

# Poster Paper

## Two-Level Assurance of QoS Requirements for Distributed Real-time and Embedded Systems \*

Shih-Hsi Liu  
Barrett R. Bryant  
Jeffrey G. Gray  
Department of Computer and  
Information Sciences  
University of Alabama at  
Birmingham  
Birmingham, AL 35294, USA  
{liush, bryant, gray}  
@cis.uab.edu

Rajeev Rajeev  
Andrew Olson  
Department of Computer and  
Information Science  
Indiana University Purdue  
University Indianapolis  
Indianapolis, IN 46202, USA  
{rraje, aolson}  
@cs.iupui.edu

Mikhail Auguston  
Department of Computer  
Science  
Naval Postgraduate School  
Monterey, CA 93943, USA  
maugusto@nps.navy.mil

### ABSTRACT

Assuring quality of service (QoS) requirements is critical when assembling a distributed real-time and embedded (DRE) system from a repository of existing software and hardware components. This paper presents a two-level approach for assuring satisfaction of QoS requirements in the context of a reduced design space for DRE systems. Techniques from artificial intelligence and statistics are used to fulfill these collective objectives at system assembly time. The result not only lessens the overhead of validation of QoS requirements at run-time, but also reduces the development and integration cost of DRE systems.

### Categories and Subject Descriptors

C.3 [Special-Purpose and Application-based Systems]:  
Real-time and embedded systems.

### General Terms

Algorithms, Design

### Keywords

Real-Time, Quality of Service, Domain-specific Modeling

## 1. INTRODUCTION

Distributed real-time and embedded (DRE) systems are widely used in military, manufacturing, and control systems. Many of these systems consist of legacy components, either hardware or software. Thus, there is an urgent demand to fulfill the need of the development and integration of DRE systems from existing components. During the synthesis process of a DRE system, appropriate components

\*This research was supported in part by U. S. Office of Naval Research award N00014-01-1-0746.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC '05, March 13-17, 2005, Santa Fe, New Mexico, USA.  
Copyright 2005 ACM 1-58113-964-0/05/0003 ...\$5.00.

are selected from a repository. A validation process subsequently ensures that the assembled system fulfills the requirements. In addition to functional requirements, quality of service (QoS) is an important requirement of DRE systems. UniFrame [8] is a QoS-based approach for building distributed systems from heterogeneous components. While identifying relevant QoS properties to be validated in a distributed domain, UniFrame does not currently address the problem of building DRE systems. In this paper, a two-level assurance technique for QoS of DRE systems assembled from components is presented. This technique, based on artificial intelligence and statistics, reduces the design space and validates QoS requirements at system assembly time. Consequently, we believe that this technique may ease the overhead of validation of QoS requirements at runtime, and reduce the development and integration cost of DRE systems. In addition, the usage of a modeling tool for assurance framework construction and statistical concepts for assembled cases (i.e., solutions of design space) facilitate the reusability and flexibility in different perspectives. Section 2 describes the framework of the system. Section 3 concludes our work.

## 2. FRAMEWORK

In this section, the framework of the system is described. Starting from nonfunctional requirements, a use case scenario is analyzed to determine the static and dynamic QoS requirements. Then, a Petri Net-based QoS model as an analysis and assurance toolkit is constructed. Backtracking and branch-and-bound algorithms are employed to prune off infeasible assembled cases based on static QoS requirements at the first level. A domain-specific scripting language then further discards less probable assembled cases based on previous states and observations of dynamic QoS requirements of components stored in a knowledge base. More details of each level and Petri Net model will be addressed in the following subsections. Figure 1 shows the QoS assurance framework of the system.

### 2.1 Petri Net-based QoS Modeling

The achievement of quality of services usually requires cooperation of collective components of a DRE system. Therefore, a formal approach to model and analyze the components of a DRE system with respect to its quality of services is necessary: a Petri Net-based QoS model is created in the

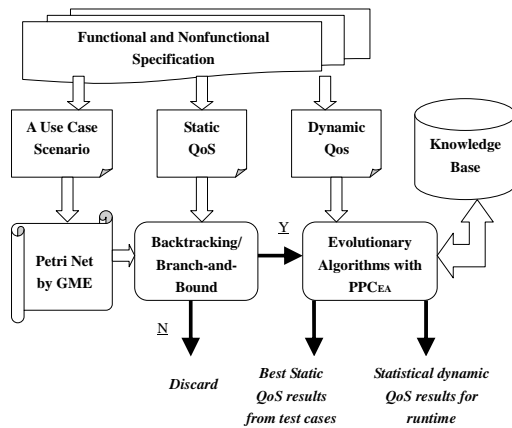


Figure 1: The QoS assurance framework.

Generic Modeling Environment (GME) [4]. A Petri Net represented by a Petri Net graph is a formalism beneficial in modeling concurrent and asynchronous systems [7]. Its characteristics are appropriate for simulating data and control flow of QoS parameters (i.e., QoS systemic paths) among components involved. In order to diagnose data and control flow from a Petri Net graph, a reachability tree is derived to show various QoS systemic paths analyzed by analysis behaviors. Analysis behaviors crosscut the source code of the Petri Net model GME interpreter generated can be weaved in by using AspectJ [1]. There are several merits to implement Petri Nets with GME and AspectJ. Because GME is a metaconfigurable modeling tool that permits the customization of visual domain languages, new features of Petri Nets can be easily added to the Petri Net metamodel to facilitate extensibility and reusability of the Petri Net model [4]. In addition, separation of concerns of simulation of QoS systemic paths and analysis behaviors promotes reusability and modularity of source code.

## 2.2 Backtracking and branch-and-bound

A Petri Net acquires numerous possible solution of a design space by modeling and analyzing a DRE system. However, most of these solutions are inappropriate for the DRE system, because they do not satisfy strict static QoS requirements. To decrease the solution space of a DRE system with strict static QoS requirements, backtracking or branch-and-bound (B/B) approaches are applied [3] at the first level of assurance: if the results of the internal nodes of the reachability tree do not satisfy strict static QoS requirements during the trace, the approach discards all subsequent branches. Unlike most of the assurance approaches such as [6] that validates one design space solution at a time, B/B approaches utilize a “parallel pruning concept” that cuts infeasible descendant leaves concurrently. Therefore, the computation time of B/B approaches is faster than those pruning approaches without the concept of parallel computation.

## 2.3 Evolutionary Algorithms

In the DRE domain, validating each QoS requirement individually may ignore the probable impacts on the effect that QoS requirements have on each other. Fitness functions of evolutionary algorithms (EAs) solve this problem by combining all of the associated concerns of QoS requirements into a mathematical formula.

B/B approaches assure the static QoS requirements that are imperative and orthogonal. However, it is time consuming for B/B approaches to evaluate non-imperative and/or non-orthogonal static QoS requirements. Hence, an EA evaluates the best results of joint non-strict static QoS parameters by a fitness function.

Evaluating dynamic QoS requires the cooperation of the deployment environment. However, the statistical results of dynamic QoS by EAs at component assembly time may serve as excellent estimates during runtime. Dynamic QoS requirement validation utilizes the previous state information of a component in the knowledge base to obtain the statistical results computed by EAs.

A domain-specific scripting language, Programmable Parameter Control for Evolutionary Algorithms (PPCEA) [5], is developed as a metaprogram of EAs to support the pruning of the design space. By a user-defined discard rate, EAs decide which assembled cases should be deleted. For example, if the worst result of an assembled case is not close to the dynamic QoS requirement, this case can be discarded before runtime.

## 3. CONCLUSION

The earlier an error is found, the less costly software is developed [2]. Our approach obeys this golden rule to reduce the design space at system assembly time. It not only lessens the workload of QoS assurance at runtime, but also economizes the development and integration cost of DRE systems constructed by assembly of components. Besides, constructing Petri Net-based QoS modeling in the GME collaborating AspectJ facilitates reusability, extensibility and flexibility. B/B approaches utilize the parallel pruning concept to expedite the reduction of design space. PPCEA provides a flexible, reusable and statistical means to delete less probable cases, and auxiliary statistical results as the reference at runtime.

## 4. REFERENCES

- [1] AspectJ. <http://eclipse.org/aspectj/>.
- [2] B. W. Boehm. *Software engineering economics*. Englewood Cliffs, N.J., Prentice-Hall, 1981.
- [3] E. Horowitz, S. Sahni, and S. Rajasekaran. *Computer Algorithms*. Computer Science Press, 1998.
- [4] Á. Lédeczi, Á. Bakay, M. Maróti, P. Völgyesi, G. Nordstrom, J. Sprinkle, and G. Karsai. Composing domain-specific design environments. *Computer*, 34(11):44–51, Nov. 2001.
- [5] S.-H. Liu, M. Mernik, and B. R. Bryant. Parameter control in evolutionary algorithms by domain-specific scripting language PPCEA. In *Proc. Int. Conf. Bioinspired Optimization Methods and Their Applications*, pages 41–50, 2004.
- [6] S. Neema, J. Sztipanovits, G. Karsai, and K. Butts. Constraint-based design space exploration and model synthesis. In *Proc. EMSOFT 2003, 3<sup>rd</sup> Intl. Conf. Embedded Software*, pages 290–305, 2003.
- [7] J. L. Peterson. Petri nets. *ACM Computing Surveys*, 9(3):223–252, Sept. 1977.
- [8] R. R. Raje, M. Auguston, B. R. Bryant, A. M. Olson, and C. C. Burt. A quality of service-based framework for creating distributed heterogeneous software components. *Concurrency and Computation: Practice and Experience*, 14(12):1009–1034, 2000.