Open Approach Enables Cost Reduction for Naval Radar Displays

By using general purpose hardware and a flexible open software structure that is open to change, systems integrators can develop advanced, flexible real-time radar displays while reducing lifetime costs.

David Johnson, Technical Director
Cambridge Pixel

A modern military command and control display system typically combines the graphical elements of a user interface with the need for complex mapping and real-time sensor display such as radar video (Figure 1). A system integrator needs to consider a broad range of functional, environmental and performance criteria when designing a naval radar display system. An effective system architecture is one that loosely connects well-defined software interfaces and is based on standardized hardware components that are available from multiple vendors. By exploiting the computing potential of the CPU/GPU combination, special purpose hardware can be minimized and, to the dismay of the hardware vendor, the system integrator has choices for the supply of processing hardware, thereby reducing costs.

However, as support and maintenance account for the majority of lifetime costs in military command and control displays, it's vital that the system integrator avoids getting "locked-in" to a proprietary architecture, as this will mean that future enhancements are vendor-specific, and therefore expensive. As always in engineering design, the goal is the simplest solution that meets the requirement. But it's good practice to think of the "requirement" as not just the functional, performance and environmental specifications, but also the ability to respond to changing needs in a cost-effective and timely way. It's what good design is about, but it's also hard to measure and test.

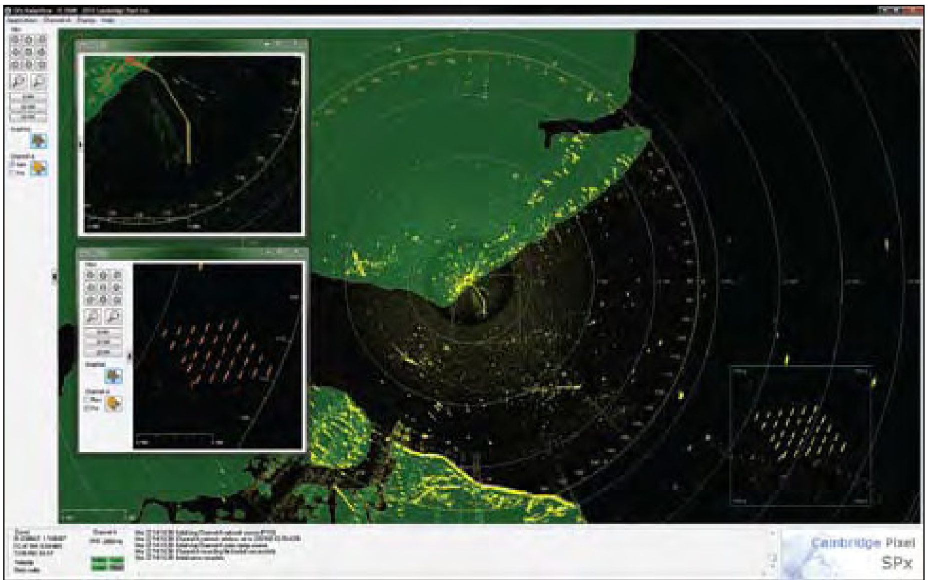

Figure 1

Radar imagery is combined with map and overlay graphics to provide a situational awareness display.

One approach to the provision of solutions for naval radar display is to provide cost-effective and enhanced capability using general-purpose hardware and a flexible software structure that is open to change. This has the combined benefit

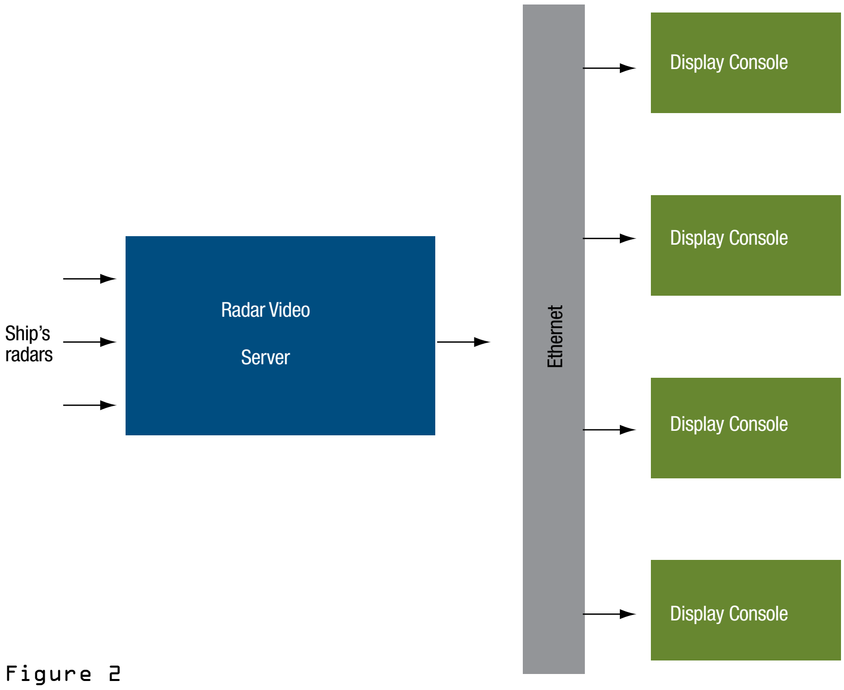

Figure 2

This naval radar display system comprises server application and multiple client consoles connected through Ethernet networks and switches.

of reducing initial procurement costs and enabling the system integrator to support, expand and modify the solution across the project lifetime.

From Sensor to Display

For example, the modern implementation of a naval radar display system has a centralized server that receives video from one or more sensors on the ship (Figure 2). These sensors may provide analog or digital signals that are captured by signal acquisition hardware in the server, or alternatively the server may receive network data direct from the sensor.

The sensor information is processed in the server, which combines multiple data sets into a common format, with standardized format, timing and network structure. The data can then be compressed for distribution across Ethernet networks to multiple client displays. On the client, the display scan converts the radar video and combines with maps and other tactical data sources.

However, the trend is to reduce the complexity and hence the cost of the client console by moving to "thin clients" that offer the benefit of being smaller, cheaper and more adaptable. Advances in processor (CPU) and graphics (GPU) capabilities have enabled the steady reduction in specialized hardware, so that sensor data (radar, sonar, video) can be received into a general-purpose computing platform and processed and displayed in a mission-dependent way. The same hardware platform can now fulfill a range of applications, and can easily be reconfigured to a different operational role, with the connected network providing access to the sensor and tactical data as needed.

With a common hardware display platform there are fewer variants and hence the greatest potential for competitive supply and reduced prices. It is no surprise that ruggedized Linux and Windows PCs now appear as workhorse consoles in many worldwide naval programs. They are cheap to deploy, maintain and eventually replace. So with a simplified hardware platform using modern multicore CPU and graphics processing engines, the emphasis is moved to the software architecture. The desired goal remains to receive sensor data and present a complex multilayered real-time display.

Software Architectures

A hardware processing architecture based around industry-standard buses, backplanes and connections provides well-defined interfaces between functional components. It's common practice to buy a single-board computer from one vendor and install a graphics card from another. Software though can be a very different situation. At the application level, one can be confident that a program built for Windows or Linux will work, but at lower levels, the ability to build a system solution using "functional modules" is less dependable and the silver bullet of software reuse is hard to realize in practice.

A key factor in providing software for any advanced requirement is to recognize that the solution must evolve. This arises because the requirements change—perhaps they were never fully defined, or perhaps the final customer needs something new, or perhaps the platform changes. Changes will occur. Good software design is recognized not by its ability to meet a functional or performance requirement, which can, eventually at least, be met by poorly written code, but rather by its ability to respond to requirement change.

Sometimes the nature of change can be predicted—it's possible to build in the capability to adapt parameters of a process through configuration rather than making coding changes. Good software design is about forecasting changes. As changes to the requirements of a system occur, well-designed system architecture will remain robust. This will ensure that changes will be cost-effective, since they have only a local effect, which can be more easily verified through regression testing. A poorly designed system cannot easily accommodate changes, which are only achieved at the price of coding complexity and substantial verification testing.

Cost and Quality

For a complex software application that needs to evolve and meet changing requirements, the cost of support will be defined by the quality of the design. The engineering process of software development will quantify performance against requirements, but measuring the quality of the design, and hence the resilience to change as the principal component of the lifetime cost, is not so easily handled with simple metrics.

Software that can be successfully maintained, enhanced and adapted needs to be structured into a collection of modules that encapsulate logical grouping of capabilities. These modules need clearly defined interfaces, allowing for replacement or upgrade, akin to hardware modules. The limited board space on a computer interface card forces a board-level structure and the need for interfacing. Software design isn't forced by necessity, but good design demands a similar organization of components with loose-coupled interfaces. These principles of good software design have been long promoted, and they remain the key criteria of a successful solution that can be supported in software that runs the real world.

Two Key Factors

To achieve the much sought lifetime cost benefits for naval displays, two conditions must be met. First, the system is built from industry-standard hardware components available from multiple suppliers, minimizing the use of vendor-specific features and maximizing the use of software for sensor processing using CPU/GPU processing. This approach reduces the initial cost, reduces the number of system variants and reduces the dependency on a specific vendor.

Secondly, the software must be modularized, open and extensible. This partitioned approach allows modules to be upgraded or replaced in isolation. It allows for an application to be built from a combination of modules that may be provided by a specialist company, or may be custom written. Since each module has a well-defined interface, each module can be upgraded in isolation.

Cambridge Pixel's SPx Software library is an example of an open, extensible toolkit approach to the receipt, processing and display of radar video for military command and control. It is architectured to run on standard processing platforms (multicore CPUs and standard GPUs) thereby giving the vendor maximum choice for the hardware. The modular software can be configured by connecting functional units to form processing chains from sensor acquisition through to display. In the case of a server-client configuration, the server is built by combining acquisition, processing, compression and distribution components.

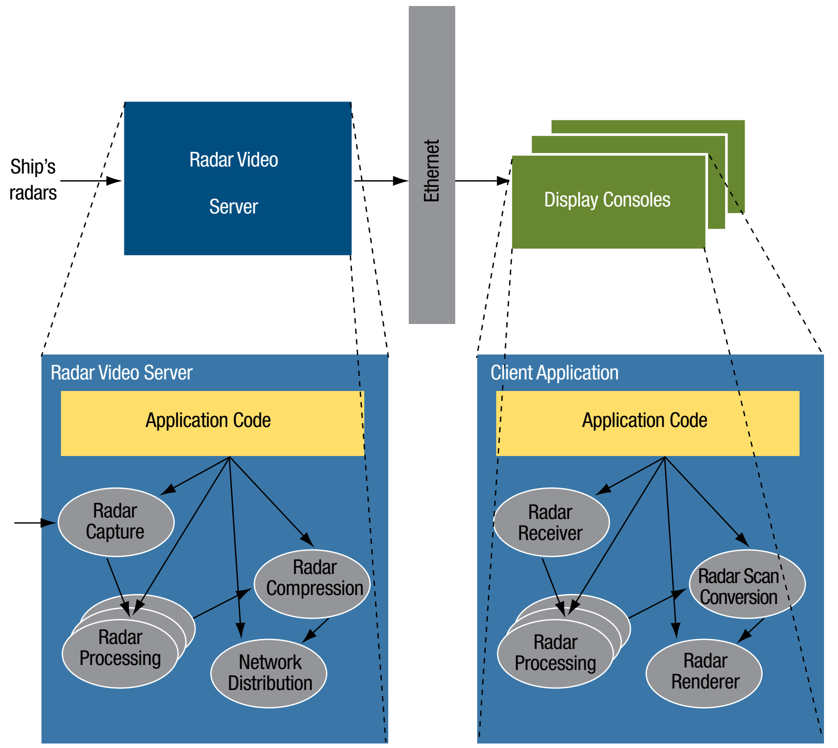

Figure 3

Modular software running in a server and client application improves ease-of-maintenance for radar distribution and display solution.

Each of these modules may be maintained and upgraded in isolation, ensuring the solution can be evolved as requirements change. For a client, the solution uses a network receive module, decompression and software-based radar scan conversion.

Radar and Tactical Imagery

Significantly, the software scan converter is able to combine the radar image with the tactical imagery from the application, allowing a weak-coupling and hence easier maintenance of the graphics and radar components. The solution works using Windows graphics (Direct-X, GDI, GDI+) or with Linux (X Windows, GTK etc.) and is able to exploit multiple processor cores to run operations in parallel. Capabilities of the graphics processor (GPU) are deployed to allow the radar image to be blended with application graphics.

The use of general-purpose hardware processing (x86/AMD multicore processing) and graphics processors (AMD or Nvidia) ensures that the solution is portable and easily adaptable to hardware from multiple vendors. There are no vendor-specific features used by the solution, so the choice of hardware ensures the most competition and best economy in the choice of hardware.

The modular software allows the development of server and client applications that build on the functional modules in the library, while permitting custom software to handle project-specific requirements. The proprietary software modules with their well-defined interfaces may be individually upgraded, enhanced (for example using established object-oriented principles of sub-classing to adapt capabilities without fundamen-

Figure 4

SPx radar processing software has been used in the command and control client software for deployment on the British Royal Navy's Type 45 destroyers.
(Photo courtesy of BAE Systems)

tal changes) or even replaced.

A system integrator can use this approach to build a complex radar server or client application, gaining the benefits of using cost-effective hardware, while retaining design control of the application and the capability for first-line support, local customization and long-term enhancement. As the requirements of the application evolve, as they inevitably will, the framework of software will be available to support variants of the solution. The cost of software change is therefore reduced from an implementation based on a tightly coupled software structure or one based on proprietary hardware.

Application Study

For example, BAE Systems Mission Systems, in New Malden and Portsmouth, England, has integrated Cambridge Pixel's software-based SPx radar processing software into its command and control client software (Figure 3) for deployment on the British Royal Navy's Type 45 destroyers (Figure 4) and the Queen Elizabeth Class aircraft carriers. This solution allows radar video to be received from multiple radars on board the ships into a server application and then distributed over Ethernet networks to command and control displays across the ship.

The server application is built from Cambridge Pixel's HPx-100 radar acquisition cards and modular SPx software for compression and network interfacing. The application software in the server remains the responsibility of BAE Systems. The new client-side software-based radar video rendering provides enhanced flexibility and capability at reduced cost over previous generation hardware rendering solutions.

BAE Systems Mission Systems opted for Cambridge Pixel's solution for these programs because of the advanced software solution and flexible product architecture. Also, BAE Systems software engineers were able to work with Cambridge Pixel to integrate the SPx capabilities into its own server and client software solution. ▮

Cambridge Pixel
Royston, UK
+44 (0) 1763 852749.
[www.cambridgepixel.com].