Volume 2, Issue 11, November 2012 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Throughput Improvement in Distributed Systems – An Investigation on Java Messaging Servers using Adaptive Control

G Ravi Kumar*
HP Bangalore,
Research Scholar, JNTUH Hyderabad,

Dr.C.Muthusamy
Yahoo
Bangalore, India

Dr.A.Vinaya Babu
JNTUH Coll of Engineering,
JNTUH, Hyderabad, India

Abstract— Messaging Servers are one of the commonly used technologies for Distributed Systems. It is important to manage the performance of Messaging Servers with ability to handle different workload dynamics. There are different solutions to self-manage the performance of Messaging Servers, We investigated in applying control system based solutions to self-manage and regulate the throughput in Messaging Servers. In the previous experiments we adjusted the maximum number of subscribers to self-manage the throughput using intelligent control based on Fuzzy mechanisms and employing P, PI Control. In order to regulate the performance we constructed a Context object that enables in providing pro-active control. In this paper we improved our previous solutions further where an adaptive control solution is proposed that extends the Context Object scope. Additionally the stability of the proposed control is analysed using scilab.

Keywords— Enterprise Messaging Servers, Message Throughput, Control Systems, Adaptive Control, Distributed Systems

I. INTRODUCTION

Distributed Systems development is simplified with Enterprise Messaging ability of asynchronous communication [1]. Enterprise Messaging is used in many critical enterprise and cloud based applications and services. Integration of enterprise applications is much simplified using messaging servers which are based on messaging oriented Middleware [2]. It is important to ensure a high performance of Messaging Servers which play a significant role in building mission critical applications and enterprise level application integration [3]. There is variety of solutions in dealing with the performance management of Messaging Servers reactively. But it is desirable to provide pro-active control mechanisms also that ensure the self-managing performance of messaging servers. Control Systems theory is a successful theory in building such self-managing computing systems [4]. There was a limited investigation in applying control systems in Enterprise Messaging; which motivated us to explore in applying control system concepts in self-managing the throughput in Enterprise messaging systems. We observed the improvement in the analytical throughput in our previous experiments and investigation with JMS Providers for our experiments [5]. This paper is an extension of our previous work on investigating control systems applicability in Distributed systems with a specific case study on Java Messaging servers [6].

II. PROBLEM AND RELATED WORK

Performance management is always a challenging problem for the enterprise class software systems [7]. This becomes more critical when the software systems are part of the mission critical applications and services. Enterprise Messaging systems is a very popular mechanisms deployed in integrating large scale enterprise applications and services [3]. There are various mechanisms in handling their performance challenges. The most obvious option is to employ administrators to configure and tune the system parameters [8]. As an advanced option conditional programming is also followed to handle this issue [9]. There are various bottlenecks associated with these procedures as discussed in [6]. Another important dimension is the missing aspect of pro-active control mechanisms in the messaging servers that can regulate the input parameters such that the desired message throughput is being maintained at any given instant of time [10]. Control System theory has the built-in capabilities in providing the pro-active control capabilities to any computing system [11]. In our previous investigations we verified the improvement of message throughput using classic control theory by adopting P-controller [10]. We extended the work by implementing intelligent control mechanisms that work as adaptive controllers in conjunction with PI-Controller to regulate the message throughput [6]. There is a context pattern object that is constructed which consists of the predicted load pattern [12], and an appropriate controller required for such a pattern is being determined using the rules defined. Additionally these experiments have been run by modifying the number of subscribers linearly and with some spikes. In this paper we have extended the solution further where the Context pattern object scope is enhanced, adaptive control capabilities are extended, the experiments are run with a distinct maximum number of subscribers. This helps in determining the optimal maximum number of subscribers for a given future or current dynamics on the messaging server rather than providing a range of allowed maximum number of subscribers. The stability of the closed loop control system with our solution is being verified using scilab [13] and observed encouraging results.

© 2012, IJARCSSE All Rights Reserved

III. MODELLING

According to Control Systems theory in order to control the output parameter of the system to be managed, it is required to model it mathematically [14]. ARX, ARMA [15] are some of such models that are commonly adopted to represent the system. We have chosen to use ARMA models to mathematically model JMS Provider

A. Mathematical Model

Considering the ARMA model, any linear system can be represented using the Equation (1). Though there will be non-linearities in any computing system, but they can be approximated to linear system representation [16].

The equation (1) below represents the SISO model of controlled system.

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

Applying the Equation (1) to JMS Provider considering MISO model, the JMS Provider can be represented using the equation (2) below.

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

Though we have shown a MISO model, most of our experiments are conducted with SISO model [17] considering only subscribers as the input that determine the message throughput, the number of publishers and the message sizes published are kept constant

$$T(t + 1) = aT(t) + bS_{max}(t) \quad (3)$$

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) = message Throughput output in the next step

B. Parameter Estimation and Reference Throughput

Control system design for Throughput improvement requires the model coefficients also known as model parameters to be estimated. The throughput is predicted using the Equation (3) above and the parameters are estimated by considering a cost function such that error calculated between actual throughput and predicted throughput is minimum. There are different such cost functions where there are different error types that need to be minimized such as RMSE [18], MAE [19]. In this paper RMSE is considered to estimate the model coefficients. The modelling parameter estimation implementation is run for different values of S_{max} upto 10000. The average throughput for each such distinct S_{max} is considered to determine the reference throughput.

IV. ADAPTIVE CONTROL SOLUTION

An adaptive Control solution is proposed to self-manage the message throughput of JMS Providers. In this section the architecture, context object are discussed.

A. Architecture

The Fig 1 below shows the architecture for self-managing the message throughput. It primarily classified into feedback and adaptive control blocks. The control solution proposed has various flavours of adaptive control such as Gain Scheduling, Stochastic control, dual control [20] to tune the server for current workload dynamics and also for future conditions. The proposed control solution has some abilities of MIAC [20] where the JMS Provider models are dynamically adjusted. The advantage of our novel solution is that any non-linear nature of the workloads is addressed. The solution has more than one controller in the architecture and the most optimal controller is identified suitable for the workload variations at any given instant of time. There are two control blocks in the architecture Adaptive and Feedback. The Adaptive control block predicts future message throughput based on the current data, creates a context pattern object, and dynamically selects the controller based on the context pattern object, computes the controller gains. The full context pattern object is passed to the feedback control block. The feedback controller uses the context pattern object contents and applies the new gains computed to regulate the throughput.

The Feedback control block consists of P, PI, PID [21] and Fuzzy controllers [22]. One of these controllers will be invoked based on the controller mentioned in the Context Pattern object. In the case of P, PI Controllers, new gains to be applied for each loop are present in the Context Pattern. The Derivative (D) Controller is sensitive to stochastic nature of the computing environments. PI Controller is suitable to bring the server with less settling time and steady state error. The input S_{max} is adjusted such that message throughput is in the desired range. Though there are standard ways to do PI control design such as pole placement, root-locus, LQR [23], we implemented the PI Controller using a cost function approach which keeps RMSE as low as possible and meeting the JMS Providers Throughput [10]. A similar technique is adopted in self-managing the cache hit ratio of JDBC drivers [24] and our previous experiments on JMS Providers [6].

© 2012, IJARCSSE All Rights Reserved

The Fuzzy control will be tuning the maximum subscribers based on the information from the Context Pattern object. The fuzzy controller will be chosen in the adaptive control block, when there are predictions about having a highly varying behaviour of the throughput or there are sudden variations in the load affecting the throughput. The fuzzy control will identify the controller gains suitable for the kind of variations pattern identified. The typical feedback controls tune and regulate the system to be controlled based on the current output and previous input. But in our solution we make the feedback control work in conjunction with the adaptive control block, this enables self-regulating the JMS Provider performance and keep the optimum maximum subscribers input to meet the desired Message throughput. The controller design uses the classic schemes such as root locus, pole placement to meet the SASO properties [16]. But in our solution based on the predicted future throughput values, the controller type is selected dynamically; the controller gains are calculated considering the cost function which is RMSE and the Reference throughput (T_{ref}).

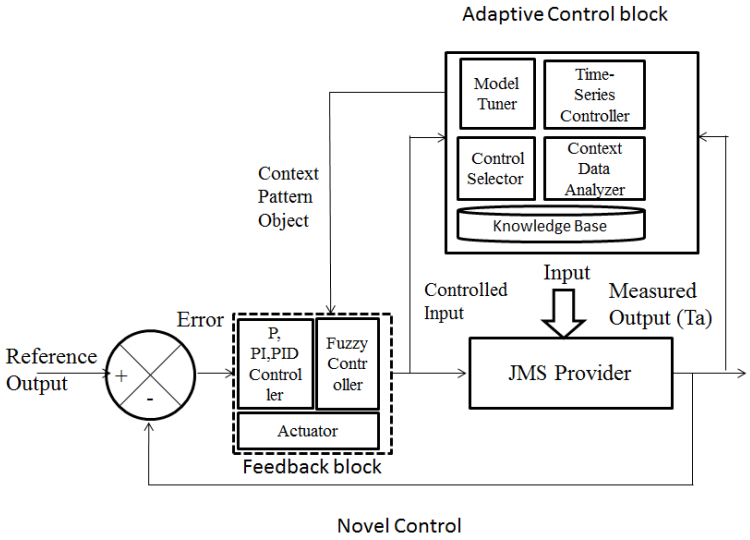

Fig. 1 Adaptive Control Architecture

B. Context Object

A 2-level Prediction Algorithm [25] is extended that constructs the context pattern object using Context data analyser. The object gets constructed for every control loop. The constructed context pattern objects are stored as rules into knowledge base. The time-series and fuzzy controllers are used to predict the future values for the message throughput. The choice of a controller is determined based on the data pattern variation as discussed in [6]. The values of PID controller gains are calculated for the cost function to keep RMSE low and also meet the reference throughput.

The Fig 2 below shows a sample Context Pattern Object

```
<ContextPatternObject>
  <pattern>Linear Increase</pattern>
  <outParam>MessageThroughput</outParam>
  <outParamPredVal>350</outParamPredVal>
  <inParam>Smax</inParam>
  <inParamVal>175</inParamVal>
  <Kp>0.5</Kp>
  <KI>0.2</KI>
  <KD>0.6</KD>
  <ControllerType>PID</ControllerType>
  <tuneNow>false</tuneNow>
  <numPeriodsAhead>5</numPeriodsAhead>
</ContextPatternObject>
```

Fig. 2 Sample Context Object

V. IMPLEMENTATION AND ANALYSIS

The proposed solution is implemented by modeling the control system by collecting the sample data by running a test JMS application. The same application is run on ActiveMQ server [26] to collect test data. We have run our solution on the test data collected to calculate the analytical controlled throughput. Also, the performance of the analytical controlled throughput is analyzed. The stability analysis on the closed loop unity feedback control system that is discussed in section D "Stability Analysis".

A. Test Setup

© 2012, IJARCSSE All Rights Reserved

The Fig 3. Below shows the test setup deployed to model the JMS Provider. Sample JMS applications are performed that are used to load the messaging server. The JMS Provider statistics such as Throughput are calculated using the JMX MBean APIs [27]. The Throughput in our experiment is measured per unit which is around 50 ms. A single JMS topic is chosen as the broker to run our experiment. A single Publisher is implemented that sends messages with fixed message length to the topic. There is a JMS Subscriber application that creates multiple subscriber threads that subscribe to the topic and get messages. The monitoring data from the ActiveMQ server is collected and throughput is measured. The number of subscriber threads that are concurrently subscribed and the number of times the publisher application sends messages to the topic are set as configurable properties.

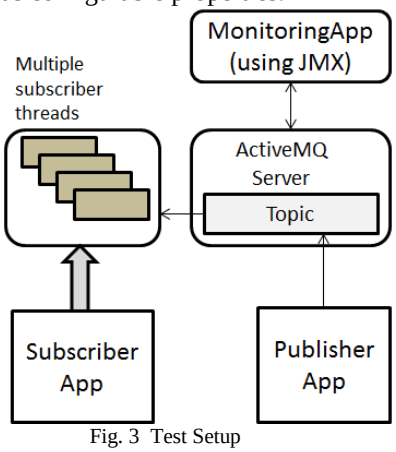

Fig. 3 Test Setup

B. Modeling

The JMS Provider can be modelled as shown in the Equation (2) where the max number of subscribers is one of the important metric that influence the throughput. The model coefficients *a* and *b* are determined using the equation by keeping fixed number of publishers. The experiment is run with different values of S_{max}. The ARMA model is used to predict the throughput one step period ahead. Initially a fixed values of *a* and *b* are considered. The predicted data set is used to optimize the model parameters by minimizing the RMSE values. The Fig 4 below shows optimized modelling data for 10000 subscribers. The Fig 5 is the modelling behaviour for 5000 subscribers

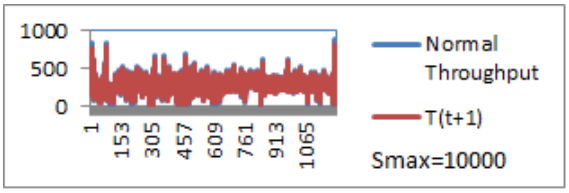

Fig. 4 Model Parameter Estimation – Smax 10000

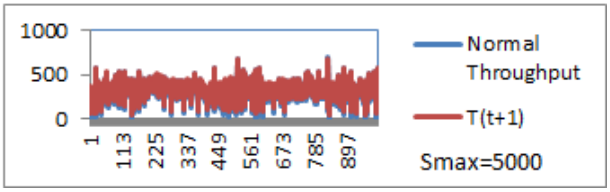

Fig. 5 Model Parameter Estimation – Smax 5000

The Table I below shows the optimized and estimated model parameters for the different subscribers. There are multiple parameters estimated in few instances but the parameters with minimum RMSE are determined as the model coefficients for implementing the control throughput.

TABLE I
MODEL PARAMETER ESTIMATION

S_{max}	*a*	*b*	RMSE
100	0.91	0.31,	8.209,3.872
500	0.92	0.055	6.833
1000	0.92, 0.38	0.023, 0.026, 0.2	9.25
5000	0.91,0.92, 0.91	0.01	1.84, 7.6
10000	0.91,	0.004, 0.008	1.6

C. Performance Analysis

A significant improvement in the analytical throughput is observed using the proposed solution for the different S_{max} values. There is an improvement on the throughput is analysed for the different S_{max} values and on average across the different S_{max} is considered and the performance improvement is around 50%. Additional observation we can see is the improvement in the S_{max} values to support the T_{ref} and improve the throughput with an exception for S_{max} when it is 5000, where our algorithm determines the optimal value of subscribers is 4361 to meet the desired throughput T_{ref}. Our experiments considered PI controller in the feedback block, but these controller gains are dynamically computed during

© 2012, IJARCSSE All Rights Reserved

the run time and populated into the context object. The feedback controllers use these context objects controller gains in every control cycle to adjust S_{max}.

The future throughput patterns are predicted and the controller gains are adjusted dynamically at run time to ensure the T_{ref} value is maintained. There are sudden changes to performance as shown in Fig 9 which are predicted in the adaptive control block and the patterns are identified as "sudden variations" for which the Fuzzy controller is selected in the adaptive block to calculate the controller gains. Based on this pattern, the controller gains are selected from the knowledge base. The following Table II shows the throughput for different S_{max} values and the controller gains, which is tuned is tuned in every control cycle to ensure that the throughput is close to the T_{ref}.

TABLE II
CONTROLLER GAINS AND THROUGHPUT

PID Controller Gains						
K_p Range	**K_{PI} Range**	**Test Smax**	**Controlled Smax**	**Test Throughput**	**Controlled Throughput**	**T_{ref}**
0.6	0.06	100	118	334	399	400
0.5	0.2	500	735	342	493	400
0.5, 0.8, 0.32, 0.28, 0.4	0.2, 0.44, 0.4, 0.12, 0.14, 0.08	1000	1372	288	493	400
0.48	0.22	5000	4361	310	480	500
0.5, 0.7, 0.92, 0.88, 0.78	0.2, 0.4, 0.56, 0.48, 0.6	10000	10255	290	454, 499.70	500

Also, we can observe that these controller gains have different values when Smax is set 10000. The experiments reveal the operating range of the throughput varies for different number of subscribers which suggests to change the value of T_{ref} appropriately.

In Fig 6. when the S_{max} is set to 100, though the normal throughput is varying significantly, the fixed values of controller gains ensured a constant controlled throughput after initial ramp up. A similar pattern can be observed for S_{max} is 500 in Fig 7. except that there is a spike to be handled. When the S_{max} is set to 1000 in Fig 8. a different behaviour is noticed where the pattern identified is "continous variation" for which the PI controller is selected with appropriate variation in the controller gains except for a couple of sudden spikes. In the case of S_{max} 5000 in Fig 9. is a combination of sudden variations and constant behaviour for which the fuzzy controller in adaptive block determined the gains. The S_{max} set to 10000 is closer to a real time behaviour of the server as shown in Fig 10. where there variations at regular intervals except a few spikes in the initial stage before reaching a steady state. The PI controller gains are adjusted in the adaptive block , along with using the fuzzy controller.

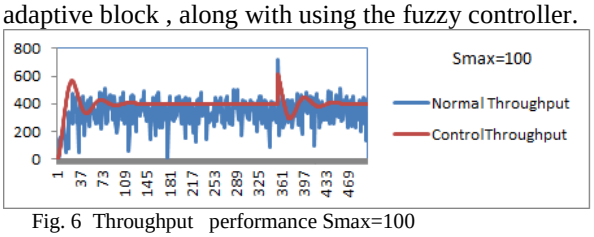

Fig. 6 Throughput performance Smax=100

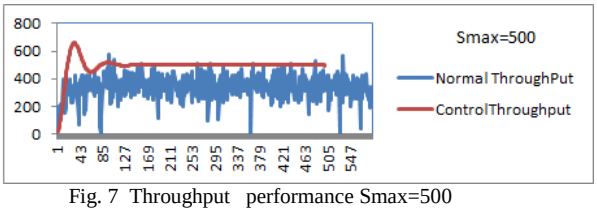

Fig. 7 Throughput performance Smax=500

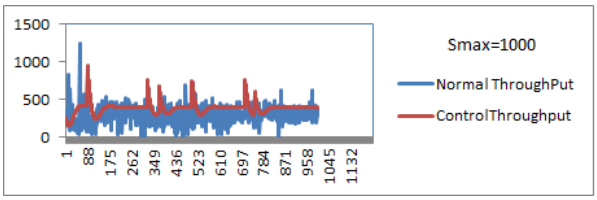

Fig. 8 Throughput performance Smax=1000

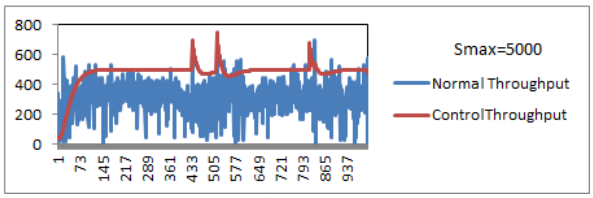

Fig. 9 Throughput performance Smax=5000

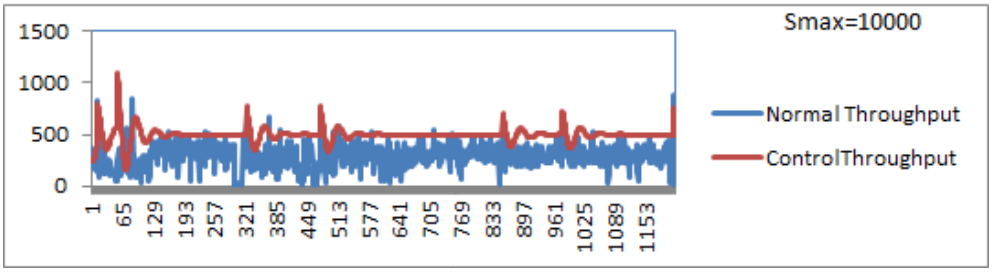

Fig. 10 Throughput performance Smax=10000

© 2012, IJARCSSE All Rights Reserved

The Fig 11 show average measured S_{max} and Fig 12 show the average analytical controlled throughput, there is a distinct difference and improvement in analytical controlled throughput using our solution. Also, there is an increase in the number of maximum subscribers that can be supported and tuned to meet the desired reference throughput (T_{ref}).

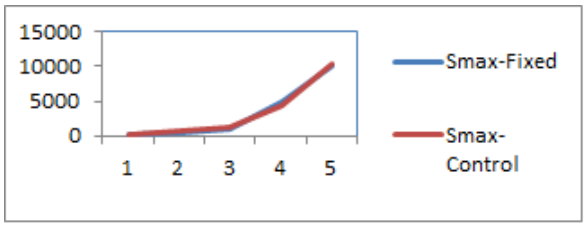

Fig. 11 Smax Variation

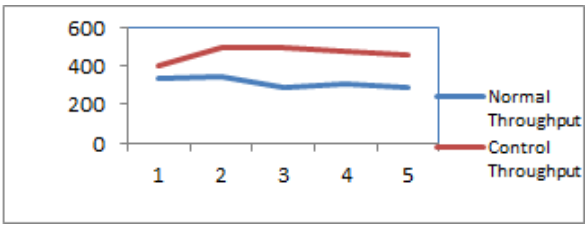

Fig. 12 Average Throughput Variation

Also, we measured the S_{max} that can be supported by allowing some amount of degradation on the throughput, but definitely better than the normal throughput and found interesting results. It is possible to achieve by adjusting the model parameters and the controller gain K_I. The following Fig 13 shows such behaviour when tested for S_{max} with 10000 against its higher value. The Fig 14 shows the improved throughput

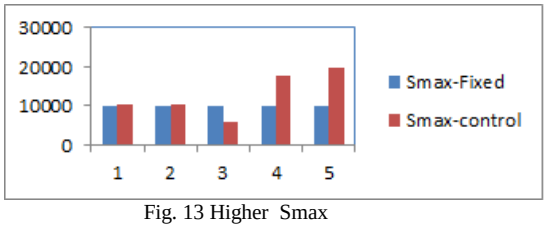

Fig. 13 Higher Smax

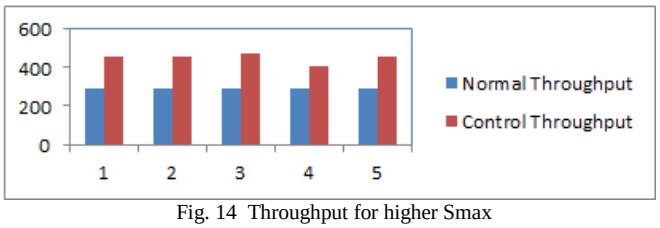

Fig. 14 Throughput for higher Smax

D. Stability Analysis

The stability analysis is also run on the closed loop unity feedback control system, besides the performance analysis. The following Fig 15 shows the scilab polar plot showing the pole of the closed loop feedback system is within the unit circle which indicates the stability. We have verified the stability for S_{max} when set to 10000.

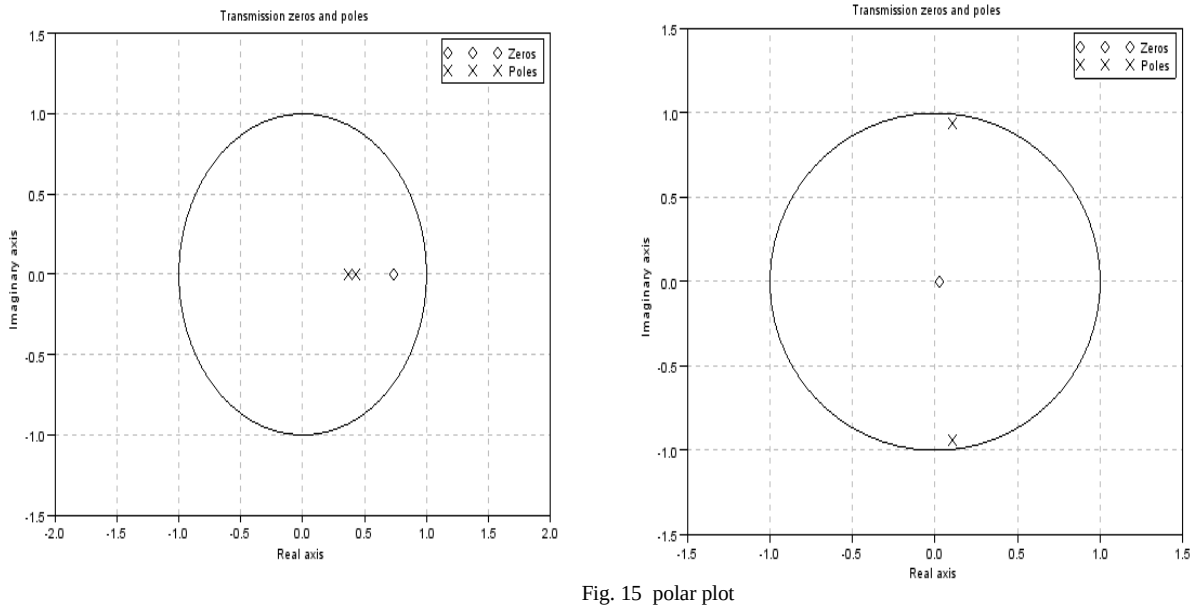

Fig. 15 polar plot

The settling time (*ks*) of the closed loop system is one of the SASO properties besides the stability ranging. The range is between 4.7 and 44 considering for different parameters and control gains, the average is measured as 20.

VI. CONCLUSIONS AND FUTURE WORK

The analysis of performance and stability proves the ability of applying adaptive control capabilities in JMS Provider throughput improvement. The mechanism of proposed solution where the dynamics of the server throughput for future periods are predicted and control selection is done dynamically. The selection of controller, capturing all the details as context objects gives more flexibility in adjusting the maximum number of subscribers that can be allowed at a given instant of time, also ensuring the desired throughput is maintained. In our approach the controller design is dynamically implemented using the cost function optimization approach by minimizing RMSE. We are in the process of verifying the

© 2012, IJARCSSE All Rights Reserved

proposed solution for other Distributed System middleware components such as EJB. Additionally we see possibility in extending this research considering varying number of publishers, topics which makes the JMS Provider as MISO model. We have captured the message processing time in this experiment, though not shown in this paper, it will be an interesting aspect to extend this work further in tuning. The important future work possibility is to apply the solution on a real time JMS Provider as most of our solution is analytical in nature. Another important research extension is to consider the discussed approach as inherent feature within JMS Providers during UML design and development.

REFERENCES

[1] Richard Monson-Haefel and David A.Chappell, "Java Message Service", O'Reilly 2001
[2] Message Oriented Middleware (MoM): http://en.wikipedia.org/wiki/Message-oriented_middleware
[3] Matjaz B.Juric, S.Jeelani Basha, Rick Leander, Ramesh Nagappan, "Professional J2EE EAI", Shroff Publishers 2005
[4] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu , "Control Systems application in Java based Enterprise and Cloud Environments – A Survey" , IJACSA, Volume 2, No 8, August 2011
[5] JMS Providers: http://en.wikipedia.org/wiki/Java_Message_Service
[6] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu , "Throughput Regulation of Messaging Servers – An Intelligent Control Solution", International Journal of Advanced Research in Computer Science and Software Engineering (IJARCSSE) – Vol 2, Issue 4, Apr 2012
[7] Tarek Abdelzaher, Yixin Diao, Joseph L Hellerstein, Chenyang Lu, and Xiaoyun Zhu., "Introduction to Control Theory and Its Applications to Computing Systems", International Conference on Measurement and Modeling of Computer Systems SIGMETRICS"08
[8] http://www.precisejava.com/javaperf/j2ee/JMS.htm#JMS111
[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 and Dr.A.Vinaya Babu , "Self-Regulating Message Throughput in Enterprise Messaging Servers", International Journal of Advanced Computer Science and Applications (IJACSA), *Vol 3, No 12, 2012*
[11] Joseph L. Hellerstein, Yixin Diao, Sujay Parekh, and Dawn Tilbury Feedback Control of Computing Systems, John Wiley 2004
[12] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu , "Self-Managing Performance in Application Servers – Modelling and Data Architecture**" –** ICCSIT, May 2012, Bangalore, pp 101-106
[13] Scilab: http://www.scilab.org/support/documentation
[14] mathematical model: http://en.wikipedia.org/wiki/Mathematical_model
[15] ARMA: http://en.wikipedia.org/wiki/Autoregressive_moving_average_model
[16] Hellerstein, J.L. ; Parekh, S. ; Griffith, R. ; Kaiser, G.E. ; Phung, D., "A control theory foundation for self-managing computing systems", IEEE Journal on Selected Areas in Communications, Volume 23, Number 12, December 2005
[17] SISO Model: "SISO: http://en.wikipedia.org/wiki/Single-Input_and_Single-Output"
[18] RMSE:http://en.wikipedia.org/wiki/Root-mean-square_deviation
[19] MAE" http://en.wikipedia.org/wiki/Mean_absolute_error
[20] Karl J.Astrom and Bjorn Wittenmark, "Adaptive Control", Pearson Education, 2009
[21] B.C.Kuo, "Automatic Control Systems", PHI 2010, New Delhi India
[22] Fuzzy control: http://en.wikipedia.org/wiki/Fuzzy_control_system
[23] LQR: http://en.wikipedia.org/wiki/Linear-quadratic_regulator
[24] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu ,"A FEEDBACK CONTROL SOLUTION IN IMPROVING DATABASE DRIVER CACHING" , International Journal of Engineering Science and Technology, Vol3, No 7, Jul 2011
[25] Ravi Kumar Gullapalli, Dr.Chelliah Muthusamy and Dr.A.Vinaya Babu , "Intelligent Application Servers – A Vision for self-managing performance**",** Third International Conference on Recent Trends in Information, Telecommunication and Computing, Bangalore, Aug 2012, Springer LNEE
[26] ActiveMQ: http://activemq.apache.org/
[27] Bruce Snyder, Dejan Bosanac and Rob Davies, "ActiveMQ In Action", Dreamtech Press, 2011

© 2012, IJARCSSE All Rights Reserved