Adapting Distributed Real-time and Embedded Pub/Sub Middleware for Cloud Computing Environments *

Joe Hoffert**, Douglas C. Schmidt, and Aniruddha Gokhale

Vanderbilt University, VU Station B #1829, 2015 Terrace Place, Nashville, TN 37203

Abstract. Enterprise distributed real-time and embedded (DRE) publish/subscribe (pub/sub) systems manage resources and data that are vital to users. Cloud computing—where computing resources are provisioned elastically and leased as a service—is an increasingly popular deployment paradigm. Enterprise DRE pub/-sub systems can leverage cloud computing provisioning services to execute needed functionality when on-site computing resources are not available. Although cloud computing provides flexible on-demand computing and networking resources, enterprise DRE pub/sub systems often cannot accurately characterize their behavior *a priori* for the variety of resource configurations cloud computing supplies (*e.g.*, CPU and network bandwidth), which makes it hard for DRE systems to leverage conventional cloud computing platforms.

This paper provides two contributions to the study of how autonomic configuration of DRE pub/sub middleware can provision and use on-demand cloud resources effectively. We first describe how supervised machine learning can configure DRE pub/sub middleware services and transport protocols autonomically to support end-to-end quality-of-service (QoS) requirements based on cloud computing resources. We then present results that empirically validate how computing and networking resources affect enterprise DRE pub/sub system QoS. These results show how supervised machine learning can configure DRE pub/sub middleware adaptively in < 10 μsec with bounded time complexity to support key QoS reliability and latency requirements.

Keywords: Autonomic configuration, pub/sub middleware, DRE systems, cloud computing

1 Introduction

Emerging trends and challenges. Enterprise distributed real-time and embedded (DRE) publish/subscribe (pub/sub) systems manage data and resources that are critical to the ongoing system operations. Examples include testing and training of experimental aircraft across a large geographic area, air traffic management systems, and disaster recovery operations. These types of enterprise DRE systems must be configured correctly to leverage available resources and respond to the system deployment environment. For example, search and rescue missions in disaster recovery operations need to configure the image resolution used to detect and track survivors depending on the available resources (*e.g.*, computing power and network bandwidth) [20].

* This work is sponsored by NSF TRUST and AFRL.

** Contact author's email address: jhoffert@dre.vanderbilt.edu

Many enterprise DRE systems are implemented and developed for a specific computing/networking platform and deployed with the expectation of specific computing and networking resources being available at runtime. This approach simplifies development complexity since system developers need only focus on how the system behaves in one operating environment. Thus considerations of multiple infrastructure platforms are ameliorated with respect to system quality-of-service (QoS) properties (*e.g.*, responsiveness of computing platform, latency and reliability of networked data, etc.). Focusing on only a single operating environment, however, decreases the flexibility of the system and makes it hard to integrate into different operating environments, *e.g.*, porting to new computing and networking hardware.

Cloud computing [6, 17] is an increasingly popular infrastructure paradigm where computing and networking resources are provided to a system or application as a service—typically for a "pay-as-you-go" usage fee. Provisioning services in cloud environments relieve enterprise operators of many tedious tasks associated with managing hardware and software resources used by systems and applications. Cloud computing also provides enterprise application developers and operators with additional flexibility by virtualizing resources, such as providing virtual machines that can differ from the actual hardware machines used.

Several pub/sub middleware platforms (such as the Java Message Service [16], and Web Services Brokered Notification [14]) can (1) leverage cloud environments, (2) support large-scale data-centric distributed systems, and (3) ease development and deployment of these systems. These pub/sub platforms, however, do not support fine-grained and robust QoS that are needed for enterprise DRE systems. Some large-scale distributed system platforms, such as the Global Information Grid [1] and Network-centric Enterprise Services [2], require rapid response, reliability, bandwidth guarantees, scalability, and fault-tolerance.

Conversely, conventional cloud environments are problematic for enterprise DRE systems since applications within these systems often cannot characterize the utilization of their specific resources (*e.g.*, CPU speeds and memory) accurately *a priori*. Consequently, applications in DRE systems may need to adjust to the available resources supplied by the cloud environment (*e.g.*, using compression algorithms optimized for given CPU power and memory) since the presence/absence of these resources affect timeliness and other QoS properties crucial to proper operation. If these adjustments take too long the mission that the DRE system supports could be jeopardized.

Configuring an enterprise DRE pub/sub system in a cloud environment is hard because the DRE system must understand how the computing and networking resources affect end-to-end QoS. For example, transport protocols provide different types of QoS (*e.g.*, reliability and latency) that must be configured in conjunction with the pub/sub middleware. To work properly, however, QoS-enabled pub/sub middleware must understand how these protocols behave with different cloud infrastructures. Likewise, the middleware must be configured with appropriate transport protocols to support the required end-to-end QoS. Manual or *ad hoc* configuration of the transport and middleware can be tedious, error-prone, and time consuming.

Solution approach → Supervised Machine Learning for Autonomous Configuration of DRE Pub/Sub Middleware in Cloud Computing Environments. This

paper describes how we are (1) evaluating multiple QoS concerns (*i.e.*, reliability and latency) based on differences in computing and networking resources and (2) configuring QoS-enabled pub/sub middleware autonomically for cloud environments based on these evaluations. We have prototyped this approach in the *ADAptive Middleware And Network Transports* (ADAMANT) platform, which addresses the problem of configuring QoS-enabled DRE pub/sub middleware for cloud environments. Our approach provides the following contributions to research on autonomic configuration of DRE pub/sub middleware in cloud environments:

- **Supervised machine learning as a knowledge base to provide fast and predictable resource management in cloud environments**. *Artificial Neural Network* (ANN) tools determine in a timely manner the appropriate transport protocol for the QoS-enabled pub/sub middleware platform given the computing resources available in the cloud environment. ANN tools are trained on particular computing and networking configurations to provide the best QoS support for those configurations. Moreover, they provide predictable response times needed for DRE systems.

- **Configuration of DRE pub/sub middleware based on guidance from supervised machine learning**. Our ADAMANT middleware uses the *Adaptive Network Transports* (ANT) [10] to select the transport protocol(s) that best address multiple QoS concerns for given computing resources. ANT provides infrastructure for composing and configuring transport protocols using the scalable reliable multicast-based Ricochet transport protocol [3]. Supported protocols such as Ricochet enable trade-offs between latency and reliability to support middleware for enterprise DRE pub/sub systems in cloud environments.

We have implemented ADAMANT using multiple open-source pub/sub middleware implementations (*i.e.*, OpenDDS(`www.opendds.org`) and OpenSplice(`www.opensplice.org`)) of the OMG Data Distribution Service (DDS) [18] specification. DDS defines a QoS-enabled DRE pub/sub middleware standard that enables applications to communicate by publishing information they have and subscribing to information they need in a timely manner. The OpenDDS and OpenSplice implementations of DDS provide pluggable protocol frameworks that can support standard transport protocols (such as TCP, UDP, and IP multicast), as well as custom transport protocols (such as Ricochet and reliable multicast).

Our prior work [10, 11] developed composite metrics to evaluate pub/sub middleware with various ANT-based transport protocols based on differences in application parameters (*e.g.*, number of data receivers and data sending rate). We also evaluated multiple approaches for adapting to application parameter changes in a dedicated (*i.e.*, non-cloud) operating environment without regard to changes in computing or networking resources. This paper extends our prior work by (1) evaluating pub/sub middleware in a cloud environment to take into account differences in computing and networking resources and (2) conducting empirical evaluations of an artificial neural network machine learning tool with respect to timeliness and configuration accuracy.

We validated ADAMANT by configuring Emulab (`www.emulab.net`) to emulate a cloud environment that allows test programs to request and configure several types of computing and networking resources on-demand. We then applied several composite metrics developed to ascertain how ADAMANT supports relevant QoS concerns