SOFTWARE CORNER

Intel focuses on high availability

By Curtis A. Schwaderer

CompactPCI Systems

The Intel Developer's Forum was held August 28th to 30th at the San Jose Convention Center in San Jose, California. This is the fourth year I've attended the Intel Developer's Forum and each year the conference has grown more focused on communications systems. This year, there was an especially good presentation by James Lawrence from Intel about high availability (HA).

Intel has spent a great deal of time with regards to HA, forming a HA Forum comprising leading board, system, and software vendors such as Intel, RadiSys, and GoAhead Software. The output of this forum is a market requirements document that addresses capabilities required for true HA networks. This document and more information about the forum can be found at http://developer.intel.com/platforms/applied/eiacomm/haforum.htm.

High availability defined

High availability is defined by the HA Forum as "the ability of a system/network to continue to deliver acceptable service regardless of hardware, software, or operator event."

To support this definition, the system must first be inherently reliable. Historically, this came in the form of extensive system test and validation. However, with the ever-increasing system complexities for communications equipment, even the most rigorous of test and validation processes are inadequate to ensure reliability under all circumstances. With this increasing complexity, focus shifts from "we'll just implement it correctly" to "what happens when an undiscovered failure actually does happen?" This gets into the second important premise of HA – fault management.

When a fault occurs, how does it affect the system? In order to answer this question, you must first look to the foundation of the software architecture – the operating system. Operating systems come in two models – threads and process. Threads model operating systems such as VxWorks, pSOS, and Nucleus sacrifice reliability for fast inter-thread communication. The threads model allows any application to access any area of memory without restriction. If a thread accidentally writes to an area of memory it isn't supposed to, the corruption can cause unpredictable results. Process model operating systems like OS-9, QNX, OSE, and Linux incorporate a memory management unit (MMU). The operating system keeps track of memory allocated to each application and if memory accesses outside the bounds occurs, process model operating systems trap, isolate, and contain illegal accesses, preventing the fault from proliferating through the system.

While process model operating systems provide important boundary/memory access protection and exception handling, this alone doesn't constitute HA. Hot swapping of system components while the system is on-line and in-use is also important. CompactPCI is especially attractive from the hardware point of view, because it allows for hot plug of hardware. However, if the software doesn't allow for the dynamic upgrade of drivers for those boards or other software components, the hardware capability alone has very limited usefulness. The OS-9 operating system has hot-swap software capability inherent within the operating system. OS-9 is a component based operating system that allows system or application components to be dynamically added, removed, or replaced while in operation. In addition to this inherent capability, HA middleware must also be in place to provide fault isolation and reporting, redundancy, load sharing, and hot-swap services control for the system. Companies like GoAhead and Jungo provide these kinds of HA middleware components.

Continuing up the software stack, applications should also be written with some level of HA in mind. This kind of mind set may be to write applications correctly in the event where the application would be terminated then restarted to take care of the exception condition. More extensively, perhaps a service set of APIs could be provided to allow for the application to query fault conditions and take action to resolve faults in more detailed processing.

Current state of high availability

One of the main reasons the HA Forum came to being is to address interoperability issues. Maybe the biggest impact moment of James Lawrence's presentation at the Intel Developer's Forum was the story he told about talking to system providers about HA. The story is summarized in Figure 1. HA has made significant

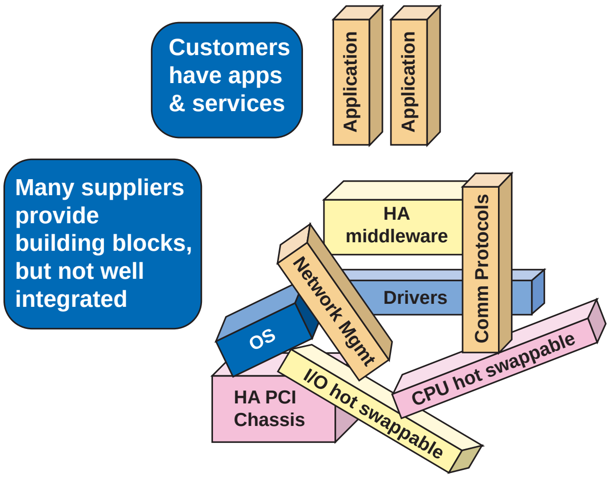

Figure 1. Incompatible HA pieces cause nightmares for system integrators

advancements over the years in the amount of hardware and software that is available to support HA. CompactPCI is a big step forward. HA middleware, drivers, network management, and protocol stacks are all available as well. The problem comes when it's time to integrate all the components. The main problem is that there is no overall framework that ensures plug-and-play interoperability between all system hardware and software components.

Intel's vision for high availability

Intel's vision seems to be to foster an ecosystem that provides "plug-and-play" interoperability between all HA components, hardware and software. Figure 2 shows the HA architecture for integrating a complete hardware/software HA solution.

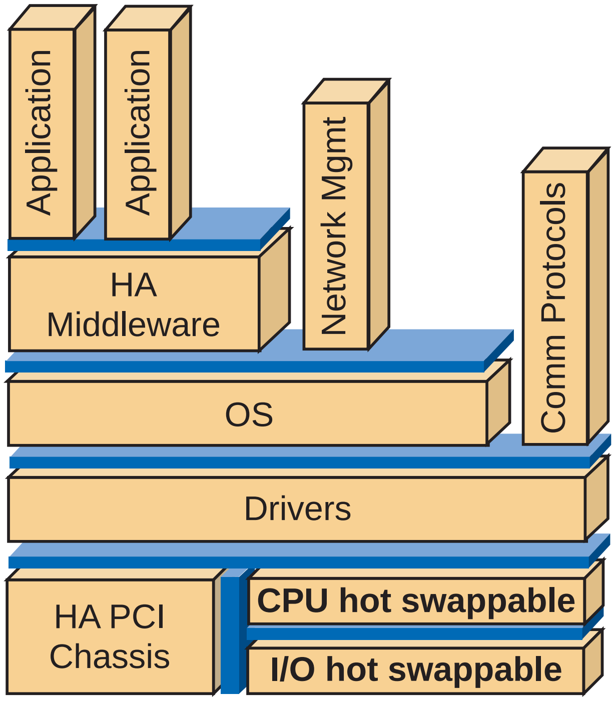

Figure 2. Intel Developer Forum diagram on HA architecture

In order to address interoperability, "HA glue" must be created. Service definitions for interfacing between cards, chassis, and software components must be defined in order for component providers to "plug-and-play" in the context of the entire HA system. Intel was not specific about steps in this direction, but obviously, this would need to come in the form of some kind of industry standard HA programming interface specification between each component in order to fully realize interoperability.

Balancing HA functions is an important part of the architecture as well. James Lawrence summarized this concept by saying, "Where practical, manage faults at the point of failure." Right now, HA middleware is depended upon to provide complete fault management when all the other components may or may not have any HA awareness. If each class of component were developed to handle common faults that occur within their realm and perhaps simply report the event to the HA middleware, it would relieve much of the complexity and bottleneck processing that now resides inside the HA middleware component.

The third main point of the presentation was the optimization of HA performance. Fault tolerance and exception handling cost time and complexity, especially if there is an HA middleware component expected to "do it all." By distributing the HA responsibility among all the components of the system, handling HA at the component level can be optimized. There may also be levels of HA that trade off performance and reliability to some extent.

Summary

It appears as though the communications industry has finally evolved to the point where serious attention is being paid to providing true HA within our communications infrastructure. With the emergence of commercial middleware packages from companies like GoAhead and Jungo, focus has shifted to providing interoperability between all hardware and software components. Intel seems to be serious about providing fully integrated system HA as evidenced by the HA Forum and the organization put in place to address HA. Lack of HA systems is an obstacle to growth for the communications industry. Current HA middleware-centric approaches are slow and very difficult to integrate. The key to advancing the state of high availability is to provide component interoperability, balance of HA processing and services, and optimization for HA components.

By the way, if you've never attended the Intel Developer's Forum before, I highly recommend it. This event may be the best kept secret in the industry for industry-leading communications software and hardware information.