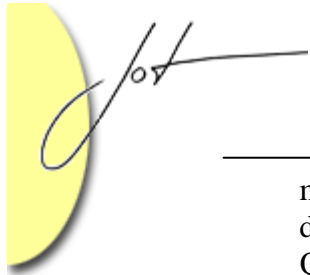


Contents

	Page
Editorial	5
<hr/> REFEREED ARTICLES <hr/>	
From formal specifications to QoS monitors <i>By Sebastien Soudrais, Olivier Barais, Laurence Duchien and Noel Plouzeau</i>	7
<p>In the domain of soft real-time application design, the gap between component specification models and the implementations often implies that the implementations cannot fully take advantage of the specification models.</p> <p>To limit this gap, this paper proposes an approach to generate a QoS monitor from the timed behavior specification. To support this approach, we rely on two different component models: one focused on formal description and the other on practical implementation. Those models are interconnected by model transformation, using a Model-Driven Engineering style.</p>	
Quality of Service Contract Specification, Establishment, and Monitoring for Service Level Management <i>By Changzhou Wang, Haiqin Wang, Alice Chen and Rodolfo Santiago</i>	25
<p>This paper describes a Quality of Service (QoS) management approach and architecture as well as a case study for Service Level Management (SLM). Our approach brings in a new perspective to the SLM problem by using QoS management and QoS Contract specification, establishment, and monitoring. In SLM, the service consumer side and the service provider side must share a common understanding of QoS characteristics and use a common language for specifying desired QoS parameters in the form of QoS contracts. A service consumer must negotiate with the service provider to establish mutually agreed QoS contracts for an interaction session. When establishing a new QoS contract, the service provider must consider both QoS contracts already agreed upon with existing consumers and system resource conditions.</p> <p>Similarly, a service consumer must be prepared in revising its contract with the service provider as conditions change over time. Once a QoS contract is established, SLM must monitor QoS status to make sure that the service quality is provided at the agreed range. If necessary, SLM</p>	



must activate adaptation mechanisms to bring the service quality to the desired level. A case study is presented in this paper to validate the QoS contract management design approach and architecture for SLM.

Model Driven Prediction and Control

45

By Assel Akzhalova, Assel Altayeva and Nurzhan Duzbayev

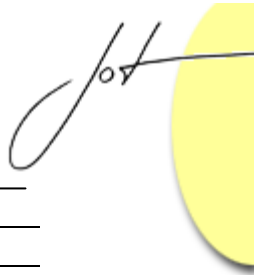
Self-adaptive systems are capable of changing their behaviour at runtime to meet target constraints. An important research question is how quality of service models can inform runtime adaptation. We sketch one solution to this question by application of control theory to improve performance of queued systems by means of architectural adaptation. Previous research by our group has shown how Auto-Regressive Integrated Moving Average techniques can be utilized to forecast how Quality of Service (QoS) characteristics are likely to evolve in the near future. This is particularly important in cases where systems can be adapted to counter QoS constraint violations. In this paper, we show how, given a similar type of QoS characteristic forecasts, strategies of architectural adaptation can be implemented that pre-emptively avoid QoS violations. The novelty of our approach is that we use classical control theory to ensure that our adaptation strategies are stable, in the sense that they do not oscillate between choices. We provide a description of how our control theoretic model can be implemented using context-based interception in .NET via model driven engineering.

Scheduling Real-Time Components Using Jitter-Constrained Streams

69

By Claude-Joachim Hamann and Steffen Zschaler

Component-based applications require good middleware support. In particular, business logic should be separated from management code for guaranteeing nonfunctional proper ties of a system. We present an approach called Container-Managed Quality Assurance, in which a component container uses nonfunctional specifications of components to determine how to use these components, and which system resources to allocate, to provide cer tain ser vices with guaranteed nonfunctional proper ties. As an example, we show how this technique can be applied to automatically allocating CPU and memory resources for components with real-time constraints. To this end, we use a mathematical model based on jitter-constrained streams, a mathematical abstraction of event streams.



CONTENTS

OUTLOOK

A brief outlook to the next issue

87