Avionics Integration for CNS/ATM

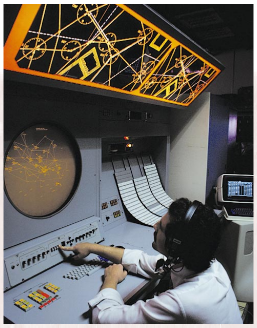
Todd Carpenter, Honeywell

Commercial aviation is changing around the globe. As the number of travelers increases, parts of the system experience severe congestion. Airports, air traffic controllers, communication networks, and the airspace itself (governed by laws stipulating minimum vertical, lateral, and head-to-tail separation of aircraft) are near their capacity during peak times.

The ramifications of this congestion are many, encompassing economics, union workload issues, safety, and performance. One prediction is that as the entire system becomes saturated around 2005, small average delays will highly perturb the system, resulting in large variances in actual arrival times. Basically, the entire air transportation system will become unpredictable.

SAFETY CONCERNS

Even if for this discussion we ignore aircraft interaction (such as increased risk due to higher pilot workloads in areas of dense traffic), we are obviously increasing our exposure to failures. While an individual passenger's risk will not necessarily increase, the number of accidents per year could rise, fueling the perception that air travel is increasingly risky.



Integration is key to a system that safely satisfies unpredictable changes throughout development and eventual deployment and operation.

Therefore, we must increase safety while alleviating the congestion and workload issues. At the same time, we must also increase economy and flexibility and minimize environmental impact such as emissions and noise. We also need to manage this change across different political and socioeconomic systems in the Americas, Europe, Asia, Australia, and Africa. This poses an enormous technical challenge.

ARCHITECTURAL IMPACTS

What ideas does the technical community have for solving the congestion problem? One proposed solution is the Communication, Navigation, Surveillance/Air Traffic Management system (see the "CNS/ATM Puts More Control in the Cockpit" sidebar). Such a system has special architectural requirements to safely satisfy unpredictable changes throughout development and eventual deployment and operation.

Minimize hardware changes

We cannot afford to rewire aircraft each time requirements change, since that would take an airplane out of service for an extended period. Even exchanging avionics boxes of equal size and with the same connections is nontrivial. Imagine the logistics—getting a new box from the depot, bringing every airplane into the maintenance base, switching the boxes, retesting the new box, and disposing of the old box (perhaps returning it for refurbishment)—for several thousand aircraft. And we also often customize avionics boxes for each model, potentially for each airplane.

Therefore, we need to minimize hardware changes. Retrofit of some airplanes can require downtime regardless of the approach, but we want to do that just once.

Move toward integration

Traditional avionics architectures are federated. That is, a unique avionics box hosts each major subsystem—navigation, display generation, and flight management, for example. Each box requires a mounting rack, power supply, interfaces, and cabling. Each will have unique maintenance procedures and require stockpiles in depots. From a stand-alone reliability perspective, it's a convenient solution: Failure of one box is unlikely to propagate to another except when they share resources such as external power. Preventing failure propagation via federation mechanisms is well understood.

However, when you consider life cycle costs, federation is expensive. Each depot needs parts, and the added weight and power of single boxes can be significant.

Integrating functionality can significantly reduce such overhead. Integration can take the form of shared computing, power, and communication resources

Editors: Jerzy W. Rozenblit, University of Arizona, ECE 320E, Tucson, AZ 85721; jr@ece.arizona.edu; and Sanjaya Kumar, Honeywell Technology Center, MS MN65-2200, 3660 Technology Dr., Minneapolis, MN 55418; skumar@htc.honeywell.com

(hardware). Sharing software libraries, tools, displays, and information (databases, for example) are also integration strategies.

Also, consider the case of CNS/ATM. What if this system should require entirely new functionality? Where does it go? Installing a new box can require extensive cabling changes.

On the other hand, should we merely jam the new functionality into an existing box? What if the box with spare capacity has nothing to do with the new function—do we want to mix communications and displays, for instance? Even worse, how do we handle different or competing suppliers? How do we resolve proprietary data issues? An integrated system with existing margin (in terms of computing, memory, and communications), and with room for future expansion (for instance, extra slots for new hardware such as radios, I/O devices, or special processors), can alleviate these growth issues.

Modular rather than monolithic

The limit of integration is a monolithic system—that is, one piece of hardware and one large software program that supports all necessary avionics functions. Although a monolithic system might be attractive from a purely operational cost in terms of size, weight, and power, easy maintainability is unlikely.

One facet of a system that affects maintainability is its ability to adapt to change and the testing each change requires. Testing sufficient to achieve Federal Aviation Authority (FAA) certification can cost tens of millions of dollars for large applications. If a single application—for instance, a pilot display and interaction menu—within a monolithic avionics suite undergoes change, it might be necessary to recertify the whole suite.

Facing such testing overhead, developers resist change, knowing that each small change could incur major costs. That would make minor increments and tests more difficult to justify and could impede safety enhancements. Therefore, while we want an integrated architecture, we also want a modular one, so that components and applications can be inserted and activated as necessary.

Modular integration isn't a particularly new concept. A typical desktop computer manages it to some extent. However, such a computer is also prone to failures and interference between applications. In avionics systems, a concept called *robust partitioning* guarantees performance and predictability of interaction.

PARTITIONING

Partitioning for safety requires at least two major components. Neither element is sufficient by itself.

Temporal partitioning guarantees that a process has temporal access to its specified resources. Temporal requirements include processing and data rates, data latencies, and processing bandwidth. Typical "real-time" operating systems or runtimes (bare bones operating systems) provide this service under some conditions, which tend to be cooperative processing without failures.

For instance, a flight control process might require five milliseconds of CPU

CNS/ATM Puts More Control in the Cockpit

A proposed solution to traffic congestion problems is to put more authority for route and flight planning in the cockpit. This strategy migrates the air traffic controller's basic role from direction to intervention and mitigation. Bear in mind this is a *proposed* solution: People in many countries at many different levels are working on it, and details have *not* been ratified or implemented.

There are, of course, many issues with this approach. This is crucial to the nature of this article. In developing a Communication, Navigation, Surveillance/Air Traffic Management (CNS/ATM) system, we seek an architecture that can integrate ensuing changes, but one that also maintains safety.

Changes to support aircraft autonomy will affect many avionics functions:

- *Communication:* Increase the affordable and safe bandwidth available to aircraft. Reduce the problems with air-to-ground (air traffic controllers and airline companies) and air-to-air communication channels, which are saturated and compete with other nonavionics users. Provide higher bandwidth, clearer, and more reliable communications.
- *Navigation:* Permit the safe reduction of separation between aircraft during all phases of flight. Permit dynamic rerouting of aircraft to avoid traffic and adverse weather and to optimize performance.
- *Surveillance:* Increase safety by enhancing onboard surveillance of the local airspace, weather, and terrain, reducing the chance of conflict.
- *Air Traffic Management:* Increase or significantly change the air traffic management system and the interaction between air traffic controllers and pilots. Reduce workload on both sides and increase the time available for safe reaction to conflict.

Groups are evaluating new approaches in each of these areas. Because various regions are exploring alternatives, avionics suppliers must support multiple variants. For instance, in Europe, the latest voice communication channel separation is 8.33 kHz; in the US, it's 25 kHz.

In navigation, countries are concerned about the Global Positioning System (GPS) serving as the sole source of information. Issues from the technical (How do you guarantee availability, especially in the presence of jamming?) to the regional (Why should the US pay to support the rest of the world's navigation? Alternatively, how can the rest of world rely on the US to maintain the system?) The surveillance protocol also differs between the US and Europe.

In addition, pilots, air traffic controllers, and maintenance personnel are all concerned about their workloads and responsibilities. One major issue is how to accommodate their current needs and maintain safety while evaluating new techniques.

time every 25 milliseconds. Once started, it must run to completion within 10 milliseconds. Each time the process activates, it must have input from various sensors, and that data must come from sensor readings taken no earlier than 20 milliseconds before the start of process activation. After each processing cycle is complete, the system must deliver various actuator commands and status to their destination within 15 milliseconds of the process activation's end.

In stricter terms, a temporally partitioned system guarantees that one application can't interfere with another's timing. Stealing processing, communications, or device time, for example, is prohibited. In the previous example, if a display process executing on the same processor overruns its deadline, robust temporal partitioning will prevent that process from taking more time at the expense of the flight control process.

Temporal partitioning puts the onus on the system to provide bounds so that developers can create their applications and know, at system-build time, whether or not the system satisfies their requirements. Otherwise, developers must integrate, then test, applications rather than rely on unit, stand-alone tests. In the flight controls example, a developer would not want to test the software for each possible combination of displays, engine controls, and/or navigation running on the same processor. The developer certainly doesn't want to test it for each change to the automatic coffeemaker control code.

There are multiple ways to provide this capability. Honeywell has built and certified systems using different approaches, including one with static, cyclic scheduling and another with rate-monotonic scheduling. Honeywell uses various approaches to check correct operation, including monitors and systems that use dual, self-checking processing and communications. Each approach has unique benefits that match different application requirements.

Spatial partitioning

Spatial partitioning guarantees that a process has unique control over its key data and state information. The strict view holds that an application cannot in any way affect another's data, except over defined interfaces such as for data sharing or communications.

For instance, a flight control process might require 500 bytes of sensor input data and generate 200 bytes of actuator and status information. When the process executes, it needs 20 Kbytes of working memory, and between activations it needs another Kbyte of state information.

In this example, you of course don't want another process to corrupt the flight control instruction data process. Neither do you want another process to modify state or working information. You also want assurance that indirect effects, such as other processes' accesses to the sensor input, cannot somehow change the data. Nor can the system permit unanticipated modification of instruction or data caches during context switches.

Partitioning allows the same system to host functionality of different criticality.

Again, there are multiple ways to provide this capability. Honeywell has certified systems incorporating processors that differentiate between user and supervisor modes, and also employ the services of memory management units that enforce spatial partitioning. Again, the checking subsystem is supported by either monitor or self-checking pairs.

All together now

Using both types of partitioning, it is impossible for a process to change its state to a supervisor (a spatial violation), and then raise its priority to steal cycles from other processes (a temporal violation).

Robust partitioning allows the same system to host functionality of different criticality (which also implies different testing levels). In the past, such cohosting would require testing all software at the level of the application with the highest criticality, incurring significant additional cost. This also lets us change any function without retesting the temporal/spatial behavior of other applications.

Since we maintain partitioning between various system builds, we can also tell at system build (roughly equivalent to "compile") whether or not the system satisfies each function's complete requirements. In addition, when we do reach the limit of some resource (for instance, a high-priority, high-criticality application just won't fit), we can negotiate with other applications on the same resource and control the change's collateral effects.

Robust partitioning has let Honeywell integrate diverse applications into a single system. One example is the Boeing 777, in which functions such as flight management, display, and various communications and data recording applications all reside in a single cabinet. A second cabinet acts as a live spare.

Each cabinet has a single data backplane that provides communication to the various processing and communications elements in the cabinet. Each processing element in the cabinet hosts multiple functions of varying criticality. This approach allowed us to cut significant size, weight, and power from the airplane, including external cabling. At the same time, we can mix components of varying criticality and make changes without retesting applications unaffected by the change.

This integrated, modular system with robust temporal and spatial partitioning provides the ability to host multiple, diverse applications in a convenient environment. Such a system allows for change and expansion without incurring the overhead of federation and avoiding the costs of systemwide recertification for every change. This means that new CNS/ATM functionality can be added or modified as time goes on without rewiring or new hardware—all you need is a software load to upgrade functionality. Such a strategy works equally well for experimentation on test aircraft and eventual deployment and service. These systems are already operational on aircraft like the Boeing 777 and 737. ❖

Todd Carpenter is a research scientist at Honeywell. Contact him at carpent@ htc.honeywell. com.