Volume 2, Issue 4, April 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Throughput Regulation of Messaging Servers – An Intelligent Control Solution

Ravi Kumar G
HP Bangalore, Research Scholar JNTUH
ravikgullapalli@gmail.com

Dr.C.Muthusamy
Yahoo, Bangalore

Dr.A.Vinaya Babu
Principal, JNTUH Coll of Engg

Abstract— **Enterprise Messaging is a significant building block in message exchange mechanisms at enterprise level. Higher message throughput is always a desired objective to be supported by the Enterprise Messaging Servers. The typical performance tuning of Messaging Servers is carried out either through manual tuning of the server parameters or by implementing conditional programming when the messaging servers are developed. It is evident that these two mechanisms have the limitations of huge manual intervention and inabilities to implement runtime dynamics respectively. In order to address these challenges we had an initial success of applying classic controllers in self-regulating message throughput of Messaging servers. In this paper we are extending our work by proposing an intelligent control solution which indicated a performance improvement over using classic controllers.**

Keywords— **Enterprise Messaging Servers, Message Throughput, Feedback Control Systems, Intelligent control**

I. INTRODUCTION

Enterprise Messaging is a prominent message exchange mechanism in Distributed Computing Systems in Enterprise and internet environments [1]. It allows the different applications to send and receive messages. This includes legacy systems such as business workflow applications, data warehouses, EAI [2]. Enterprise Messaging simplifies the integration of applications using standard based asynchronous messaging. It is important the Enterprise Messaging Servers provide high performance. There are various solutions in providing high performance during the messaging server operation but there are challenges associated with such procedures. We have successfully investigated the applicability of control systems theory in self-managing the performance of Enterprise Messaging Servers [3]. In this paper we have extended our work with improved version of applying a control system based solution to self-regulate the performance. The Java based Messaging Servers (JMS [4]) are used for discussion in this paper, which are referred as JMS Providers [5].

II. PROBLEM AND RELATED WORK

Typically the performance of JMS Providers is measured by its message throughput, which will depend upon various factors such as the number of subscribers, message size, number of publishers, and number of JMS message brokers. The performance can be tuned by adjusting these parameters. There are solutions to improve JMS provider's performance by following few best practices such as setting non-durable messages, set the message time to live parameter appropriately, close message publishers and subscribers when they complete their jobs [6]. But these kinds of practices will be limited and cannot address different kinds of JMS environments. The other mechanism is to provide the facilities to the administrators to configure [7] and fix the various parameters values which influence the JMS Provider performance. But it is a challenging task for administrator to tune these values accurately and periodically, due to the dynamics of workload. During sudden huge loads administrator may decide to add additional resources but they may be left unutilized [8] later when there are relatively lesser loads leading ineffective IT asset utilization. Another way to manage the performance is to include conditional programming within JMS Provider implementation to change the values of the parameters at runtime based on the workload and deviation from the expected performance. This method is very complex because during design the workload dynamics needs to be accurately estimated. Implementing the conditional programming is very complex [9] as the conditions implemented may not be sufficient to meet the run time dynamics, any spikes in the workload. We provided a solution to address these issues using feedback control mechanism which improved the message throughput. But our previous work has a limitation of using a simple P-controller and a Time-Series Controller [10]. In this paper we have improved upon our previous version, by introducing an intelligent control framework which analyses the patterns in the data and predicts the future load patterns on the Message Server. Additionally a dynamic control selector is used to choose a right controller based on the predicted pattern.

III. MODELING

The system to be controlled needs to be modelled before applying any control mechanisms. We have considered a

SISO ARMA model. According to ARMA [11], in a Single Input Single Output (SISO) model [12], for a given sample data set, the next sample of the output can be predicted using the current and previous inputs. The same is explained in the equation (1) below:

$$y(t + 1) = ay(t) + bu(t) \qquad (1)$$

1) *Parameter Estimation*:

In JMS Provider message throughput is defined as a function of the following:

- The number of subscribers of the JMS Provider
- The number of publishers and number of message brokers also influence the JMS Provider

In this paper we considered only the number of subscribers affecting the performance. The ARMA model if applied to model the JMS Provider, it is represented by the equation (2):

$$T(t + 1) = aT(t) + bS_{max}(t) \qquad (2)$$

Where
$T(t)$ = current output of message Throughput
$S_{max}(t)$ = current input of maximum number of Subscribers
a = model parameter to be estimated
b = model parameter to be estimated
$T(t + 1)$ = output in the next step

The ARMA model is used to estimate the model parameters 'a' and 'b'. The details of the experiments and the estimated values are shown in the section V. "Implementation and Analysis" Table II.

2) *Operating Range*:

It is important to determine the operating range of the maximum number of Subscribers (Smax). The training data set is used again to determine the range of Smax that provides the desired Tref.

In order to achieve the desired value of the Tref, the maximum number of subscribers will have to be adjusted. This value of Smax again will change during runtime due to the stochastic nature of the load and the controller is useful to automatically adjust the Smax to meet the Tref.

IV. Intelligent Control

Intelligent control architecture is proposed to self-regulate message throughput of the Enterprise Messaging Servers. The Fig 1 below shows the proposed Intelligent Control solution. The intelligent control components are present in the outer loop help in self-regulating the performance of Enterprise Messaging Server effectively.

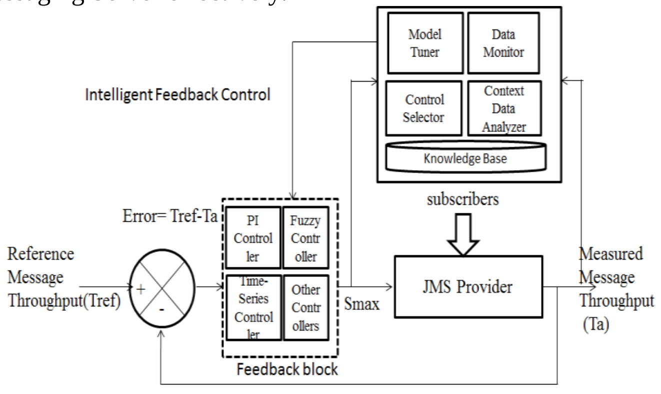

Fig. 1 Intelligent Control Solution

The performance data is monitored and captured, analysed and indicates the future patterns of the managed system behaviour. This indication will be used to determine the right controller for tuning the controller parameters. Such a controller present in the outer loop forwards the Context Pattern to the feedback block. Though the current controller set in the feedback block is limited but this will be extended as part of our bigger research work in which we are building an Expert Control System Solution for Application Servers [13] [14]. The Fig 2 below shows the Intelligent Control Architecture whose components are spread in the in the outer loop and the feedback block of the control system.

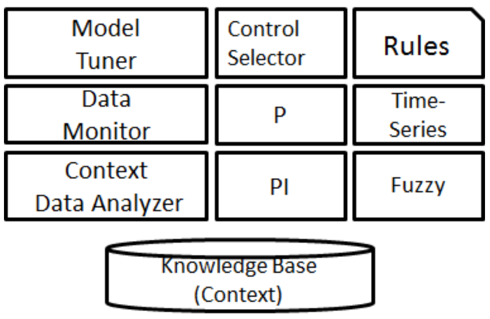

Fig. 2 Intelligent Control Architecture

The Context Data Analyser analyses the performance data captured by the Data Monitor and predicts the future patterns that we refer as Context Patterns, of the system. The sample Context Pattern definition is shown in the Fig 3 below:

```
<ContextPattern>
  <Pattern>SuddenVariations</Pattern>
  <OutParam>MessageThroughput</OutParam>
  <OutParamVal>50</OutParamVal>
  <InParam>Smax</InParam>
  <InParamVal>75</InParamVal>
</ContextPattern>
```

Fig. 3 Context Pattern Definition

The Controller will use the Context Pattern definition that contains the predicted values of the output parameter indicating whether the system's performance is going to be affected. Such information is used by the respective controllers and which determine the controller parameters.

There is a rule mapping between the Context Pattern generated and the corresponding Controllers to be used. The rule mapping is shown in the Table I below:

TABLE I
CONTEXT PATTERNS AND CONTROLLERS RULES

Context Pattern	Controllers	Action
Sudden Variations	Fuzzy	
Gradual increase	Time-Series	
Constant	Time-Series	
Quick Adaptation	PID	
Model Tune		Tune the model parameters

Based on the context pattern, the control selector updates the Context Pattern by adding another field `ControllerType` as shown below, in Fig 4:

```
<ContextPattern>
  <Pattern>SuddenVariations</Pattern>
  <OutParam>MessageThroughput</OutParam>
  <OutParamVal>50</OutParamVal>
  <InParam>Smax</InParam>
  <InParamVal>75</InParamVal>
  <ControllerType>Fuzzy</ControllerType>
</ContextPattern>
```

Fig. 4 Updated Context Pattern Definitions with Controller Type

The Model Tuner will tune the model parameters based on the "`Pattern`" value present in the Context Pattern definition. The model tuner then considers the actual data present in the knowledge based and estimates the model parameters. The estimated model parameters and the Context Pattern definition are forwarded to the Feedback controller block which in turn tunes the control input to the managed system (JMS Provider) there by enabling the self-tuning the system performance.

A library of controllers is part of our solution. The current set consists of a PI controller, Time-Series Controller; Fuzzy controllers. One of these controllers will be used to tune the controller parameters as suggested by Context Pattern.

V. Implementation and Analysis

We implemented the proposed solution in Java and the data is collected from ActiveMQ server [15] by running sample JMS application that we developed. We implemented the PI, Time-Series, Fuzzy controllers which are actuated by a basic control selector, based on the context pattern generated by a primitive version of the data analyzer that is implemented using standard Data Mining techniques. A primitive implementation of Control Selector, Data Analyzer is done to run the experiments.

A. *Modeling*

The model parameters as in Equation (2) are estimated for the JMS Provider with two different data sets and the parameters with least error are used for the simulation based experiments. The Table II shows the model parameters

TABLE III
MODEL PARAMETER ESTIMATION

Data Sets	**Model Parameter Estimation**		
	a	***b***	***Percentage of Error***
Data Set 1	1	0.28	9.12
Data Set 2	0.91	0.12	8.33

B. *Intelligent Controllers*

The Fig 5 below shows the performance evaluation of the message throughput without Controller and with the intelligent controller proposed in this paper. We observe that the message throughput using proposed Controller is better by about 50 % which is a significant improvement in message throughput over the throughput without controller. We can notice that there are spikes where there is a sudden increase of the number of subscribers. The actual message throughput has reduced suddenly in such cases; Fuzzy control is selected to regulate the message throughput. The operating range of message throughput is between 250 and 350. But we can observe that there is a delay in achieving the steady state, again when the number of subscribers is changed significantly, though the controller output is regulated, there is a variation. We are working to address these kinds of issues.

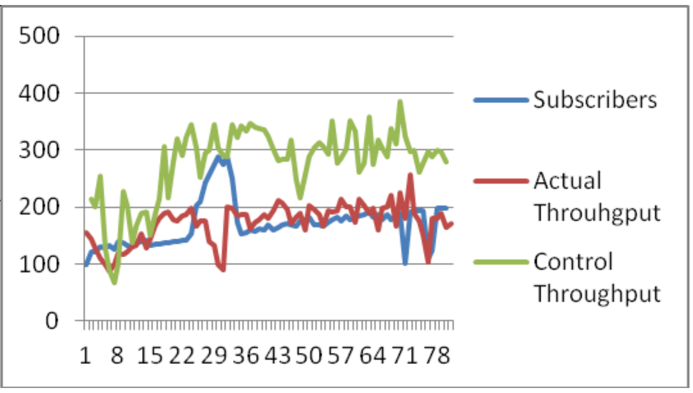

Fig. 5 Performance Evaluation of Message Throughput using Intelligent Control

The Table III below shows the PI controller gains and the corresponding input operating range.

TABLE IIIII
CONTROLLER GAINS

PI-Controller Gain (K_p) Range			
K_P Range	***K_{PI} Range***	***S_{max} Range***	***T_{ref}***
0.12, 0.4, 0.18, 0.34	0.17, 0.22, 0.34, 0.11	20-75	250

VI. Conclusions and Future Work

We have observed that providing hybrid controllers in association with data analyzing mechanisms provides an improved performance management solution for Messaging Servers. We are developing a generic intelligent framework that enables the Application Servers (JEE) [16] to self-manage the performance in an intelligent and automated manner with a minimal manual intervention. The same concepts are being explored to apply in other parts of the JEE Servers such as Web, EJB.

The current solution limits to considering a SISO model where the Message throughput is depends upon the number of subscribers, but it is required to consider the publishers, number of topics or queues present in a single instance of the JMS Provider that determine the message throughput. Additionally, the current experiments are run in simulated environment. We intend to progress on these two limitations and run experiments, validate the results.

References

[1] Message Oriented Middleware (MoM): http://en.wikipedia.org/wiki/Message-oriented_middleware
[2] Matjaz B.Juric, S.Jeelani Basha, Rick Leander, Ramesh Nagappan, "*Professional J2EE EAI*", Shroff Publishers 2005
[3] Ravi Kumar G, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu , "Self-regulating Message Throughput in Enterprise Messaging Servers – A Feedback Control Solution" , *IJACSA, Volume 3, No 1*, Jan2012
[4] JMS: "http://www.oracle.com/technetwork/java/javaee/tech/index.html
[5] JMS Providers: http://en.wikipedia.org/wiki/Java_Message_Service
[6] http://www.precisejava.com/javaperf/j2ee/JMS.htm#JMS111
[7] Bruce Snyder, Dejan Bosanac and Rob Davies, "ActiveMQ In Action", *Dreamtech Press*, 2011
[8] Pradeep Padala, Xiaoyun Zhu, Mustafa Uysal et al. "Adaptive Control of Virtualized Resources in Utility Environments". *In the proceedings of the EuroSys 2007*

[9] Evgeny Dantsin, Thomas Eiter, Georg Gottlob, Andrei Voronkov, "Complexity and expressive power of logic programming", *ACM Computing Surveys* 2003

[10] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy, Dr.A.Vinaya Babu and Raj N. Marndi, "A FEEDBACK CONTROL SOLUTION IN IMPROVING DATABASE DRIVER CACHING", *IJEST, Vol 3, No 7*, July 2011

[11] ARMA: http://en.wikipedia.org/wiki/Autoregressive_moving_average_model

[12] SISO: http://en.wikipedia.org/wiki/Single-Input_and_Single-Output

[13] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy, Dr.A.Vinaya Babu, "SELF-MANAGING THE PERFORMANCE OF DISTRIBUTED COMPUTING SYSTEMS – AN EXPERT CONTROL SYSTEM SOLUTION", accepted in ICADC 2012, Bangalore (unpublished)

[14] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy, Dr.A.Vinaya Babu, "Intelligent Enterprise Application Servers – A Vision for self-managing performance" – submitted to ITC2012, Bangalore (unpublished)

[15] ActiveMQ: http://activemq.apache.org/

[16] JEE Specification : http://www.oracle.com/technetwork/java/javaee/tech/in dex.html