

High Availability application ready platform design

After the downturn of the last several years, the telecom market is bouncing back and is once again cautiously expanding converged and wireless networks. There is a new and growing demand from consumers and corporate customers for converged voice and data services. This has challenged the Telecom Equipment Manufacturers (TEMs) to provide new and innovative services at unprecedented costs and schedules.

The challenges faced by the large TEMs today are unique because their staff size is not tracking their business growth due to the massive staff reductions of the past few years. With tight budgets, aggressive schedules, and precious few resources, the TEMs must make tough development resource choices.

By Asif Naseem

Competitive pressure

The emergence of services using IP has brought unconventional players to the market, such as fast and nimble startup companies. This is intensifying the competition, and forcing TEMs and service providers to come up with new development and deployment strategies that enable them to quickly bring cost effective applications and services to the market.

Traditional platforms

Service providers generate revenue from applications and services that run on platforms generally provided by the TEMs. A high level

view of a platform consists of the following four layers shown in Figure 1:

- Applications
- Middleware
- Operating System
- Platform Hardware

The lower three layers represent a platform that must be built or acquired before the end-user applications in the top layer can be developed and deployed.

Proprietary platforms

Traditionally, large equipment manufacturers have invested enormous resources to build in-house hardware and middleware for proprietary platforms.

While such implementations address the specific requirements of a particular application or service, they require significant rework each time change is introduced in the hardware, the operating system, or the application. This makes the reuse of most of the functionality layers very costly, time consuming, and risky. Even though TEMs recognized the large costs and efforts involved, they were forced to build proprietary platforms due to the lack of Commercial Off-The-Shelf (COTS) components that were standards-based, pretested, and pre-integrated.

Proprietary platform projects have invariably faced the following challenges:

- Long development and integration cycles due to proprietary function development, and third party and legacy component integration.

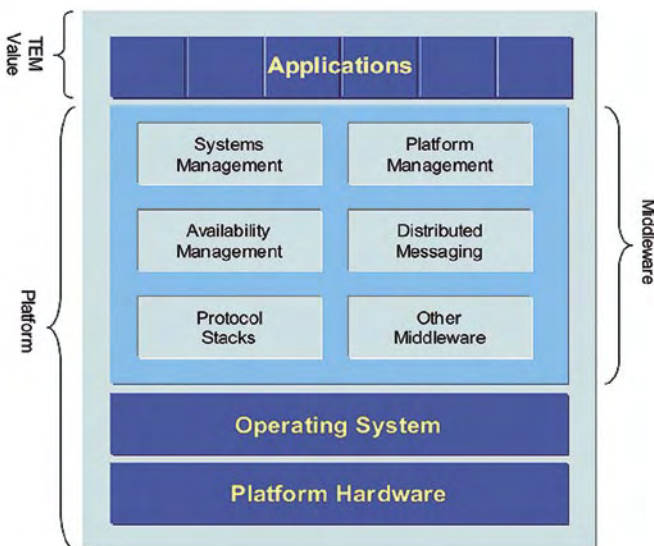


Figure 1

- Product commercialization cycles that are measured in years (2–3 years for a major release), when the market demands a much faster time-to-market (12–18 months).
- Missed deadlines caused by underestimated development and integration tasks.
- Competitive price pressures due to late market introduction, and high development cost.

Paradigm shift

In the last few years, market realities have led to a paradigm shift that has changed the way TEMs address these challenges. Now, the following efforts are taking place:

- Third party integration: Tight budgets, aggressive time-to-market requirements, increasing cost pressures, and fewer resources have led the TEMs to integrate third party hardware and middleware into their platforms.
- Standardization: Several new and emerging standards allow designers to build systems by combining a set of interoperable COTS building blocks from a variety of competing vendors. Numerous industry consortia are addressing the standards associated with the layers shown in Figure 1.

Application ready platform

Once a TEM has integrated third party and standardized COTS components into their platform, they are free to concentrate on revenue generating application development.

To quickly integrate, a TEM can acquire an *application ready platform* with pre-integrated hardware, middleware, and third party components. Each platform is pretested to ensure that it meets the TEM's most stringent requirements.

High availability

An application-ready platform must meet strict High Availability (HA) requirements before it is deployed in a mission or business critical application. Most mission critical applications require a system uptime of 99.999% (a downtime of less than five minutes per year) or better.

To ensure continuous operation and uninterrupted service, the application-ready platform designers must identify all potential single points of failure in the service path, and then implement methods to eliminate them. A comprehensive approach to designing a high availability application-ready platform includes:

- The upfront determination of key availability and performance requirements.
- The definition of a system architecture that supports key build vs. buy decisions, and leverages COTS elements.
- A special focus on key high availability attributes.

We now will examine the three development points.

Requirement determination

Careful upfront planning is required when you determine availability and performance requirements for the desired service, and when you allocate an appropriate budget for each required subsystem for the service.

One of the fundamental requirements of a highly available system is to minimize, if not eliminate, the time it takes to return the system to an operational state following a failure. This requires the definition of efficient availability services, which execute at a sufficient speed so a service interruption is imperceptible to the user.

For example, to support continuous service in a wireless network, failure recovery must take place in less than 50 ms. Restoring a system to an operational state following a failure is a multistage process that involves fault detection, diagnosis, isolation, recovery, repair, and reconfiguration. For our wireless network example, each recovery stage must be allocated a performance budget so that the total recovery process completes in less than 50 ms.

Modern communication systems often consist of multiple nodes connected in a cluster. This topology allows designers to include redundant nodes in the cluster that can take over operation of other nodes in the event of a failure. Collectively, these nodes work as a unit to provide a variety of communication services, and they must therefore have the capability to send and receive messages at very high speeds (such as event notifications, heartbeats, alarms, or application state information).

Such a messaging mechanism must provide efficient means of storage, retrieval, and transmission of real-time information within a node and across the cluster. For example, to support an uninterrupted wireless communication service, the messaging system must support anywhere from 10k–50k messages per second depending on the message size. These performance requirements often demand that an efficient in-memory data-store is used by the system for storage and retrieval of real-time information.

System architecture definition

In the 1990's, the computer industry transitioned from vertically integrated systems that were primarily offered by individual system vendors, to modular computing where an enterprise or an end-user can purchase COTS components from different vendors and put together an operational system with relatively little effort.

“The development of an application-ready platform that will provide uninterrupted service availability is an arduous task that involves complex hardware and software.”

The telecom industry is beginning a similar transition and faces several challenges. Traditionally, hardware and middleware – in addition to the core communication applications – were considered as TEM market differentiators. As plentiful development resources were at-hand, most large TEMs pursued programs to implement most, if not all, of the system functionality in-house.

Recently, however, several important trends – especially on the standardization front – are causing the telecom industry to consider adopting similar system design approaches as the computer industry.

For example, PCI Computer Manufacturing Group (PICMG), a consortium of several companies including TEMs, develops and promotes carrier grade equipment standards. A more recent set of specifications, AdvancedTCA, is quickly gaining wide industry acceptance. It primarily targets telecommunication application developers, and defines new architecture standards designed for integration ease and the migration of telecom applications across platforms.

The Open Software Development Laboratory (OSDL) is an industry body dedicated to accelerating the adoption of Linux for enterprise computing and carrier applications. The Carrier Grade Linux (CGL) working group of OSDL is defining feature roadmaps and specifications for use in telecommunications architectures.

The Service Availability Forum (SAForum) is a vendor consortium that develops and promotes standard specifications that allow for easy interoperable middleware COTS component integration. They publish two specifications:

- The Hardware Platform Interface (HPI) defines a standard interface between the high availability middleware and the hardware platform. It is an established standard with currently available commercial implementations.
- The Application Interface Specification (AIS) establishes an interface between the HA middleware and the application layer. This specification is currently in revision.

These specifications and standards facilitate the portability of middleware and applications across multiple platforms, and therefore reduce the startup cost and the integration effort.

The proliferation of the platform standards enables designers to rapidly build application-ready platforms that utilize various COTS building blocks as shown in Figure 2.

This allows the equipment vendors to minimize the cost and effort involved in building a platform, and allows them to focus their resources on their core competence – communication applications.

The intended system architecture must take into consideration these market trends and realities. Well designed system architecture meets service availability, cost, and schedule requirements, and the modular design of the system allows for future changes and/or replacements of modules without risking enormous cost and effort.

High availability attributes

The development of an application-ready platform that will provide uninterrupted service availability is an arduous task that involves complex hardware and software.

Our experience shows that such systems evolve over multiple releases until they are able to meet the most stringent service availability requirements. Some of the most critical attributes that characterize these highly available systems are:

- the ability to thoroughly model various resources in the platform
- comprehensive high availability services
- an efficient messaging engine

The combination of these attributes with effective system management yields an application-ready platform that will ensure uninterrupted service availability.

System modeling

The system modeling capability provides a mechanism to represent physical and logical resources that make up the overall system. It also defines the relationships and dependencies among such resources in a hierarchical network of managed objects. These objects have states and attributes that represent the actual and desired state of the corresponding physical resource.

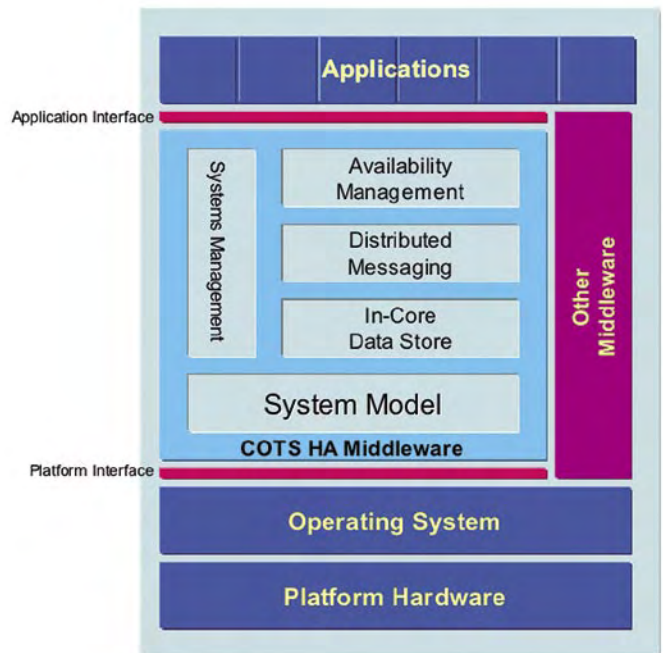


Figure 2

The systems model has the capability to group sets of like objects into service groups and then define respective recovery policies for each group that are executed in the event of a failure. Any change in the state of the objects that can affect service availability (such as a failed application, a hardware error, or planned downtime) causes an update to the systems model that in turn triggers the appropriate recovery action to maintain the overall service availability.

High availability services

High availability services manage resource failures without interrupting service. Such services are responsible for providing seamless switchover among redundant components, masking the end-user from any faults or resulting failures. High availability services must support a large number of nodes in a cluster, with the ability to collect, preserve, and distribute application state information for stateful, seamless failover between the nodes in the cluster. Key functions provided by high availability services include the following:

- The creation and maintenance of availability management information.
- The availability engine that applies policies to proactively manage the system for high availability, often 99.999% or better.
- The management of the various parts of the system, such as nodes in a cluster or redundant components.
- Checkpointing services to applications and other components.

Messaging engine

A messaging engine is designed to address the need for communication between the different elements within an application-ready platform. Such an engine provides an efficient mechanism for communicating a wide variety of information such as application state information, event and error notification, or fault management information. A messaging service provides an effective way for distributed components to efficiently communicate and coordinate their activities. Instead of requiring each resource to manage its various communication complexities, the messaging service does it for them.

A messaging service must be flexible, scalable, and reliable:

- Flexibility requires that the messaging service is independent of the applications it provides services to, so that the communication responsibilities can be offloaded from the applications.
- Scalability ensures that the messaging service is architected to support change and growth in the system components, and to support increased messaging activities. Applications do not need to be modified as the system and message volume changes or grows.
- The messaging service must be reliable to ensure message delivery, even in the event of failures in the primary network connection.

Last thoughts

With the emergence and adoption of new standards and the availability of COTS hardware and HA software, the developers of application-ready platforms that are intended for business critical applications (wireless communication for example) no longer have to develop proprietary functionality in-house to meet uninterrupted service requirements.

They can choose from field-tested and hardened middleware product suites that can easily be integrated on a variety of standards-based commercial hardware platforms and operating systems. This allows them to focus their energies and resources on developing core functionality that provides them with key competitive differentiation.

Companies can significantly reduce their costs, risks, and time to market through the establishment of high availability requirements, the use of COTS building blocks, and the use of a progressive systems development approach.

GoAhead Software

Over the past several years, GoAhead Software has helped both large and small TEMs build core functionality with their SelfReliant suite of products. This suite provides high availability, distributed messaging, and embedded systems management – the key components of highly available application ready platforms. These standards-based products have been field tested and hardened through worldwide commercial deployments. **ECD**

Dr. Asif Naseem is the Senior Vice President and CTO for GoAhead Software. He has more than 18 years of experience in the computer and communications industries. Dr. Naseem has an M.S. in Electrical Engineering and a Ph.D. in Computer Engineering from Michigan State University.



For more information, contact Dr. Naseem at:

GoAhead Software
10900 NE 8th Street • Suite 1200 • Bellevue, WA 98004
Tel: 425-468-5402 • Fax: 425-637-1117
E-mail: anaseem@goahead.com
Website: www.goahead.com