with a hardware fault-tolerance approach.

Of course, software replication technology can also be used in conjunction with hardware fault-tolerant systems; the use of one does not exclude the other.

However, the cost of developing software in a way that guarantees high availability should not be underestimated. The mechanisms required to replicate application logic and data can be complex and error-prone, leading to less stable solutions. The challenge facing the computer industry is to develop generic software replication frameworks that can be reused by a wide range of applications.

A generic software replication framework must address the following issues:

- detecting software faults
- ensuring rapid application-level recovery from failures
- providing a persistent store for application state information, so that it can tolerate hardware and software faults
- providing a single-system view of the replicated software, in particular on the management and provisioning interfaces

2. Communication Service Requirements

Multiple software replication architectures exist. A first level of software fault tolerance can be achieved by simple replication of the application on a cluster of computer systems. If one instance of the application fails, it can be restarted, either on the same system or on an adjacent system within the cluster. This architecture provides crash-recovery semantics, where some data may be lost when the application fails, and the recovery time includes the time to restart the application. However, such semantics are not considered to be sufficient in the case of revenue-generating communication services.

In the case of communication services, continuous availability is considered to be the primary goal. Given that it is impossible to avoid software faults, continuous service availability can only be achieved if the time to detect a failure and recover from that failure is low. This implies that failure recovery time must be kept to an absolute minimum, where the recovery time must include the time to recover both the service logic and the service data to a fully operational state.

To achieve rapid application-level failure recovery, the application data must be continuously available. By regularly checkpointing the application's state to a persistent database or to a peer instance, data loss on failure can be minimized, and other instances of the application can participate in the recovery process. As an example of this approach, in the case of the HP OpenCall IN platform, all critical state information is held in a replicated in-memory database. If an active instance of the platform fails, a standby system can take over. Service continuity is ensured, as the in-memory database contains the latest application state information. Furthermore, the replication of this state information is managed by the platform, rather than by each application. As a result, the application developer does not need to be aware of the complex database replication and synchronization algorithms. This reduces the complexity of the application logic, leading to more robust solutions.

Ensuring continuous availability of a communication service requires that a number of other key issues be addressed:

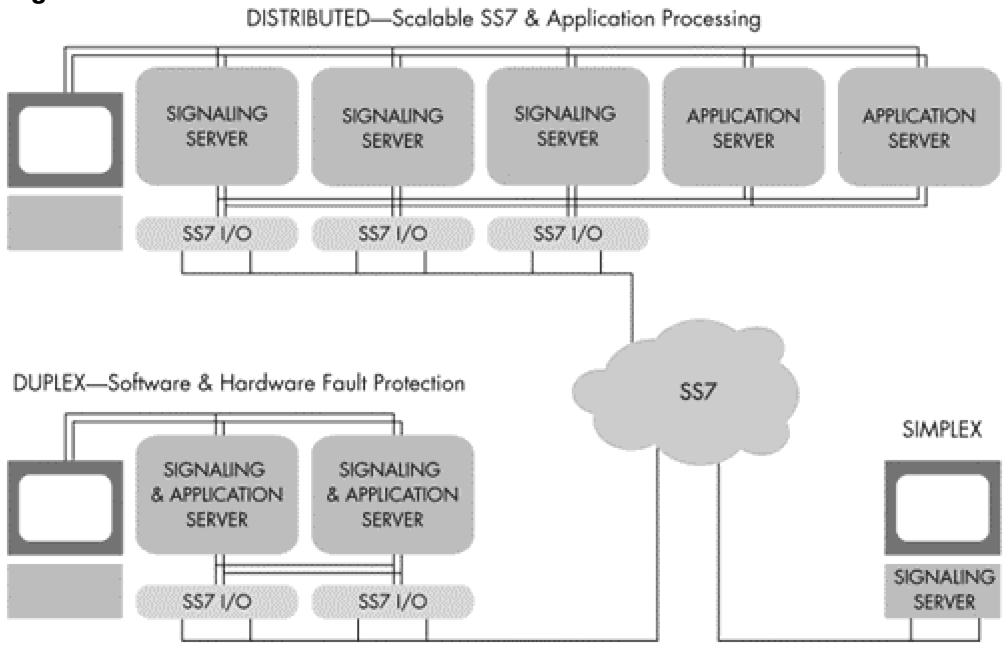
- how to ensure continuous network connectivity
- how to manage application and platform evolution
- how to ensure service-level fault isolation
- how to manage multisite scalability (required for both greater performance and protection against site failures)

Continuous Network Connectivity

All communication services require network connectivity. A software framework aimed at ensuring continuous service availability must ensure fault resilience at the network interface. If the network interface is not reliable, then the application will not be accessible from other network elements, and the service provider will not perceive the service as being continuously available.

In the case of the intelligent network (IN) services in the public switched telephone network (PSTN), a fully redundant interface to the SS7 network is required to ensure continuous network connectivity. As an example of this approach, the OpenCall SS7 signaling interface system has been designed to provide both scalability and fault resilience. The SS7 interface card is based on the standard peripheral component interconnect (PCI) bus. The telecommunication system unit (TSU) card cage, which is fully network equipment building systems (NEBS)–compliant, has also been developed. In combination with the card, the TSU enables full use of the SS7 protocol's load-sharing and fault-tolerance capabilities. The TSU is connected to the computing hardware via dual local-area networks (LANs). A TSU can hold several T1/E1/V35 cards and will monitor and manage individual card failures. For scalability and redundancy, several TSUs can be deployed. Cards within the TSU are hot-swappable, and TSUs can be deployed on-line to add capacity or make repairs without downtime. This provides a truly scalable, fault-tolerant network interface, which supports up to 256 SS7 links on a single point code.

Figure 1. SS7 Hardware Architecture


DISTRIBUTED—Scalable SS7 & Application Processing
SIGNALING SERVER
SIGNALING SERVER
SIGNALING SERVER
APPLICATION SERVER
APPLICATION SERVER
SS7 I/O
SS7 I/O
SS7 I/O
DUPLEX—Software & Hardware Fault Protection
SIGNALING & APPLICATION SERVER
SIGNALING & APPLICATION SERVER
SS7 I/O
SS7 I/O
SS7
SIMPLEX
SIGNALING SERVER

The software that implements the higher layers of the communication protocol must also be both scalable and fault tolerant. The OpenCall SS7 software can be deployed on either a mono- or multiprocessor stand-alone computer system or in a cluster environment. With OpenCall SS7 fault-tolerance controller (FTC), a redundant SS7 stack can be deployed, ensuring cluster component failover and faster switchover (typically two to three seconds) of the SS7 protocols. Based on sophisticated algorithms, the FTC triggers a switchover, whether an error occurs in the hardware, the operating system, or in OpenCall itself. The FTC also notifies the SS7 application of any switchover, allowing the application to take any necessary recovery actions. OpenCall SS7 allows several processes to connect to both the primary and the secondary SS7 stack simultaneously. This enables a very fast switchover when an application detects an internal inconsistency but does not want to trigger a switchover of the SS7 stack. The secondary application takes over but continues to communicate via the primary SS7 stack.

Service and Platform Evolution

Having the ability to upgrade all aspects of a network service execution platform is a basic requirement. However, such upgrades can cause planned downtime.

Service Life Cycle

- **subscriber base**—The success of a new service often requires an upgrade to the underlying platform to meet the demands of the growth in subscribers. Increasing platform capacity drives demand for additional disk and random-access memory (RAM) memory. In an SS7 network, an increase in the number of simultaneous calls adds pressure to expand platform capacity. Greater platform capacity typically results in a need for more SS7 links as well as a need to increase the number of simultaneous service instances that the platform can handle (requiring an increase in the number of central processing units [CPUs] and their capacity).

- **service evolution**—Services themselves evolve over time. This requires the platform to support deployment of new versions (revision levels) of a service. It usually also requires the platform to simultaneously support several versions of the same service until support for the old versions is discontinued.

- **deployment of new services**—Service providers must continuously deploy new services to stay competitive. Service providers typically install several services on the same platform to reduce the total cost of ownership (TCO) (e.g., limit management overhead and the number of point-codes in the network). As new services become available, the services are deployed on the same platform. Therefore, it is extremely important that a newly deployed service will not cause existing profitable services to fail.

Platform Evolution

The communications industry greatly benefits from the downward cost trend of standard computing platforms. Such cost pressure forces a shortened computer platform life cycle, which results in significant hardware and operating system changes over just a few years. To ensure continued support, it becomes necessary to upgrade the hardware and operating systems, even for platforms only a few years old. The use of loosely coupled computer systems is essential for managing the evolution of platforms and operating systems. As long as any single failure point is eliminated, individual computers in a loosely coupled system can be taken off-line, upgraded, and brought back on-line without disrupting the cluster's availability. All software components must support the required resynchronization to enable a smooth reconnection.

Example

A single example illustrates the need for both service and platform upgrades. To illustrate why service and platform upgrades become necessary, consider a service such as Internet call diversion. Internet call diversion is a cost-effective means of offloading Internet traffic from the public switched telephone network (PSTN). With the explosive growth of the Internet, however, this service capacity must expand and evolve. As these standards get established or as Internet service providers (ISPs) seek differentiation, platform evolutions will also be necessary to support enhanced versions of the service, such as adding support for Internet call waiting. It is unlikely that exactly the same hardware, operating system, and service logic can be used to meet these ever-changing requirements.

Service-Level Fault Isolation

Communication services can often be modeled as a large number of state machines executing in parallel, with each service instance responsible for one call or one service request. Examples include the following:

- **freephone services**—where each instance is responsible for translating a dialed freephone number
- **number portability services**—where each instance must determine whether or not a dialed number has been ported to a new service provider
- **network-based call center servers**—where each instance is responsible for directing an incoming call to the most appropriate call center agent

Deployments of such services will result in multiple instances of these services running in parallel. In such an environment, fault isolation and service upgrade become critical requirements.

Fault isolation must ensure that a software bug in one instance of the service does not disrupt other executing instances of the service. Furthermore, with tens of thousands of simultaneous calls, a software bug in one instance should not crash the entire execution environment. Instead, it must be possible to detect and isolate errors at a service instance level.

On-line upgrade of service logic is required to allow service providers to deploy new logic for a service, without needing to shut down the service execution environment and without disrupting existing service instances.

The use of virtual machine (VM) technology can address both of these requirements. As an example, the HP OpenCall IN platform uses VM technology to provide a high-level service execution environment that monitors service instances, ensuring that errors are detected quickly, with minimum disruption of other service instances. By offering an interpreted environment for service execution, this platform can also trap software bugs that would normally crash the execution environment.

A VM solution also allows on-line deployment of new services and of new versions of existing services. By controlling the deployment of service logic, the VM can enable the upgrade of services on-line, even while earlier revisions of the services are executing.

VM technology can also offer rapid prototyping capabilities, allowing services to be quickly developed, simulated, and tested, prior to deployment.

Managing Multisite Distribution

Failure against site failures is also an important consideration for communication services. The ability to scale service logic and service data across multiple sites provides an extra degree of fault tolerance as well as scalability.

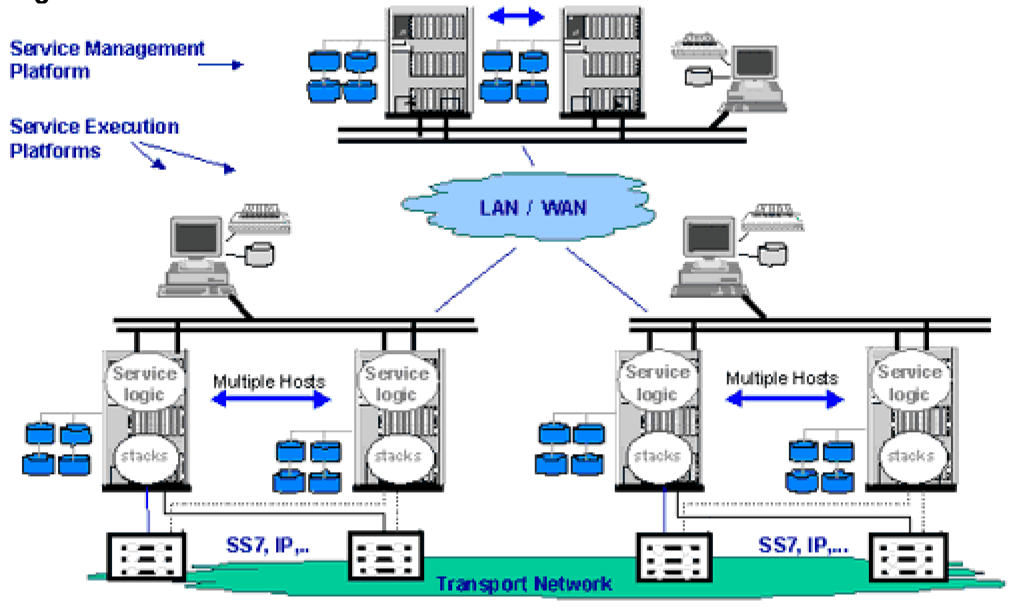
A multisite solution requires an extra layer of management to handle the distribution of logic and data across multiple sites. It also requires additional intelligence in the network, as the network must deliver service requests to the correct site or to alternative sites in the case of a site failure. In the SS7 network, signal transfer points (STPs) are typically deployed to meet this need.

Communication service platforms must also provide a single-system view to back-end systems, such as network management systems, data-provisioning applications, and customer-care systems.

One approach is to adopt a client/server architecture for service logic and service data distribution, according to which a centralized server acts as the single interface to the backend applications, while serving a number of network-facing client systems.

The HP OpenCall Service Management Platform (SMP) is an example of such a solution.

Figure 2. SEP–SMP Architecture



A single instance of the SMP can control multiple IN network elements. All service-deployment and data-provisioning requests are directed to the SMP, which then performs the necessary updates on the IN network element. In addition, the SMP is also responsible for the distribution of data updates across these IN elements.

The IN elements can operate in either mated-pair or *n*+1 mode. In a typical configuration, some logic and data will need to be distributed to all IN elements, some will be replicated on two elements (for fault tolerance), and some noncritical logic and data might only reside on a single IN network element.

Of course, the SMP must also be replicated, to avoid being a single point of failure on the service deployment and data-provisioning path.

3. Conclusion

With the increasing reliability of standard computer systems, the focus of the communication industry is shifting increasingly to software fault tolerance and software replication to ensure continuous availability of communication services.

As this tutorial demonstrates, continuous service availability must be addressed at many different levels:

- providing a persistent store for application data
- ensuring reliable network connectivity

- detecting and isolating service level faults
- addressing on-line upgradability of both the platform and the application
- offering protection against site failures, through the multisite distribution of service logic and service data

A software framework that aims to ensure continuous service availability must address all of these issues.

Some companies now offer solutions that can address the software reliability issue and that can ensure continuous availability of revenue-generating services.

By focusing on software replication and software upgradability, these solutions significantly increase service availability beyond the level that can be achieved through simple hardware fault tolerance.

Self-Test

1. When addressing continuous service availability, software fault tolerance presents a bigger challenge than hardware fault tolerance.

   a. true

   b. false

2. Analysts predict that data will consume __________ percent of the world's bandwidth by 2003.

   a. true

   b. false

3. It is possible to prevent software failures completely.

   a. true

   b. false

4. Software replication techniques are incompatible with hardware fault-tolerant solutions.

   a. true

   b. false

5. OpenCall SS7 software can be deployed on which of the following?

    a. a monoprocessor stand-alone computer system

    b. a multiprocessor stand-alone computer system

    c. a cluster environment

    d. all of the above

    e. a and b only

6. Service providers can reduce their system ownership costs by deploying multiple services on the same platform.

    a. true

    b. false

7. Continuous service availability requires fault resilience at the network interface.

    a. true

    b. false

8. With VM technology, a software bug in one instance of the service will likely crash the entire execution environment.

    a. true

    b. false

9. In order to protect against site failures, it is necessary to distribute service logic and service data across multiple sites.

    a. true

    b. false

10. In the SS7 network, IN network elements can only operate in mated-pair node.

    a. true

    b. false

Correct Answers

1. When addressing continuous service availability, software fault tolerance presents a bigger challenge than hardware fault tolerance.

    a. true

    b. false

    See Topic 1.

2. Analysts predict that data will consume ___________ percent of the world's bandwidth by 2003.

    a. true

    b. false

    See Topic 1.

3. It is possible to prevent software failures completely.

    a. true

    b. false

    See Topic 1.

4. Software replication techniques are incompatible with hardware fault-tolerant solutions.

    a. true

    b. false

    See Topic 1.

5. OpenCall SS7 software can be deployed on which of the following?

    a. a monoprocessor stand-alone computer system

    b. a multiprocessor stand-alone computer system

    c. a cluster environment

    d. all of the above

    e. a and b only

See Topic 2.

6. Service providers can reduce their system ownership costs by deploying multiple services on the same platform.

   a. true

   b. false

   See Topic 2.

7. Continuous service availability requires fault resilience at the network interface.

   a. true

   b. false

   See Topic 2.

8. With VM technology, a software bug in one instance of the service will likely crash the entire execution environment.

   a. true

   b. false

   See Topic 2.

9. In order to protect against site failures, it is necessary to distribute service logic and service data across multiple sites.

   a. true

   b. false

   See Topic 2.

10. In the SS7 network, IN network elements can only operate in mated-pair node.

    a. true

    b. false

    See Topic 2.

Glossary

CPU	central processing unit
FTC	fault-tolerance controller
IN	intelligent network
ISP	Internet service provider
LAN	local-area network
NEBS	network equipment building standards
PCI	peripheral component interconnect
PSTN	public switched telephone network
RAM	random access memory
SMP	service management platform
STP	signal transfer points
TCO	total cost of ownership
TSU	telecommunication system unit
VM	virtual machine