GUEST EDITORIAL

Middleware Technologies for Future Communication Networks

Gordon Blair   Andrew T. Campbell   Douglas C. Schmidt

Improvements in hardware and networking technologies over the past two decades have yielded dramatic increases in computer and communication capabilities. Despite these advances, however, the effort and cost required to develop, validate, port, and enhance software for networked applications remains remarkably high. Much of the complexity and cost of building networked applications can be alleviated by the use of highly flexible, efficient, dependable, and secure *middleware*, which is systems software that resides between the applications and the underlying operating systems and networks, and provides reusable services that can be composed, configured, and deployed to create networked applications rapidly and robustly.

Middleware represents the confluence of two key areas of information technology (IT): distributed systems and component-based design and programming. Techniques for developing distributed systems focus on integrating many computing devices to act as a coordinated computational resource. Likewise, techniques for developing component-based systems focus on reducing software complexity by capturing successful patterns and creating reusable component frameworks. Middleware is therefore the field dealing with component-based systems that can be distributed effectively over a myriad of computing devices and communication networks to provide developers of networked applications with the necessary platforms and tools to:

- Formalize and coordinate how parts of applications are composed and how they interoperate
- Monitor, enable, and validate the (re)configuration of resources to ensure appropriate application end-to-end quality of service (QoS), even in the face of failures or attacks

During the past decade IT developers and end users have benefitted from the commoditization of commercial off-the-shelf (COTS) hardware (e.g., CPUs and storage devices) and networking elements (e.g., IP routers). More recently, the maturation of programming languages (e.g., Java and C++), operating environments (e.g., POSIX and Java Virtual Machines), and middleware (e.g., CORBA, Java 2 Enterprise Edition, and SOAP/Web services) are helping to commoditize many COTS software components and architectural layers. Although the quality of COTS commodity software has often lagged behind hardware, recent improvements in software frameworks, component models, patterns, and development processes have encapsulated the knowledge that enables COTS middleware to be developed, integrated, and used successfully with fixed networks in an increasing number of real-world networked application domains, including telecom/datacom, enterprise e-commerce systems, desktop business applications, aerospace and defense systems, industrial process control, and financial services.

COTS middleware platforms have generally expected static connectivity, reliable communication channels, and relatively high bandwidth. Significant challenges remain, however, to design, optimize, and apply middleware for more flexible network environments, such as self-organizing peer-to-peer (P2P) networks, mobile settings, and highly resource-constrained sensor networks. For example, hiding network topologies and other deployment details from networked applications becomes harder (and often undesirable) in wireless sensor networks since applications and middleware often need to adapt according to changes in location, connectivity, bandwidth, and battery power. Concerted R&D efforts are therefore essential to devise new middleware solutions and capabilities that can fulfill the requirements of these emerging network technologies and next-generation applications.

The articles in this Special Issue of *IEEE Network* on Middleware Technologies for Future Communication Networks describe the latest results of cutting-edge middleware R&D efforts that focus on the next generation of mobile and fixed networks and services. For instance, the emergence of programmable sensor networks has yielded a new class of applications (e.g., equipment and process control, environmental monitoring, emergency management, sniper localization, active vibration damping, and smart structures) in which a wide range of coordinated devices and services can be assembled flexibly and rapidly to achieve critical missions. Today's sensor network applications are developed manually in an ad hoc and informal manner, however, which is costly, tedious, and error-prone, and often fails to achieve the desired functionality and quality under tight resource and reliability constraints. This special issue therefore contains a pair of articles — "Middleware to Support Sensor Network Applications" by Heinzelman, Murphy, Carvalho, and Perillo, and "Issues in Designing Middleware for Wireless Sensor Networks" by Yu, Krishnamachari, and Prasanna —

that classify the different types of sensor network applications and then discuss middleware principles, policies, and mechanisms to enhance the next generation of these applications.

There has also been substantial growth recently in the development and use of P2P systems, in which all nodes have similar responsibilities and communication is symmetric. Successful P2P applications include content sharing (e.g., Napster, Gnutella, and Freenet) and large-scale storage systems. Peer-to-peer computing offers several advantages over earlier networked application architectures, including self-organization, availability, automatic load balancing, and scalability. Part of the success of these systems comes from their ability to harness idle storage and network resources, often offered by computers volunteered by their owners to participate in the system. This special issue contains three articles that cover key middleware topics associated with P2P systems. "Managing Distributed Objects in Peer-to-Peer Systems" by Kalogeraki and Chen focuses on effectively managing distributed objects in P2P systems to improve their end-to-end QoS. "A Methodology for the Design of Distributed Search in P2P Middleware" by Mischke and Stiller presents a survey of search techniques for finding and retrieving networked resources in large-scale P2P systems. "A Self-Organizing Publish/Subscribe Middleware for Dynamic Peer-to-Peer Networks" by Junginger and Lee describes group communication mechanisms that are tailored to meet the challenges of decentralized P2P systems.

Middleware generally supports two models for component interaction:

- A request-response communication model, in which a component invokes a point-to-point operation on another component
- An event-based communication model, in which a component transmits arbitrarily defined messages, called *events*, to other components

Event-based middleware is becoming popular for large-scale heterogeneous distributed systems because it helps reduce software dependencies, and enhance system composability and evolution. The article "Composite Event Detection and Generic Middleware Extension" by Pietzuch, Shand, and Bacon describes extensions to existing event-based middleware, such as the Java Messaging Service (JMS), that enables applications to subscribe to complex patterns of events, thereby reducing bandwidth consumption while maintaining low event notification latency.

The next generation of large-scale networked applications (e.g., streaming video, Internet telephony, and interactive simulation systems) have increasingly stringent QoS requirements. In today's horizontally integrated and commoditized IT environment, these types of applications are being developed using multiple layers of hardware, operating systems, and middleware components. Historically, it has been hard to configure such large-scale networked applications to simultaneously satisfy multiple QoS properties, such as security, timeliness, and reliability. Moreover, in the face of changing requirements and environments, there is a need for a more flexible software infrastructure that can be (re)configured dynamically to react to changing contexts. The article "Reflective Middleware for Integrating Network Monitoring with Adaptive Object Messaging" by Han, Gutierrez-Nolasco, and Venkatasubramanian describes how reflective middleware can be developed and used to adaptively configure, monitor, and control networked applications that possess a range of interdependent QoS properties.

As the articles in this special issue of *IEEE Network* attest, recent advances from the R&D community are enabling middleware to be applied successfully to meet the needs of next-generation networked applications. Although a great deal of hoopla on these topics has appeared in the commercial software industry and trade press, it has been surprisingly hard to find solid technical material on the strengths and weaknesses of middleware for these exciting new applications. The articles in this special issue are intended to help replace the hype with solid technical results. We encourage you to get involved with others working on these topics and to contribute your insights and experience in future middleware conferences, journals, and other publication venues.

Biographies

GORDON BLAIR (gordon@comp.lancs.ac.uk) is a member of the Distributed Multimedia Research Group at Lancaster University. Following the completion of his Ph.D. at Strathclyde University, he moved to Lancaster in 1983. He currently holds a chair in distributed systems at this university, and is also an adjunct professor at the University of Tromsø in Norway. He has published over 180 papers in his field and has served on the program committees of many major international conferences in distributed systems and multimedia. He is on the steering committee of the Middleware series of conferences and was General Chair for this conference in 1998. He is also Program Co-Chair for this incarnation of DOA. His research interests are in the areas of distributed systems architectures, reflective middleware, and the application of distributed systems platforms to areas such as mobility and multimedia. He co-organized a workshop on reflective middleware (with Roy Campbell, University of Illinois) held in conjunction with Middleware 2000, and has been primarily responsible for a number of research projects at Lancaster, including Sumo (with France Telecom), Adapt, and Open ORB.

ANDREW T. CAMPBELL (campbell@ee.columbia.edu) is an associate professor of electrical engineering at Columbia University and a member of the COMET Group. He is working on emerging architectures and programmability for wireless networks. He received his Ph.D. in computer science in 1996, and the NSF CAREER Award for his research in programmable mobile networking in 1999. Currently, he is on sabbatical as a UK EPSRC Visiting Fellow at the Computer Laboratory, Cambridge University.

DOUGLAS C. SCHMIDT (d.schmidt@vanderbilt.edu) is a professor in the Electrical Engineering and Computer Science Department at Vanderbilt University. He has published over 250 technical papers and books that cover a range of research topics, including patterns, optimization techniques, and empirical analyses of software frameworks that facilitate the development of distributed real-time and embedded (DRE) middleware running over high-speed networks and embedded system interconnects. He has served as a deputy office director and program manager at DARPA, where he led the national R&D effort on DRE middleware. He has also served as co-chair for the Software Design and Productivity Coordinating Group of the U.S. government's multi-agency Information Technology Research and Development Program, which formulated the multi-agency research agenda in software design. In addition to his academic research and government service, he has over 15 years of experience leading the development of ACE and TAO, which are widely used open source DRE middleware frameworks that contain a rich set of components that implement patterns for high-performance DRE systems.