Preface

Networked Systems: Realization of Reliable Systems on Unreliable Networked Platforms

The Monterey Workshops series was initiated in 1993 by David Hislop with the purpose of exploring the critical problems associated with cost-effective development of high-quality software systems. During its 12-year history, the Monterey Workshops have brought together scientists that share a common interest in software development research serving practical advances in next-generation software-intensive systems. Each year is dedicated to a given topic such as "Software Engineering Tools: Compatibility and Integration" (Vienna in 2004), "Engineering for Embedded Systems: From Requirements to Implementation" (Chicago in 2003), "Radical Innovations of Software and Systems Engineering in the Future" (Venice in 2002), "Engineering Automation for Software Intensive System Integration" (Monterey in 2001), etc.

This 12th Monterey Workshop was held in Laguna Beach, CA during September 22–24, 2005.

Context of the 12th Workshop

Networked computing is increasingly becoming the universal integrator for large-scale systems. In addition, new generations of wireless networked embedded systems rapidly create new technological environments that imply complex interdependencies amongst all layers of societal-scale critical infrastructure, such as transportation, energy distribution and telecommunication. This trend makes reliability and safety of networked computing a crucial issue and a technical precondition for building software-intensive systems that are robust, fault tolerant, and highly available.

The 12th Monterey Workshop on "Networked Systems: Realization of Reliable Systems on Unreliable Networked Platforms" focused on new, promising directions in achieving high software and system reliability in networked systems.

All presentations at the workshop were by invitation upon the advice of the Program Committee.

Invited Speakers

Myla Archer Naval Research Lab, USA
Barrett Bryant University of Alabama, Birmingham, USA
David Corman Boeing, St Louis, USA
Nick Dutt UCI, USA
Holger Giese University of Paderborn, Germany
Chris Gill Washington University at St Louis, USA
Helen Gill NSF, USA
Klaus Havelund NASA, USA
Papers included in this volume were selected among the submissions from the workshop’s discussions.

Workshop Topics

Software is the new infrastructure of the information age. It is fundamental to economic success, scientific and technical research and national security. Our current ability to construct the large and complex software systems demanded for continued economic progress is inadequate.

The workshop discussed a range of challenges in networked systems that require further major advances in software and systems technology:

- **System Integration and Dynamic Adaptation.** A new challenge in networked systems is that stable application performance needs to be maintained in spite of the dynamically changing communication and computing platforms. Consequently, the run-time architecture must include active control mechanisms for adapting the
system/software components to changing conditions. Global system characteristics need to be achieved by increased run-time use of refection (systems that utilize their own models), advanced interface modeling, self-adaptation, and self-optimization.

- **Effects of Dynamic Structure.** The structure of networked systems is complex and highly dynamic. Because systems are formed by ad hoc networks of nodes and connections, they lack ne-grain determinism for end-to-end behaviors that span subsystem and network boundaries. In addition, there are end-to-end system qualities such as timeliness and security that can only be evaluated in this dynamically integrated context.

- **Effects of Faults.** Faults and disruptions in the underlying communication and computing infrastructure are the normal events. Since well-understood techniques for fault-tolerant computing, such as n-modular redundancy, are not applicable in the dynamically changing networked architecture, new technology is required for building safe and reliable applications on dynamic, distributed platforms.

- **Design for Reliability.** Although there are varieties of metrics and established practices for characterizing the expected failure behavior of a system after it is elded and there are established practices for specifying the desired reliability of a system, the evaluation of system or software reliability prior to elding is a significant problem.

- **System Certification.** The process for certifying that a system meets specified reliability goals under the range of conditions expected in actual use currently involves exhaustive analysis of a system, including its development history and extensive testing. Current methods do not give systems engineers the confidence they would like to have in concluding that a system will have particular reliability characteristics.

- **Effects of Scale.** Another risk that overlays all proposed solutions is scale. Scale also addresses both run-time and design-time concerns. Typically, demonstrations are the convincing drivers to technology adoption. Demonstrations of new technologies however are usually small-scale, focused efforts. It is an open problem how to scale up a demonstration that addresses the number of nodes and connections, and the number of software developers, analysts, and integrators to provide enough proof to justify technology transition.

These challenges are exaggerated in networked-embedded software systems, where computation and communication are tightly integrated with physical process.

**Approaches**

There have been important new developments during the past five years that improve our chance to meet the new challenges listed above. Contributions at the workshop identified and discussed research approaches that have direct and immediate relevance to the new challenges. Listed below are the major themes that came up in many forms in the presentations and captured in the contributions of these proceedings.
Model-based software development of network-centric system-of-systems. Model-based design is rapidly becoming one of the prominent software/system development paradigms. Models with precisely defined syntax and semantics capture system/software invariants that can be formally analyzed and used for predicting/verifying system behavior and for generating code. A new challenge in network-centric system-of-systems is that design invariants need to be maintained actively during run-time due to the dynamically changing communication and computing platforms. Consequently, the relationship between design-time modeling and model analysis and run-time behavior needs to be fundamentally different: emphasis needs to be shifted toward correct-by-construction approaches that can guarantee selected behavioral properties without the need for system-level verification, and the run-time architecture must include active control for adapting the system/software to changing conditions. Global system characteristics need to be achieved by increased run-time use of reflection (systems that utilize their own models), advanced interface modeling, self-adaptation, and self-optimization.

Foundations of future design and programming abstractions. Programming abstractions have a crucial role in the design of highly concurrent, dynamic, and time-critical networked systems. Today’s abstractions have been developed for programs with static structure, closed architectures, and stable computing platforms that are not scalable, understandable, and analyzable in complex, networked, real-time systems. We need abstractions that go beyond a narrow view of programming languages to integrate modeling, design, and analysis. They must satisfy the need for blending solid formal foundations with domain-specific expressions and must yield behavior that is predictable and understandable to system designers, even in the face of uncertain or dynamic system structure. To accomplish this, they must serve both the modeling role and the design role, leveraging generators, visual notations, formal semantics, probabilistic modeling, and yet-to-be-developed techniques for gaining an effective multiplicity of views into a design. And they must effectively express concurrency, quality-of-service constraints, and heterogeneity.

Active fault management in network-centric systems. It is important to recognize that software will never be perfect large-scale, networked systems-of-systems. Software and platform components may fail at any time. The notion of active fault management accepts this as a fact and instead of attempting to mask the faults, it focuses on their containment, mitigation, and management. Active fault management is a novel technique that is gaining acceptance in complex engineering systems (e.g., aerospace vehicles) and promises reliability through detecting, isolating and recovering from faults using algorithmic techniques for contingency management. The software engineering community took notice of these engineering techniques and applies them to software artifacts. The resulting fault management architectures are layered, as different methods may be needed on different levels of abstractions in systems and, preferably, they have to be proactive, so that they detect early precursors to larger problems (e.g., memory leak in dynamically allocated memory, or memory fragmentation) such that the system will have sufficient time to take preventive action.

Intelligent, robust middleware. Complexity of large-scale networked systems requires careful consideration on reusability of code. Middleware technologies offer architec-
tural solutions for separating application code from highly reusable components or layers in software stacks. We need to develop and validate a new generation of intelligent middleware technologies that can adapt dependably in response to dynamically changing conditions for the purpose of always utilizing the available computer and network infrastructure to the highest degree possible in support of system needs. Emerging architectures, such as service-oriented architecture (SOA), provide focus for this new generation of middleware research that will ultimately enable software whose functional and QoS-related properties can be modified either statically, (e.g., to reduce footprint, leverage capabilities that exist in specific platforms, enable functional subsetting, and minimize hardware/software infrastructure dependencies) or dynamically (e.g., to optimize system responses to changing environments or requirements, such as changing component interconnections, power-levels, CPU/network bandwidth, latency/jitter, and dependability needs).

**Model-based development of certifiable systems.** Systems that are safety certifiable are arguably some of the most costly to develop. As a result, software architectures for such systems are typically very deterministic in order to enable provable mitigation of safety hazards. The limitations of these approaches are quickly becoming unacceptable due to the advent of ad-hoc mobile networks requiring a much more dynamic structure and expected unavailability of certain resources for these safety critical systems. Model-based development approaches must be applied to enable the development of these systems within reasonable cost. These approaches should include the development of modeling syntax and semantics to express safety-critical aspects and perhaps constrain dynamism, the provision of design-time and run-time analysis that leverages this model and addresses the concerns of the safety community in the context of a network-centric system of systems, the automatic generation of artifacts that are proven by analysis to be safe, and the establishment of trust in such tools and techniques by the safety community as a whole.

**Acknowledgement**

We are grateful to the Steering Committee, the Local Organizing Committee and the invited speakers for making the workshop a success. We gratefully acknowledge sponsorship from the Army Research Office (David Hislop) and from the National Science Foundation (Helen Gill).

January 2007

Fabrice Kordon

Janos Sztipanovits
Organization

Executive Committee

Conference Chair: Kane Kim (University of California, Irvine, USA)
Program Chairs: Fabrice Kordon (Université Pierre & Marie Curie, France)
Janos Sztipanovits (Vanderbilt University, USA)

Technical Program Committee

Carlos Delgado Kloos University Carlos III of Madrid, Spain
Bertil Folliot University of Pierre & Marie Curie, Paris, France
Tom Henzinger Ecole Polytechnique Federale de Lausanne, Switzerland
Kane Kim University of California at Irvine, USA
Insup Lee University of Pennsylvania, USA
Chenyang Lu Washington University, USA
Tom Maibaum King’s College, London, UK
Ugo Montanari University of Pisa, Italy
Laurent Pautet Télécom Paris, France
Wolfgang Pree University of Salzburg, Austria
Doug Schmidt Vanderbilt University - ISIS, USA
# Table of Contents

Reinventing Computing for Real Time .............................................. 1  
   *Edward A. Lee and Yang Zhao*

Applying Service-Oriented Development to Complex Systems: BART Case Study ................................................................. 26  
   *Ingolf H. Krüger, Michael Meisinger, and Massimiliano Menarini*

Towards Dynamic Partitioning of Reactive System Behavior: A Train Controller Case Study ......................................................... 47  
   *Victor Winter and Deepak Kapur*

The GridLite DREAM: Bringing the Grid to Your Pocket .................. 70  
   *Chris A. Mattmann and Nenad Medvidovic*

DARX - A Self-healing Framework for Agents ................................. 88  
   *Olivier Marin, Valdis Berzins, and William H. Roof*

Nautical Predictive Routing Protocol (NPRP) for the Dynamic Ad-Hoc Nautical Network (DANN) ......................................................... 106  
   *Luqi, Valdis Berzins, and William H. Roof*

A Factory to Design and Build Tailorable and Verifiable Middleware 121  
   *Jérôme Hugues, Fabrice Kordon, Laurent Pautet, and Thomas Vergnaud*

A Concurrency Abstraction for Reliable Sensor Network Applications 143  
   *János Sallai, Miklós Maróti, and Ákos Lédeczi*

Outdoor Distributed Computing with Split Smart Messages .............. 161  
   *Nishkam Ravi and Liviu Iftode*

Towards a Real-Time Coordination Model for Mobile Computing ...... 184  
   *Gregory Hackmann, Christopher Gill, and Gruia-Catalin Roman*

Dynamic System Reconfiguration Via Service Composition for Dependable Computing ......................................................... 203  
   *W.T. Tsai, Weiwei Song, Yinong Chen, and Ray Paul*

A Component-Based Approach for Constructing High-Confidence Distributed Real-Time and Embedded Systems ...................... 225  
   *Shih-Hsi Liu, Barrett R. Bryant, Mikhail Auguston, Jeff Gray, Rajeev Raje, and Mihran Tuceryan*
Providing Dependable Services with Unreliable SoCs—The DECOS Approach .................................................... 248
   Hermann Kopetz

Modeling and Verification of Cooperative Self-adaptive Mechatronic Systems ........................................................ 258
   Holger Giese

Architectural Design, Behavior Modeling and Run-Time Verification of Network Embedded Systems .................................................. 281
   Man-Tak Shing and Doron Drusinsky

Approaches for Inheritance in the TMO Programming Scheme .......... 304
   K.H. (Kane) Kim, Moon-Cheol Kim, and Moon-Hae Kim

Author Index ........................................................................ 317