Model-driven Generative Framework for Automated OMG DDS Performance Testing in the Cloud

Kyoungho An, Takayuki Kuroda and Aniruddha Gokhale
ISIS, Vanderbilt University
{kyoungho, kuroda, gokhale}@isis.vanderbilt.edu

Sumant Tambe and Andrea Sorbini
Real-Time Innovations
{sumant, sorbini}@rti.com

Motivation
- The Object Management Group’s (OMG) Data Distribution Service (DDS) provides many configurable policies which determine end-to-end quality of service (QoS) of applications

Solution
- Automated performance testing framework called AUTOMATIC (AUTomated Middleware Analysis and Testing In the Cloud)
- Activity Domains
  - User - Modeling, Monitoring
  - Test-Automation System - Test Planning, Test Deployment
- Cloud Infrastructure - Test Emulation

Test Deployment
- To deploy the XML-based DDS testing applications, specifications related to the deployment are generated by the model interpreter
- Our deployment tool
  - Deploys the XML-based DDS testing applications in a cloud platform
  - Executes remotely the applications with a simulation tool called RTI Prototyper

Test Monitoring
- We employed a tool to visualize monitoring data of applications called RIT Monitor
- It helps users to understand DDS systems easily via graphical interfaces and to verify behaviors of entities as expected

Test Case 1
- We selected 10 DDS applications including one DDS application by combining the RELIABILITY, HISTORY, and DEADLINE QoS policies
- We measured performance parameters such as time, throughput, and resource usage

Test Case 2
- We measured performance parameters such as time, throughput, and resource usage

Challenge
- It is challenging
  - To predict the system’s performance in terms of latency, throughput, and resource usage
- Because...
  - Diverse combinations of QoS configurations influence QoS of applications

Framework Architecture

Domain-Specific Modeling Language
- We developed a DSDL using the Generic Modeling Environment (GME)
- Modeling a DDS application for simulation and testing its performance for various combinations of DDS QoS policies

Experience
- The experiment evaluates performance of an example DDS application by combining the RELIABILITY, HISTORY, and DEADLINE QoS policies
- Test Environment
  - OpenStack based cloud test bed employing KVM as a hypervisor
  - Each VM type has 1 vCPU and 512 MB
- DEADLINE QoS: 1 millisecond
- HISTORY QoS: KEEP_ALL

References
- Patrick Th. Eugster, Pascal A. Felber, Rachid Guerraoui, and Anne-Marie Kermarrec.
  - Middleware Analysis and Testing In the Cloud (MATTIC), 2004, pages 139–145.
  - Proceedings of the Third International Conference on Distributed Event-Based Systems (DEBS)

  - Proceedings of the First International Conference on Embedded Networked Sensors (EmNets)

  - Proceedings of the First International Conference on Embedded Networked Sensors (EmNets)

  - Proceedings of the First International Conference on Embedded Networked Sensors (EmNets)

  - Proceedings of the First International Conference on Embedded Networked Sensors (EmNets)

  - Proceedings of the First International Conference on Embedded Networked Sensors (EmNets)