

Early Quality-of-Service Validation of Distributed Real- time & Embedded Systems

DISA Mission Partner Conference
Tampa Bay, FL
April 7 – 10, 2012



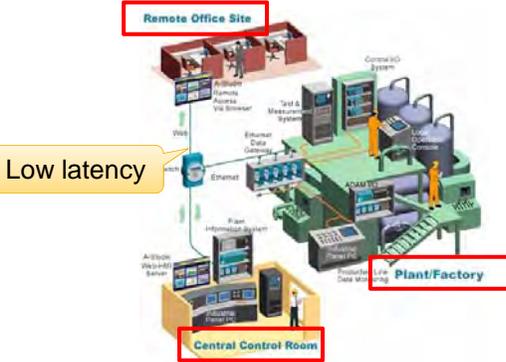
Presentation Outline

- Motivation
- Evolution & Challenges of DRE Systems
- Promising Solution: System Execution Modeling (SEM) Tools
- Examples Application of SEM Tools
- Concluding Remarks

Motivation: Large-scale DRE Systems

Large-scale distributed real-time & embedded (DRE) systems have the following characteristics:

- Stringent quality-of-service (QoS) requirements that coincide with complex functional requirements
 - e.g., performance, reliability, security, fault tolerance, & etc.
- Heterogeneity in both operational environment & technology
- Often developed as monolithic, stove-piped applications, where small changes have a large (negative) impact on quality & performance/predictability



Motivation: Large-scale DRE Systems

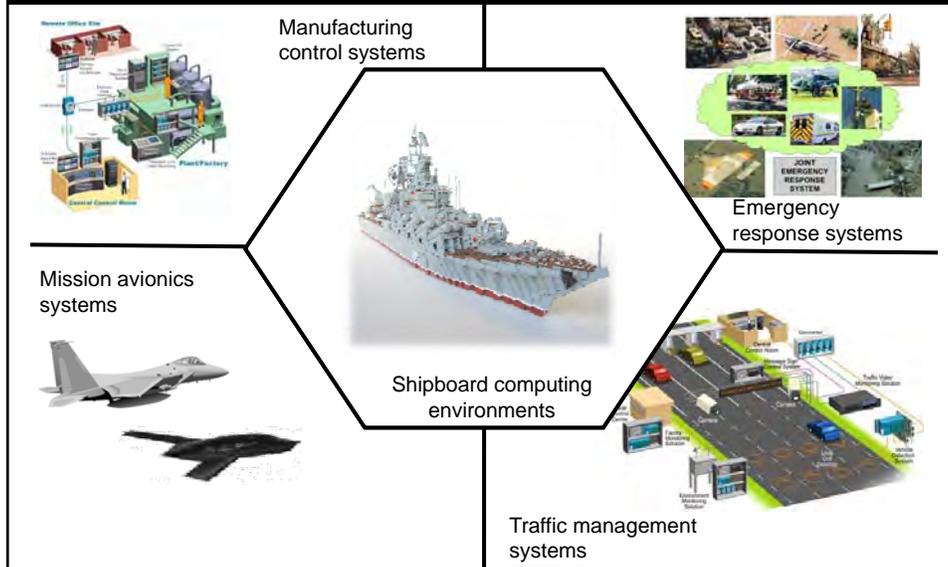
Large-scale distributed real-time & embedded (DRE) systems have the following characteristics:

- Stringent quality-of-service (QoS) requirements that coincide with complex functional requirements
 - e.g., performance, reliability, security, fault tolerance, & etc.
- Heterogeneity in both operational environment & technology
- Often developed as monolithic, stove-piped applications, where small changes have a large

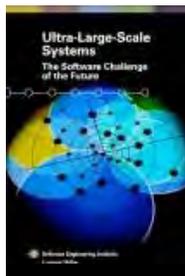


Historically, these characteristics have resulted in elongated software lifecycles realized at the expense of overrun project deadlines & effort

Example Large-scale DRE Systems



Emerging DRE Systems Domains

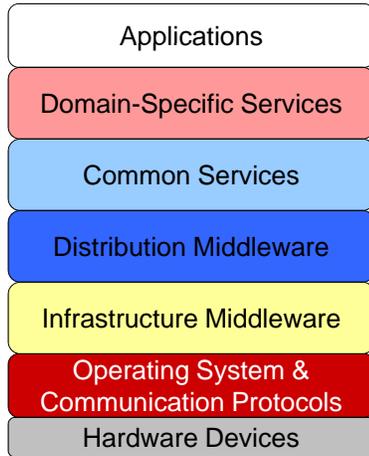


Ultra-large-scale Systems. Systems of the future that consist of billions of lines of code and are globally connected...

Cyber-Physical Systems. Systems featuring a tight combination of, and coordination between, the system's computational and physical elements.

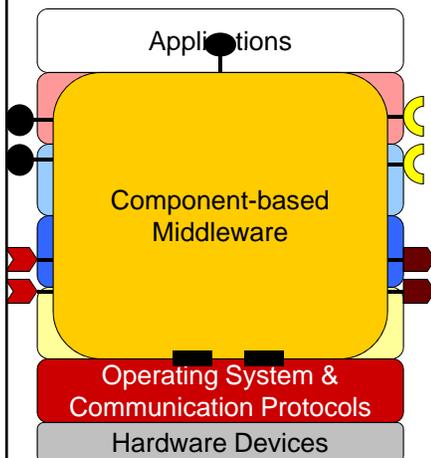


Evolution of DRE System Development



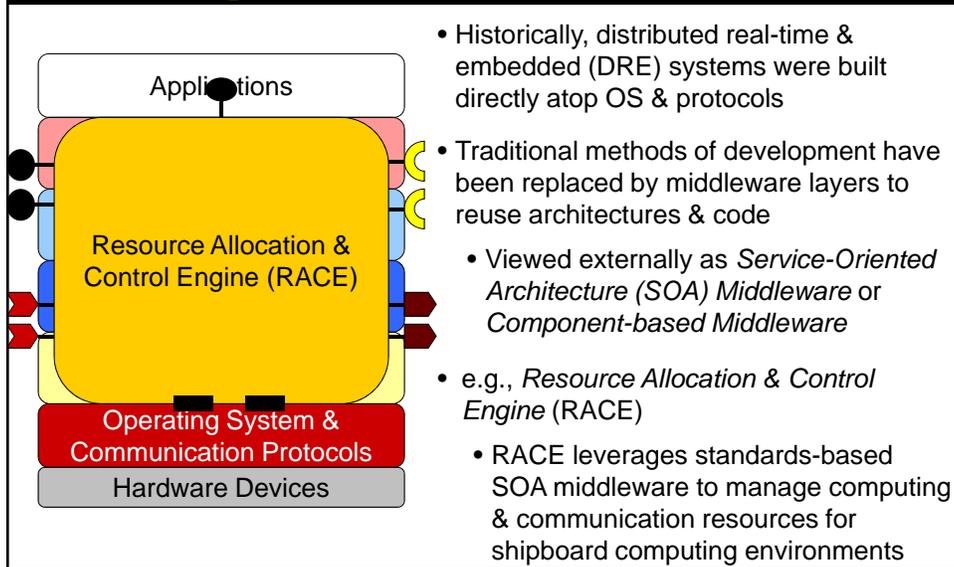
- Historically, distributed real-time & embedded (DRE) systems were built directly atop OS & protocols

Evolution of DRE System Development

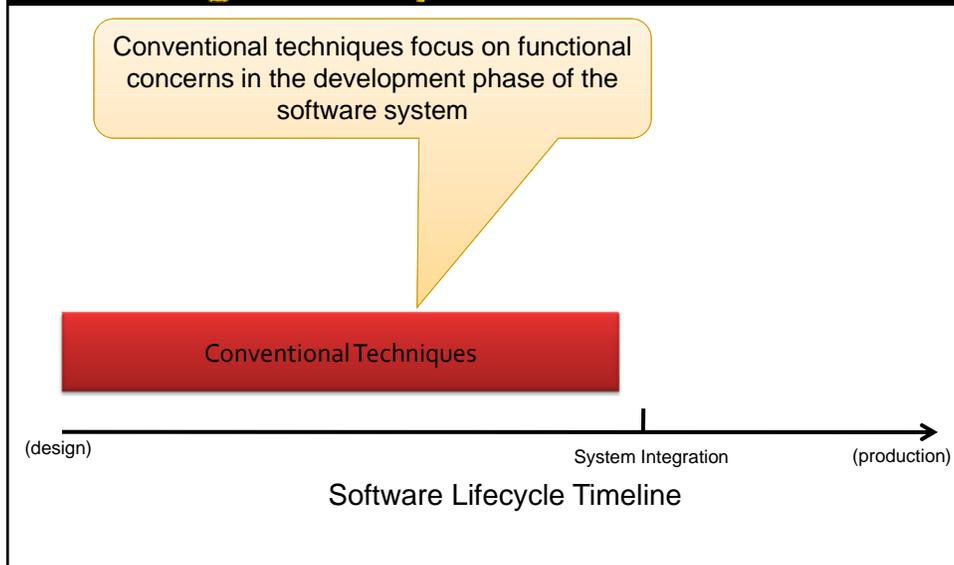


- Historically, distributed real-time & embedded (DRE) systems were built directly atop OS & protocols
- Traditional methods of development have been replaced by middleware layers to reuse architectures & code
- Viewed externally as *Service-Oriented Architecture (SOA) Middleware* or *Component-based Middleware*

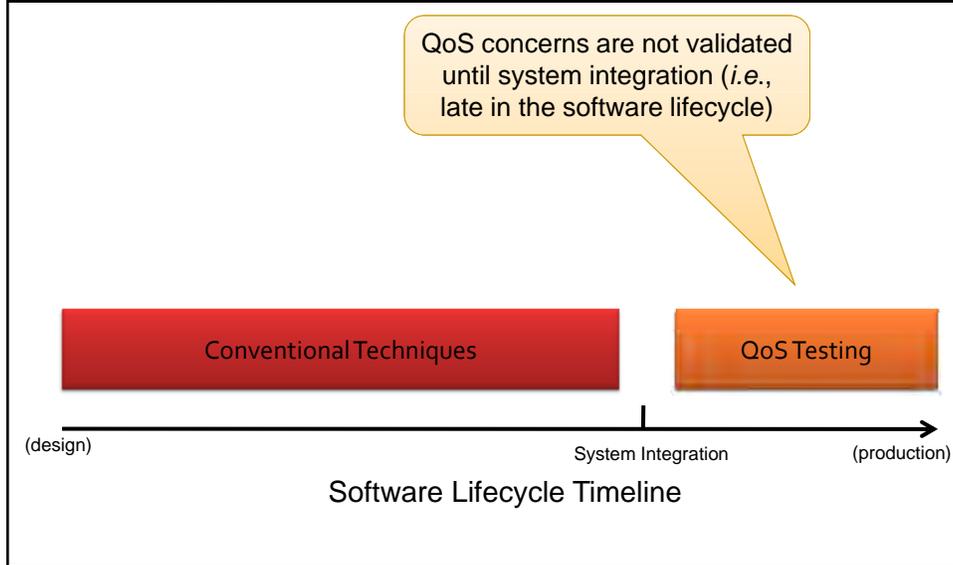
Evolution of DRE System Development



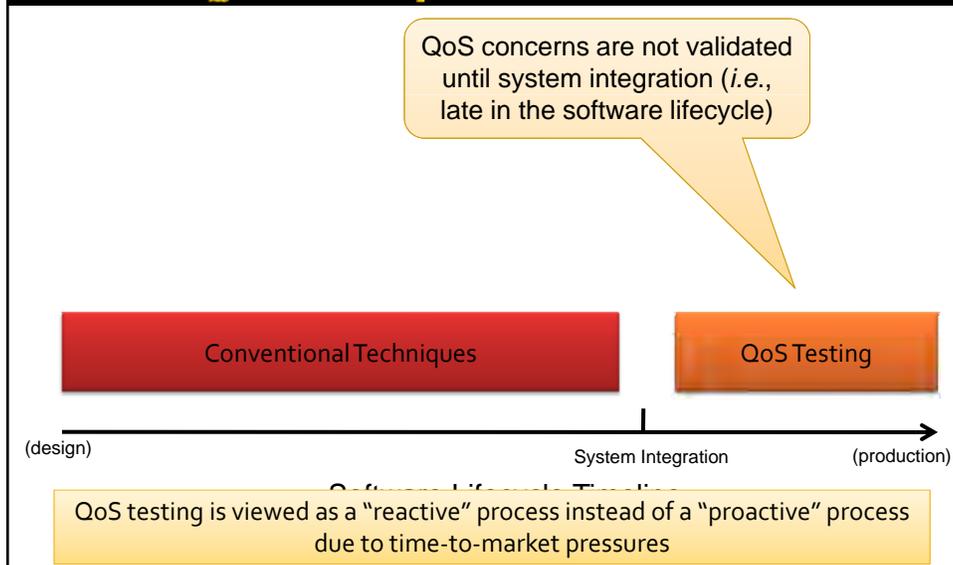
Shortcomings of Conventional Testing Techniques



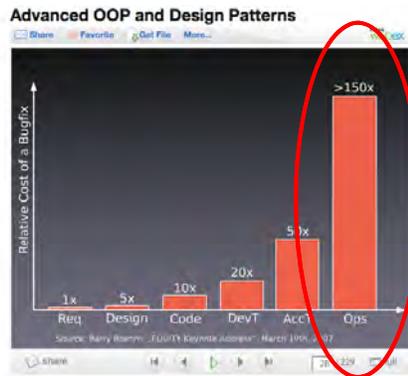
Shortcomings of Conventional Testing Techniques



Shortcomings of Conventional Testing Techniques



Relative Cost of "Reactive" QoS Testing

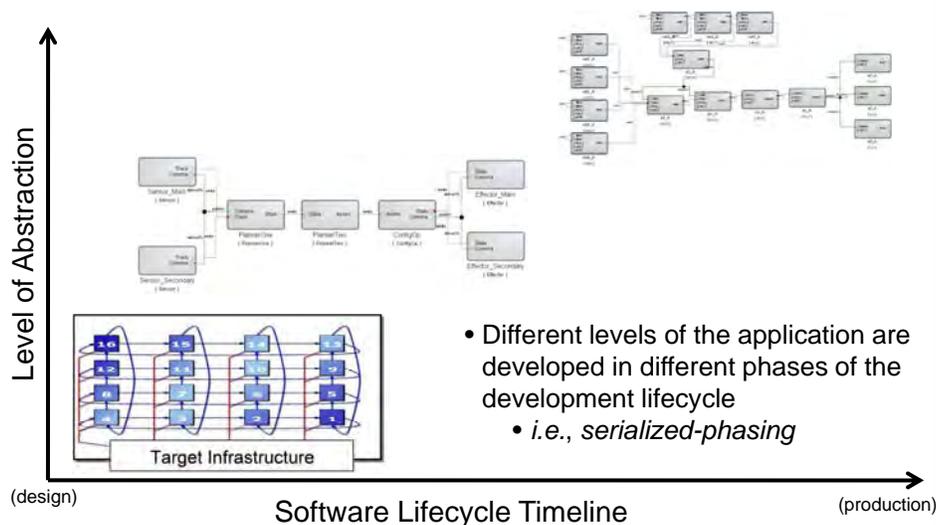


- QoS testing/validation typically begins when a system is completely developed
- In theory, you already begin with cost >150x

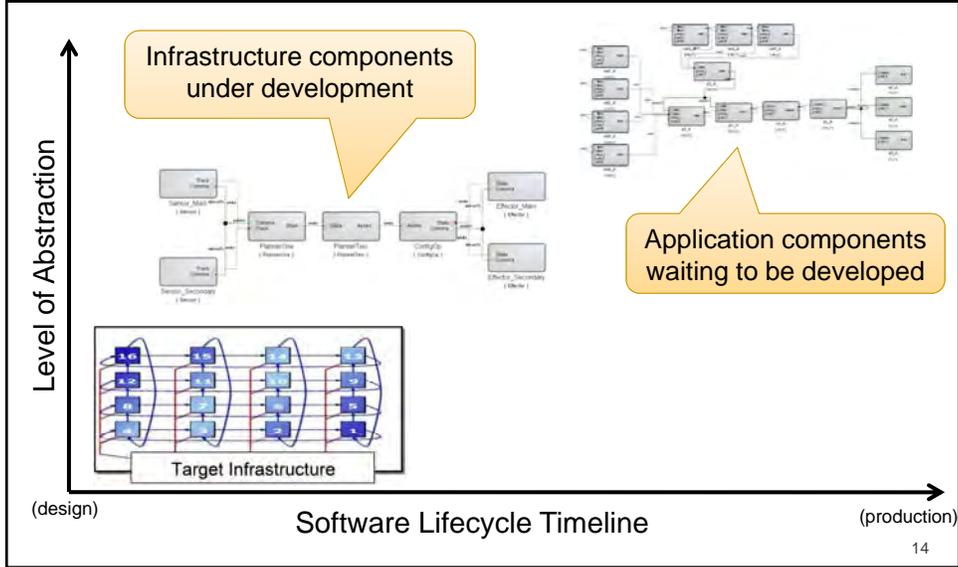
Boehm, B., *Software Engineering Economics*, Prentice Hall, New Jersey, 1981.

* Image from <http://www.superwebdeveloper.com/2009/11/25/the-incredible-rate-of-diminishing-returns-of-fixing-software-bugs/>

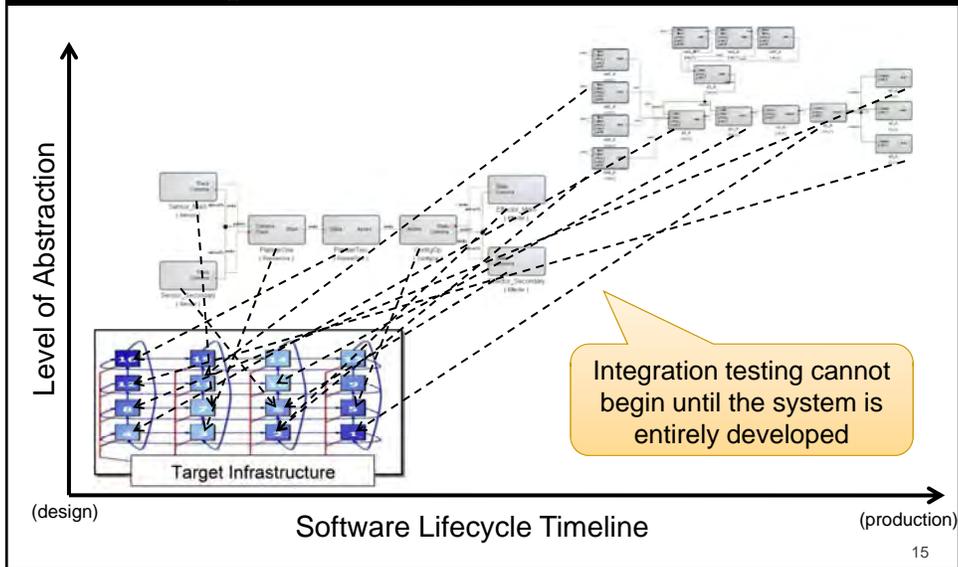
Source of Observed Gap in Software Lifecycle



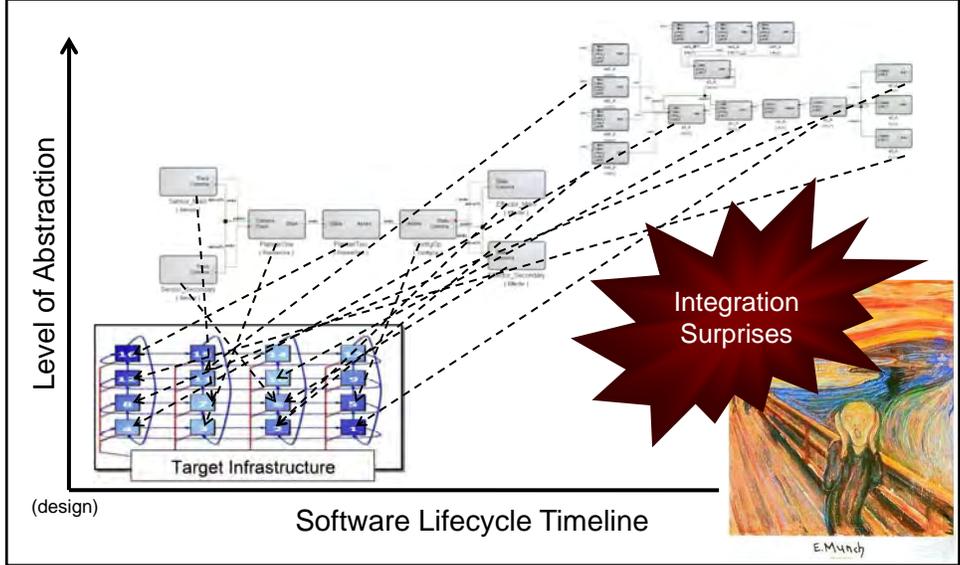
Understanding Serialized-Phasing Development (1/3)



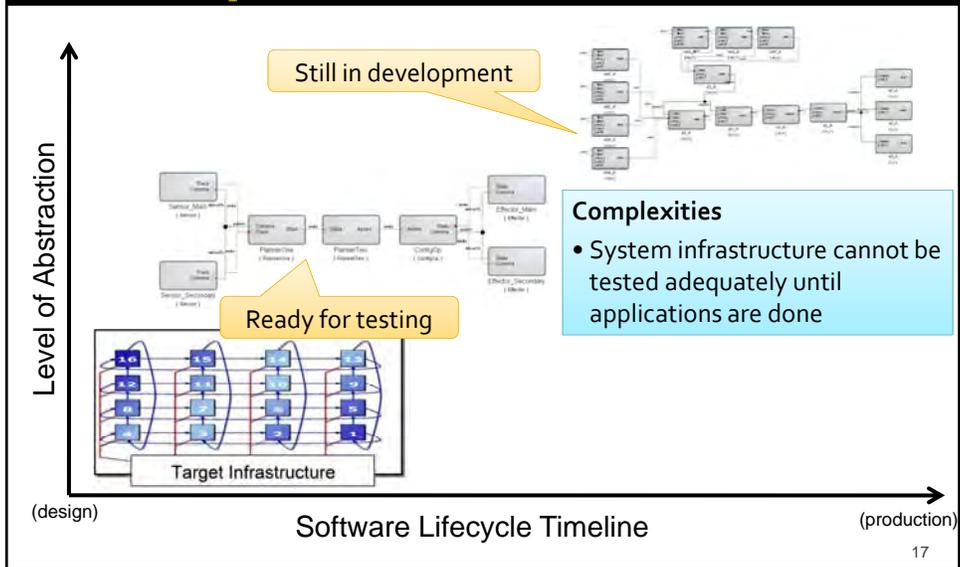
Understanding Serialized-Phasing Development (2/3)



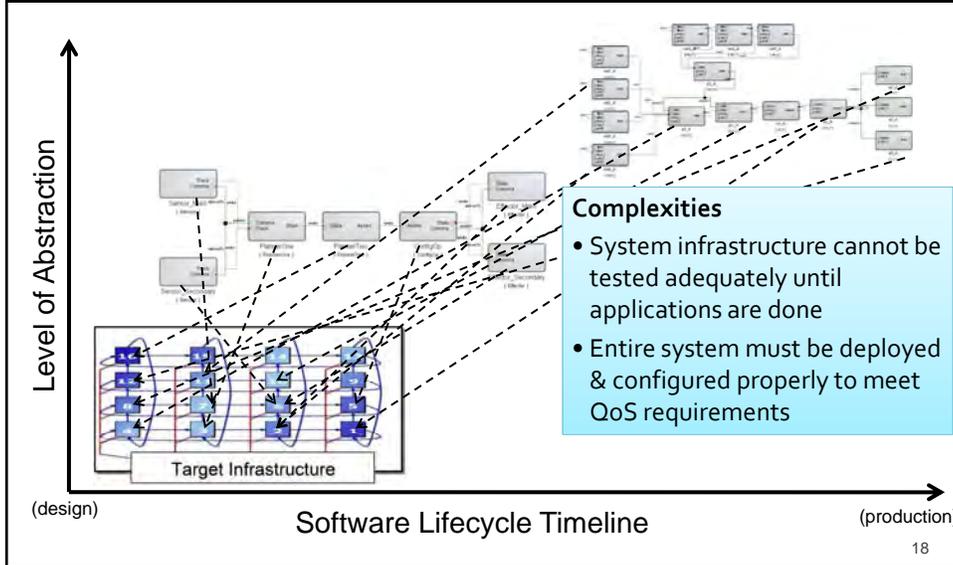
Understanding Serialized-Phasing Development (3/3)



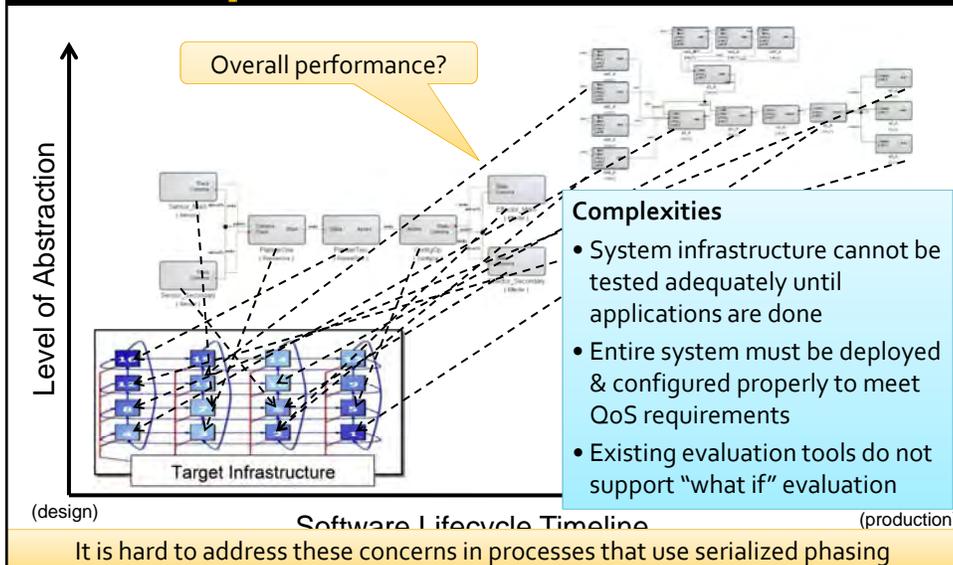
Complexities of Serialized-Phasing Development



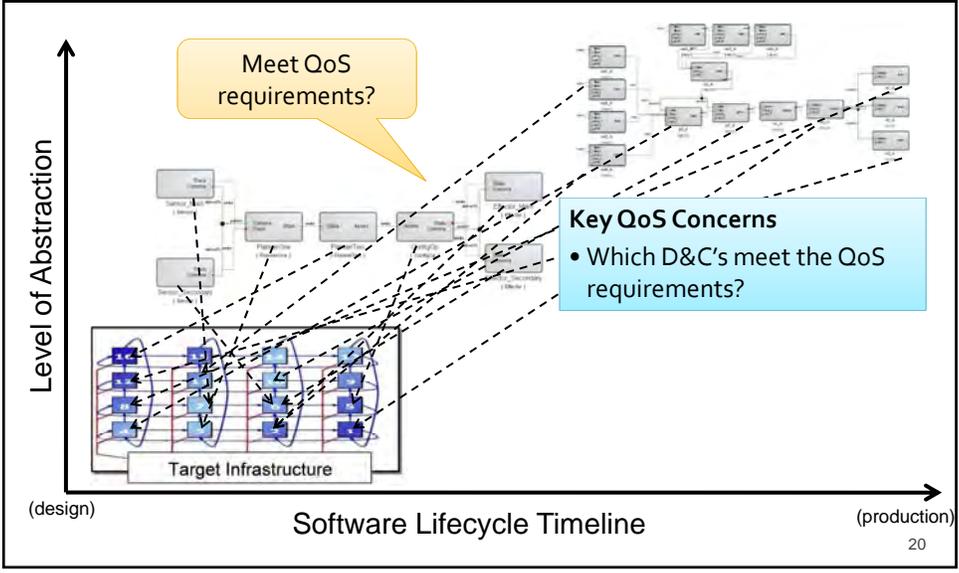
Complexities of Serialized-Phasing Development



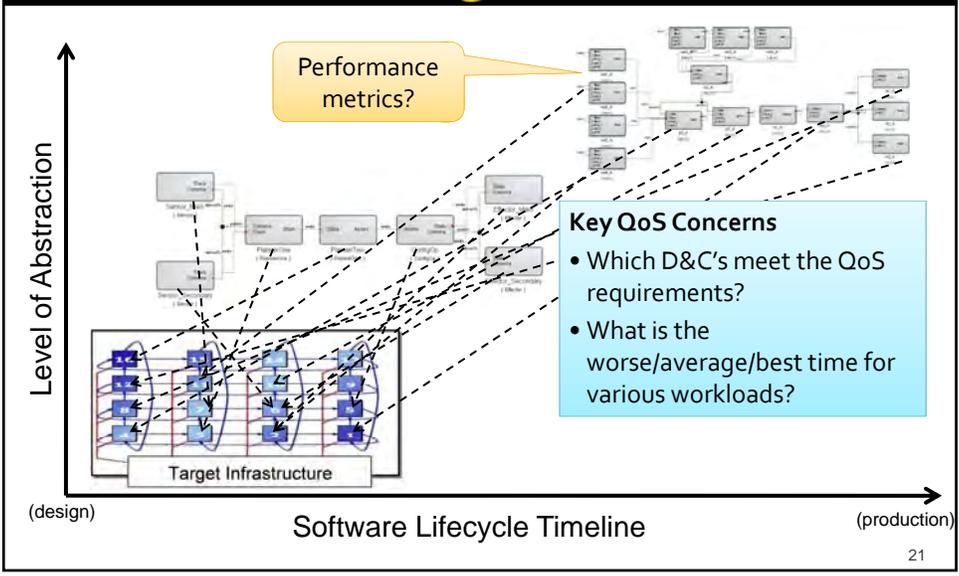
Complexities of Serialized-Phasing Development



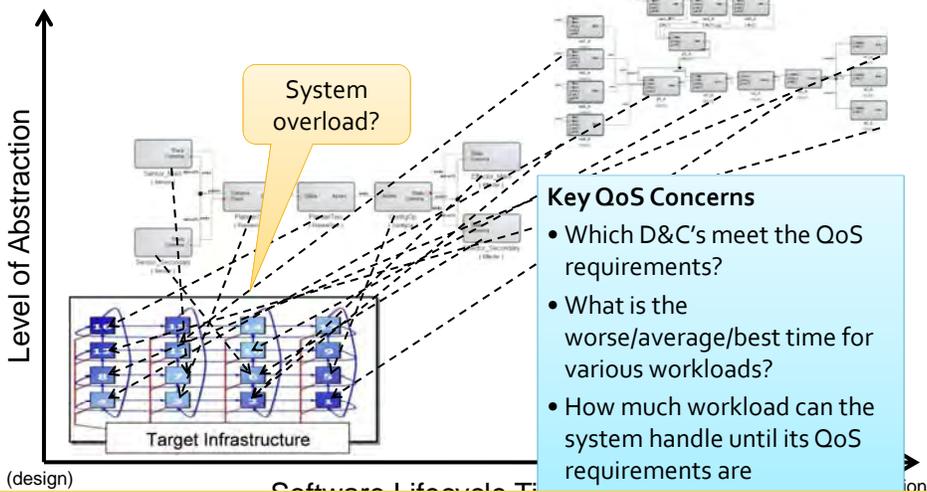
Unresolved QoS Concerns with Serialized Phasing



Unresolved QoS Concerns with Serialized Phasing



Unresolved QoS Concerns with Serialized Phasing



It is hard to address these concerns in processes that use serialized phasing

Promising Solution: System Execution Modeling Tools

Tools to express & validate design rules

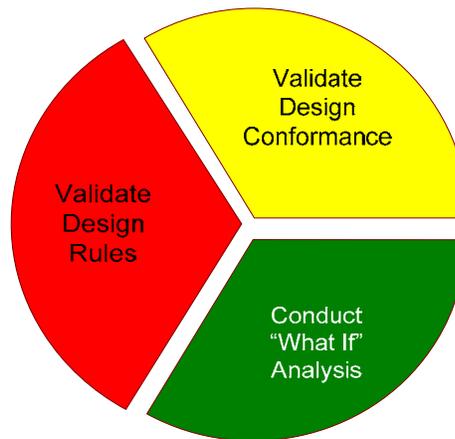
- Help applications & developers adhere to system specifications at design-time

Tools to ensure design rule conformance

- Help properly deploy & configure applications to enforce design rules throughout system lifecycle

Tools to conduct "what if" scenarios

- Help analyze QoS concerns *prior* to completing the entire system, *i.e.*, before system integration phase



Promising Solution: System Execution Modeling Tools

Tools to express & validate design rules

- Help applications & developers adhere to system specifications at design-time

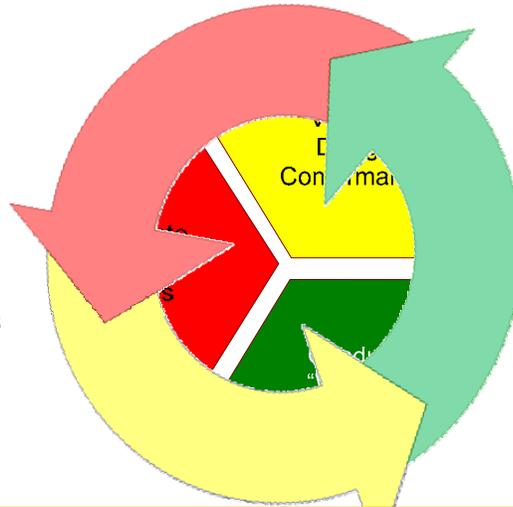
Tools to ensure design rule conformance

- Help properly deploy & configure applications to enforce design rules throughout system lifecycle

Tools to conduct "what if" scenarios

- Help analyze QoS concerns *prior* to completing the entire system, *i.e.*, before system integration phase

SEM tools should be applied continuously when developing software elements

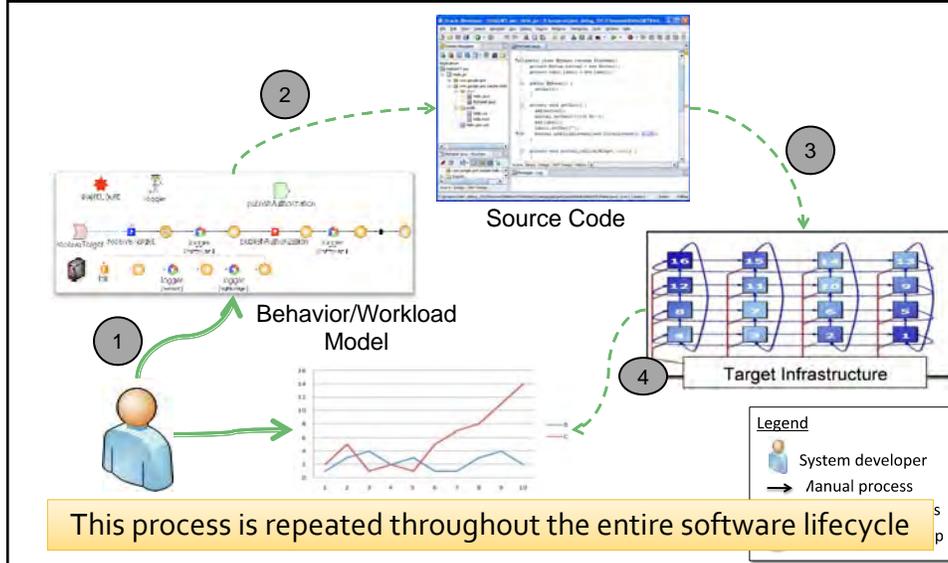


Leveraging Network Emulation Clouds & Testbeds

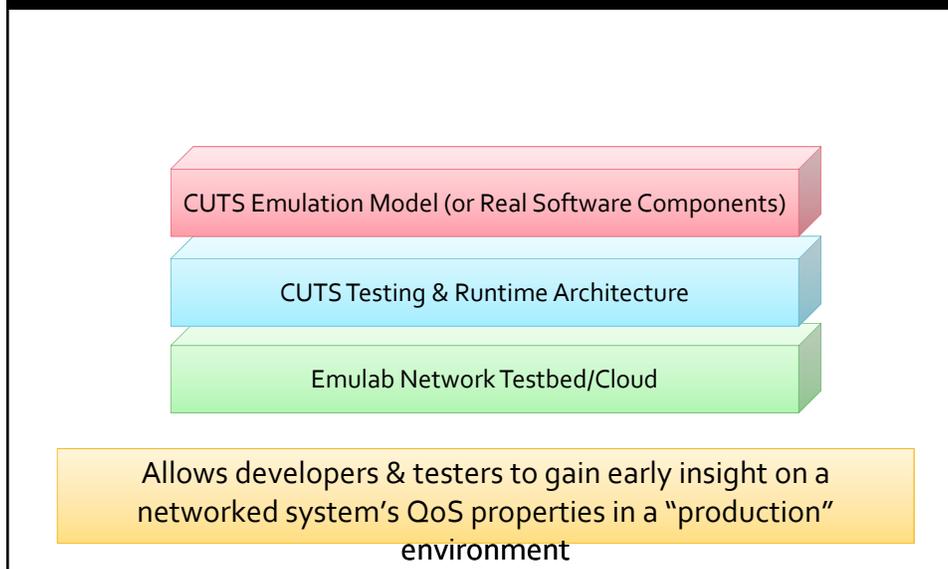
- **Emulab** – network testbed/cloud for testing, debugging, evaluating networked systems
- Allows testers to create network topologies with production characteristics
 - e.g., capacity, drop rates, traffic shaping
 - www.emulab.org
- **PlanetLab** – open-platform for constructing experiments that evaluate planetary-wide services
 - www.planet-lab.org



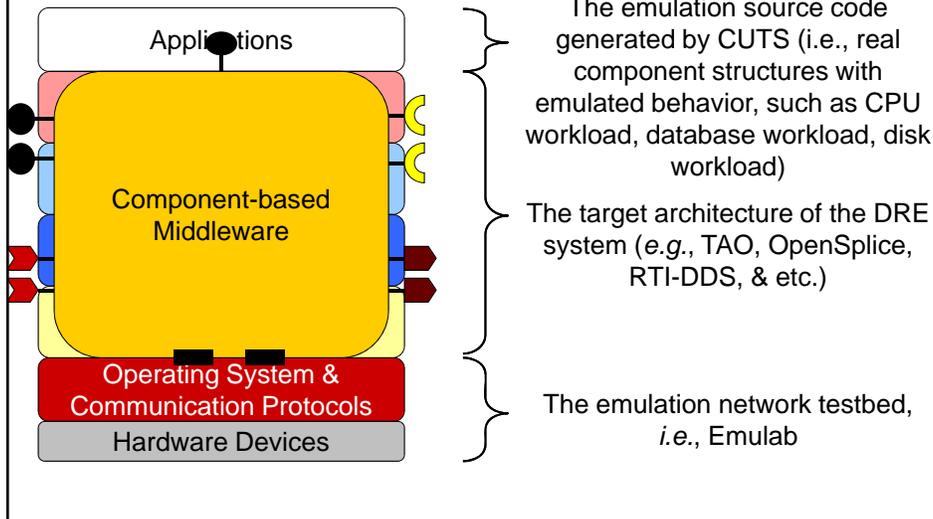
CUTS: A Next-Generation SEM Tool



Integrating SEM Tools with Network Emulation Testbeds

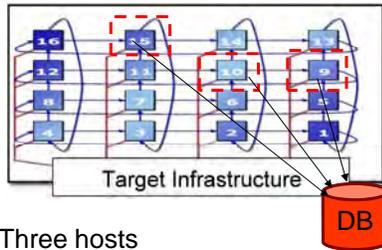
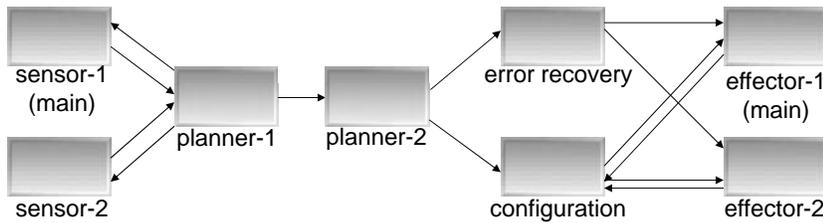


What Is Actually Being Emulated?



Examples of Applying SEM Tools to DRE Systems

SEM Example: The SLICE Scenario

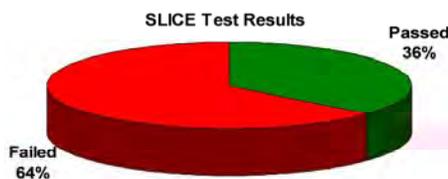


- Three hosts
- One database is shared between all hosts

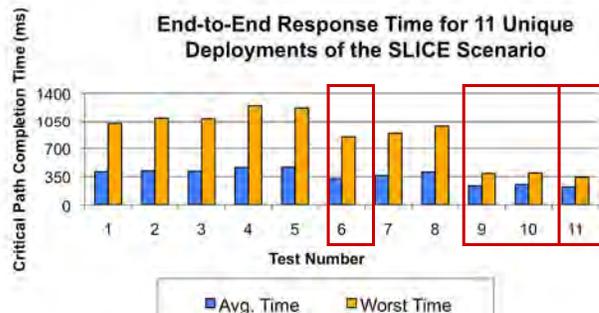
D&C & Performance Requirements

- **Critical path** deadline is 350 ms
 - main sensor to main effector through configuration
- Components in the critical paths cannot be collocated
- Main sensor & effector must be deployed on separate hosts

Early Evaluation of SLICE QoS using CUTS

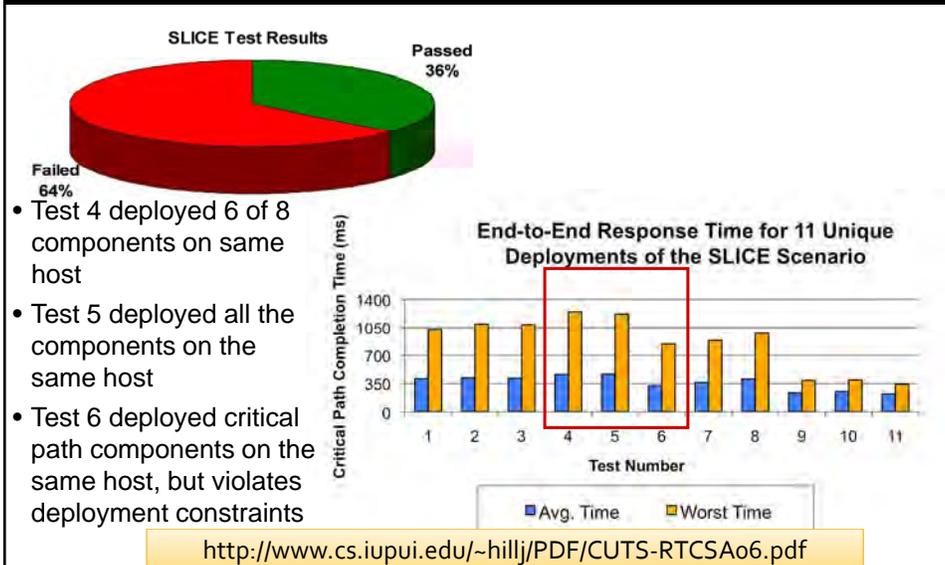


- Only 4 of 11 deployments met the 350 ms critical path deadline for average case response time
- Test 11 is the only test to meet critical path deadline for worst case response time



<http://www.cs.iupui.edu/~hillj/PDF/CUTS-RTCSAo6.pdf>

Early Evaluation of SLICE QoS using CUTS

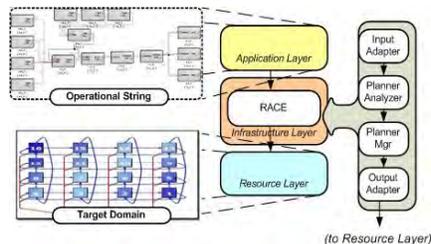


- Test 4 deployed 6 of 8 components on same host
- Test 5 deployed all the components on the same host
- Test 6 deployed critical path components on the same host, but violates deployment constraints

SEM Example: Resource Allocation and Control Engine (RACE)

The **Resource Allocation & Control Engine (RACE)** is an infrastructure-level component-based DRE system that manages application-level assemblies

- RACE supports two types of application-level deployments
 - **Static** – deployments created offline where components are pre-assigned to hosts
 - **Dynamic** – deployments created online such that components are assigned at runtime
 - e.g., based on operating conditions



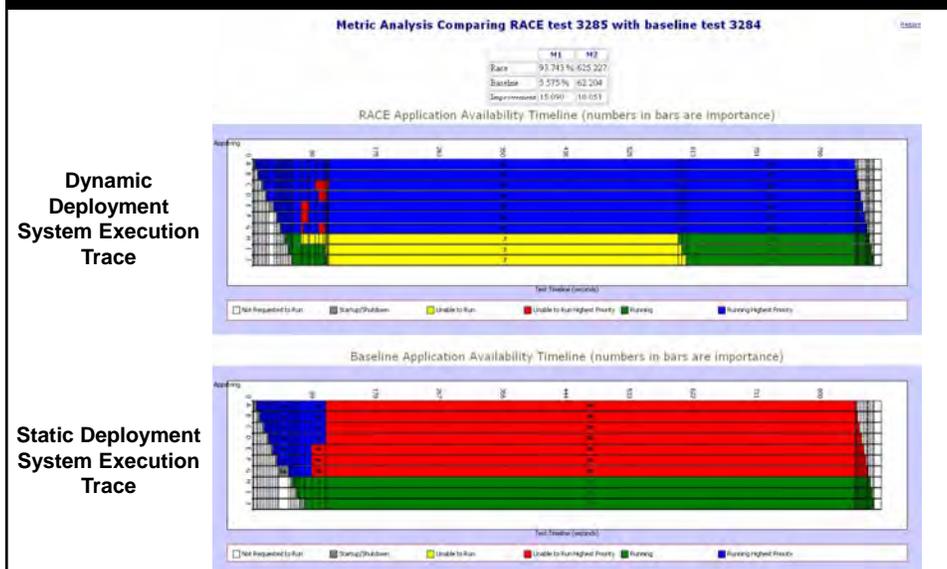
Baseline Scenario Requirement - higher priority application-level assemblies must have longer lifetime than lower priority ones

- e.g., under low resource availability

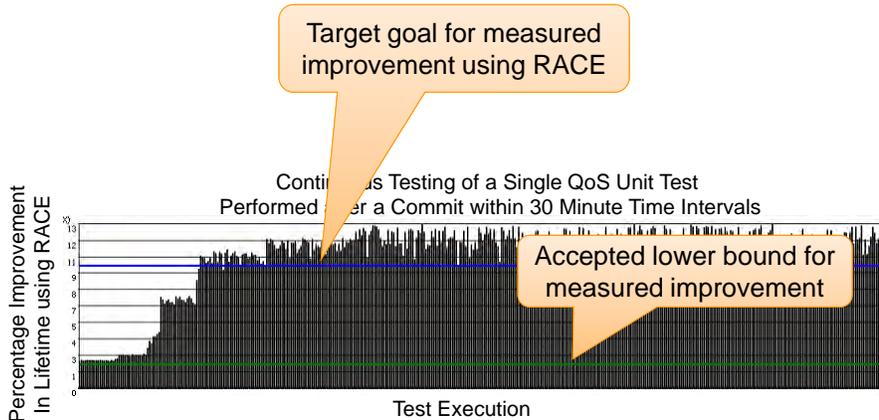
Evaluating RACE using SEM Tools

Evaluation Criteria	Description
Lifetime Improvement	RACE can improve the lifetime of workflows deployed dynamically by at least 10% vs. workflows deployed statically

Single Evaluation Using CUTS

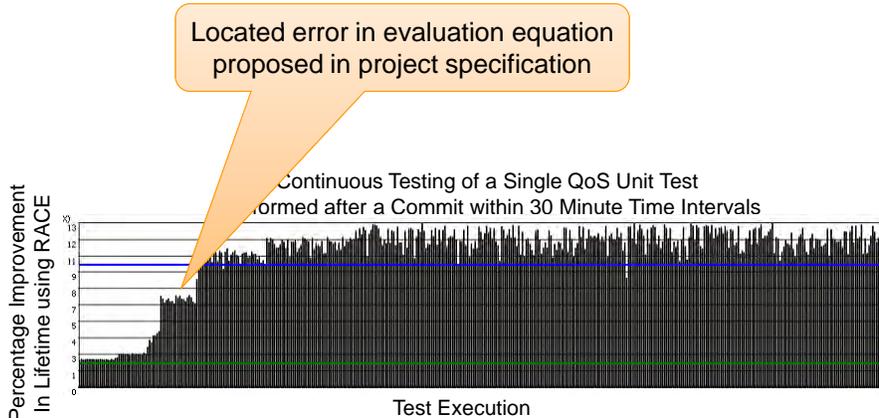


Using CUTS to Continuously Validate RACE's QoS



<http://www.cs.iupui.edu/~hillj/PDF/CiCUTS.pdf>
<http://www.cs.iupui.edu/~hillj/PDF/icst-unite.pdf>

Using CUTS to Continuously Validate RACE's QoS



<http://www.cs.iupui.edu/~hillj/PDF/CiCUTS.pdf>
<http://www.cs.iupui.edu/~hillj/PDF/icst-unite.pdf>

Using CUTS to Continuously Validate RACE's QoS

Ensured the percentage improvement was near our target continuously throughout the software lifecycle of RACE



<http://www.cs.iupui.edu/~hillj/PDF/CiCUTS.pdf>
<http://www.cs.iupui.edu/~hillj/PDF/icst-unite.pdf>

CUTS Usage on Real-World Problems & Projects

Raytheon

- Model-Driven Computing for Distributed Real-time Embedded Systems; 8/31/04 – 8/31/08



- Pollux: Enhancing the Real-time QoS of the Global Information Grid; 2/24/06 – 7/24/08
- System Execution Modeling Technologies for Large-scale Net-centric Systems; 1/1/2008 – 12/31/2010

BBN
TECHNOLOGIES

- Quality of Service Enabled Dissemination; 12/31/2007 – 9/30/2009



- CADynCE Experimentation Operations (CEO); 8/31/2007 – 12/31/2007

CUTS Usage on Real-World Problems & Projects



imagination at work

- Open Modular Embedded Architectures; 8/1/2008 to 1/31/2009



- NAOMI; 9/1/2007 to 11/30/2009



- Early Integration and Performance Testing of Heterogeneous Computing Environments; 1/1/2009 – 12/31/2010
- System Execution Modeling Research;



- pending

40

“Return On Investment” based on Prior Experience & Case Studies

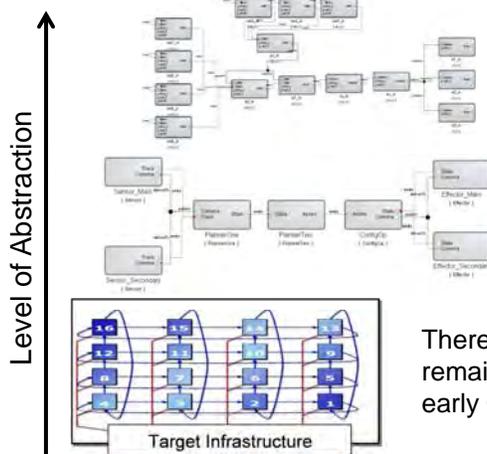
- Early identification of QoS bottlenecks
 - QoS requirements have potential to drive development effort, not customer satisfaction...
- Models can be reused in many different application domains
 - One-time upfront cost
 - The more generalized, the more reuse...
- Foundation for provisioning future systems without much upfront cost
 - “Quick-look” capabilities
- Increased QoS testing coverage...

Lessons Learned from SEM Tools

- Reduction in the semantic & knowledge gap
 - Less knowledge to realize complex scenarios
- Platform independence for rapid prototyping
 - Quick-look under competing technologies
- Infrastructure research & development
 - Provides applications to support testing



Concluding Remarks



As systems grow larger & more complex:

- the need to ensure QoS throughout software lifecycle will continue to increase
- Next-generation SEM tools are one method for addressing this need

There are still a LOT of challenges remaining in order to meet the needs of early QoS evaluation

The CUTS is available in open-source format at the following location:

<http://cuts.cs.iupui.edu>

Questions

44