Dynamic, High Confidence Certifiable Embedded Software: Position Paper

Kurt Rohloff, Richard Schantz, Joseph Loyall
{krohloff, schantz, jloyall}@bbn.com
BBN Technologies

Exhaustive testing, documentation, code review, and formal methods have been the main approaches for software certification in high confidence cyber-physical systems. Although these methods have been appropriate and sufficient in the past, the continued reliance on these methods are no longer economically feasible for increasingly complex modern, distributed, dynamic systems due to inherent problems of state-explosions. We need to develop innovative, economically feasible means to certify distributed dynamic control software for cyber-physical systems so that when these systems are deployed, harmful and potentially unpredictable emergent control behavior does not manifest itself.

There are many hurdles for even minimal acceptance of dynamic behavior embedded in distributed mission critical cyber-physical systems. Examples of such distributed mission critical systems include DoD systems (including secure, timely command, control and information sharing systems and for military logistics and transportation infrastructures), systems for manufacturing and process control (for industries whose safety is of critical national importance such as transportation, chemical, oil and natural gas), and medical systems among others.

As the use of advanced, distributed adaptive system architectures become mainstream for these applications that include multi-layer QoS, peer-to-peer behavior and ad-hoc distributed interaction strategies for reconfigurable topologies, entire new classes of certification strategies will need to be developed and implemented. These certification strategies need to handle system-wide, multi time-scale event structures with time-critical operations, the aggregation of synchronous and asynchronous computation systems, and address multi-platform, distributed resource management issues. Furthermore, for the pervasive, wide-scale architectures listed above, life-cycle issues need to be addressed so that beyond initial certification, the systems need to be easily and inexpensively recertifiable so that they can be modifiable as system requirements evolve over very large time scales (possibly years or decades) without incurring large additional costs.

Elements of architecture, design, algorithms, analysis, simulation, testing and instrumentation/logging would all contribute to increasing the acceptance and the certifiability of distributed, real-time high-confidence software for cyber-physical systems. To have maximum impact, future work on certification in this area must intelligently link augmentations of each of the elements together into one unified methodology capable of being easily adopted by governmental and industrial regulation agencies large and small.

Today, there are no meaningful, scalable techniques to certify a whole, composed network centric system, even when it is composed of certified parts. This is due, in large part, to the fact that two composed elements, even when their independent behaviors are certified, might functionally interact in ways that cause one or both of them to exhibit unpredictable emergent behavior that violates their “certified” behaviors. A simple example is that of two high priority real-time components, each of which independently are shown to meet their real-time deadlines in isolation. When composed, they utilize shared computation and network resources needed by the other, in such a way that they could affect each other's ability to meet its deadline.

Interface standards, such as those for objects and components, impose some (minimal) rigor on the functional interactions between elements. As long as system developers limit their development of independent elements to components or objects with well defined interfaces, and compose the system only through those interfaces, then some properties of the component interaction can be identified. These functional interfaces are insufficient for reasoning-based certification of our dynamic, distributed systems because components also interact through shared access to resources. Combined with dynamic resource management, these interactions are most often quite complex, difficult to completely and adequately specify, and even more difficult to certify.

Over the next 5 to 10 years, an approach to certifiable composition is to create well-defined QoS, or resource, interfaces through a middleware system and program only to within the strict limits imposed by them, analogous to using functional interfaces. Such an interface would identify the resources used by a component (the obvious ones would be CPU and network, but other interfaces such as a shared displays, memory footprints, etc. could also be considered) and identify the change in expectations as well as the behaviors of the component under various ranges of resource availability.

As a path towards the certifiability and acceptance of dynamic, distributed, real-time embedded systems, one needs to be able to separate "good" dynamic behavior from "bad" dynamic behavior from a certifiability point of view. Dynamic behavior is not always appropriate and there may be times when otherwise appropriate dynamic behavior is inappropriate. The certification of dynamic systems will need to establish the terms of evaluation for dynamic operation, and will need to establish that both "good" behaviors happen, while "bad" behaviors don't, to some degree of plausibility.

Provisioning the resource management functionality as middleware infrastructure enables us to better and more certifiably *control* the resources provided through these interfaces. In the beginning, we can use clearly partitioning resource allocation mechanisms, such as CPU reservations and network reservations. These enable the dynamic resource manager to provide a clear set of resources to each component (and component interaction), simplifying the reasoning and certification. A next step, which introduces more complexity to the interfaces and analysis, would look at priority based resource provisioning. Even there, priority lanes and admission control can help bound the problem space.

A more general problem involves removing these constraint mechanism interfaces in favor of policy-driven application control which would automatically regulate component interaction. A promising idea in this space is to use specialized, domain-specific, mutually composable sequential process languages such as a duration calculus to express the certified behaviors of individual components of the system. The union of shared expressions in these domain-specific sequential process languages would need to be expressive enough to specify the behavior of the overall system (with respect to both functional behavior and resource usage), but should be sufficiently easy to compose and perform computable operations on to certify the behavior of the overall system. The languages could be used as control specifications to regulate the behaviors of the local components with respect to these desired behaviors.

Author Bios

Kurt Rohloff
krohloff@bbn.com
Intelligent Distributed Computing
BBN Technologies
617-873-7664

Kurt Rohloff received the bachelor of electrical engineering degree with highest honors from the Georgia Institute of Technology and the master of science and PhD. degrees in electrical engineering: systems from The University of Michigan, Ann Arbor. Dr. Rohloff was a visiting researcher at the Center for Mathematics and Computation (CWI) in Amsterdam, the Netherlands, and a post-doctoral researcher at the Coordinated Science Laboratory (CSL) at the University of Illinois: Urbana-Champaign. Dr. Rohloff is currently a scientist at BBN Technologies. His research interests include discrete-event systems, resource allocation, distributed high-confidence software and certification.

Richard Schantz
schantz@bbn.com
Intelligent Distributed Computing
BBN Technologies
617-873-3550

Richard E. Schantz is a principal scientist at BBN Technologies in Cambridge, Mass., where he has been a key contributor to advanced distributed computing R&D for the past 33 years. His research has been instrumental in defining and evolving the concepts underlying middleware and distributed object middleware since its emergence in the early days of the Arpanet and Internet. Most recently, he has led research efforts toward developing and demonstrating the effectiveness of dynamic middleware support for adaptively managing realtime end-to-end Quality of Service and system survivability. Schantz received his Ph. D. degree in Computer Science from the State University of New York at Stony Brook, in 1974, and an undergraduate degree in Mathematics from NYU in 1968. He is a Fellow of the ACM.

Joseph Loyall
jloyall@bbn.com
Intelligent Distributed Computing
BBN Technologies
617-873-4679

Joseph Loyall received his B.S. degree in computer science from Indiana University and his M.S. and Ph.D. in computer science from the University of Illinois. Dr. Loyall is a Division Scientist and one of the leads of the Distributed Systems Technology Group at BBN Technologies. He has been the PI for several research programs in the areas of quality of service, resource management, real-time systems, adaptive middleware, embedded systems, software engineering, and fault tolerance. Dr. Loyall is a senior member of the IEEE and a member of the ACM and the AIAA. He holds three patents and is the author of over 50 publications, including book chapters, journal articles, conference papers, and workshop papers.