

# The advent of COTS middleware use in embedded systems

By Andrew Foster

**A**s embedded systems become increasingly more complex and heterogeneous, OEMs and system integrators face escalating technical, organizational, and cost challenges. Embedded system engineers must tackle the daunting task of designing and building complex hardware that is multifunctional and highly compatible, yet as cost sensitive as possible. The good news is that COTS middleware can simplify embedded systems development by making system integration far easier, improving application portability, and increasing software reuse.

## An attractive solution

Thanks to recent middleware developments including technology advancements, standardization initiatives, and new innovative approaches to building embedded systems, OEMs and system integrators no longer need to think in terms of trade-offs when developing their systems. With COTS middleware, assuming functionality will come at a higher price or that a highly compatible system will severely limit system resources is no longer necessary.

The embedded community has been following a path to a COTS middleware solution for several years. In sectors such as defense and aerospace, system requirements are increasingly complex, dynamic, and demanding as far as scalability (vast numbers of embedded sensors in a network) and quality of service is concerned. Today's solutions are often highly proprietary, brittle and nonadaptive, expensive to build and maintain, and extremely vulnerable to changes in the underlying platform and hardware technologies. Other sectors such as consumer electronics are encountering the same types of challenges fueled by the growing complexity of electronic devices including expanded features, networking, and multimedia capabilities, which are driving the requirement to add full commercial operating systems in many cases.

Building increasingly complex and diverse systems causes a number of obvious side effects. Initial build and integration costs are higher. In the automotive sector for example, it is common for the software development process to be highly distributed between different OEMs and for each OEM to have considerable freedom as to how they realize the required functionality.

An OEM typically has only a black box specification of how the different parts of the system should be connected, making integration and error detection difficult. In contrast, business applications are developed with usually far greater constraints on the software technologies used to facilitate a much smoother integration process. Compared to desktop-based applications, the goal of software reuse in embedded systems has been more difficult to achieve. In many cases applications are completely rewritten whenever changes to the underlying platform are made, creating a very expensive system upgrade process. As systems become more complex to build in the first place, this approach becomes less and less tenable, and a COTS middleware solution becomes more and more attractive.

Middleware technologies such as Java 2 Platform Enterprise Edition (J2EE) and Common Object Request Broker Architecture

(CORBA) have achieved wide acceptance across market sectors for building enterprise scale systems. Middleware provides a much higher level of abstraction for software engineers to write applications against. Benefits of using middleware to build software applications include:

- Simplifying component integration using different platforms, thus facilitating seamless interworking between heterogeneous systems
- Simplifying functionality distribution, independent of physical location or business domain
- Allowing a system to scale as usage requires and facilitating load balancing across components
- Enabling distributed component management
- Simplifying fault-tolerant and secure operation assurance

### Addressing middleware hang-ups

In spite of these major benefits, middleware technologies such as CORBA have been slow to gain wide acceptance within the embedded community. This sluggish take-up has mainly been due to concerns about the overhead using middleware imposes on an embedded application, particularly in terms of size (memory footprint), performance (latency, throughput), and predictability (timeliness). Now, however, new middleware developments including technology advancements, standardization initiatives for embedded systems such as Object Management Group's (OMG's) CORBA/e standard, and innovative approaches to building embedded systems are making advanced middleware use viable in even the most resource-constrained environments.

### Technology advancements

Middleware vendors such as PrismTech are producing extremely lightweight, highly optimized CORBA implementations that can be used efficiently in just about any embedded environment. For example, in the software-defined radio domain it is quite common to find small form factor radios consisting of a multiprocessor environment, which includes General Purpose Processor (GPP), DSP, and FPGA processor sets as part of the signal processing chain. Until recently, however, middleware would only be used to host parts of the application running on the GPP.

Due to concerns about footprint and performance as well as a lack of COTS middleware support, communication with any parts of the application hosted on the DSP and FPGA processors was previously based on proprietary communication drivers and custom message formats. Obviously, if the underlying hardware changed, large parts of the application would have to be rewritten. PrismTech's approach, for example, has been to develop highly optimized yet standards-conformant middleware implementations that can support GPP, DSP, and FPGA environments. Other COTS vendors are also producing implementations that can support these environments.

### Standardization initiatives

Another key facet of this next-generation embedded middleware is that through a combination of support for standard interfaces defined by OMG and product developments it is now very easy to support highly specialized transports (other than TCP/IP), such as those available in the form of high-speed bus-based interconnects. This allows the middleware to be used efficiently while maintaining excellent latency and throughput characteristics throughout the system.

The result of these advances is that it is now possible for embedded developers to utilize a highly efficient, standards-based, unified middleware layering across all processors in the system, simplifying component integration within the system, significantly improving application portability, and facilitating greater software reuse, as illustrated in Figure 1.

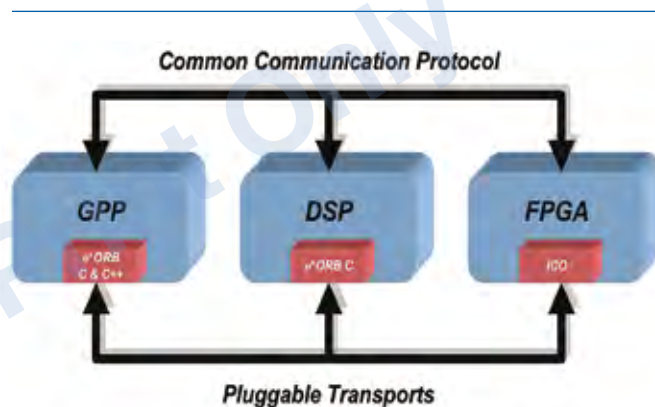


Figure 1

### Innovative development approaches

COTS middleware vendors are also combining Model Driven Development (MDD) techniques with advances in middleware for use in embedded systems to produce tool chains that support the full embedded development life cycle. By using graphical tools to model the software components in the system including the interfaces between components, it is now possible to generate a complete executable application and deploy it into a distributed embedded environment from a tool. MDD tools, with the emphasis on hiding many of the underlying complexities from the developer and automatic code generation, remove the final perceived hurdle to using middleware in embedded systems – complexity. Using these tools, any middleware complexities are completely hidden from the embedded developer, allowing more focus on developing application logic and not on infrastructure concerns.

## Lightweight CORBA

PrismTech's embedded COTS middleware product, OpenFusion e\*ORB, supports distributed CORBA

components whose footprint can take up less than 100 KB in a DSP environment, easily fitting into the onboard memory available in most modern DSPs. The company also has implemented COTS CORBA middleware in hardware, the Integrated Circuit ORB, for use in an FPGA environment.

“**Middleware vendors are producing innovative new technologies ... that are enabling middleware to be used in environments previously deemed impossible or inappropriate.**”

### **Easier integration, lower costs**

In conclusion, standards bodies such as OMG have been working hard to more adequately address the needs of the embedded development community. With broad industry support, new specifications such as OMG's CORBA/e standard are enabling vendors to produce highly optimized, standards-compliant middleware implementations that can be used successfully in the most resource-constrained embedded environments. Middleware vendors are producing innovative new technologies such as a hardware ORB for FPGAs that are enabling middleware to be used in environments previously deemed impossible or inappropriate.

Combined with modern Model Driven Development approaches to tooling, building applications using advanced embedded COTS middleware is becoming easier through automating much of the middleware code generation and allowing the embedded developer to concentrate on writing successful application logic. By leveraging the higher-level abstractions middleware provides,

embedded application portability can be improved significantly while maximizing code reuse at the same time. In the short term, complex embedded applications that use middleware are easier to build and integrate, and over the life of a system can protect the embedded OEM from expensive system rewrites whenever the hardware is changed or upgraded. **ECD**

*Andrew Foster is product manager for PrismTech's CORBA middleware technologies. He is responsible for providing both the strategic vision and managing the evolution of the highly successful range of CORBA middleware products. He has more than 12 years of experience developing Distributed Real-time and Embedded (DRE) software applications and products. He frequently presents at conferences on subjects relating to distributed middleware and embedded technologies. Andrew holds a B. Eng (honors) in Digital Systems Engineering as well as an MS in Computer-Based Plant and Process Control from the University of Sunderland in the United Kingdom.*



To learn more, contact Andrew at:

### **PrismTech Corporation**

6 Lincoln Knoll Lane, Suite 100 • Burlington, MA 01803

781-270-1177

awf@prismtech.com

www.prismtech.com