Some Aspects of the Quality of Service Support Used in Multi Agent Systems on Middleware-Based Distributed Messaging

Jose-Luis Poza-Luján, Raúl Simarro-Fernández, Juan-Luis Posadas-Yagüe and José-Enrique Simó-Ten

University Institute of Control Systems and Industrial Computing (ai2). Universitat Politècnica de València. Camino de vera, s/n. 46022 Valencia (Spain)
{jopolu, rausifer, jposadas, jsimo}@ai2.upv.es

Abstract Messaging systems are widely used in distributed systems to hide the details of the communications mechanism to the multi agents systems. However, the Quality of Service is treated in different way depending on the messaging system used. This paper presents a review and further analysis of the quality of service treatment in the mainly messaging systems used in distributed multi agent systems. The review covers the issues related to the purpose of the functions provided and the scope of the quality of service offered by every messaging system. We propose ontology for classifying and decide which parameters are relevant to the user. The results of the analysis and the ontology can be used to select the most suitable messaging system to distributed multi agent architecture and to establish the quality of service requirements in a distributed system.

1 Introduction

To make a transparent connection in a distributed system is necessary to hide the details of the communications system to the applications. To make this work, appears the concept of middleware. The scope of middleware is very extensive. Therefore, middleware is commonly defined as an intermediate layer between the application and the communications system that facilitates all aspects related with the connection [1]. The distributed messaging systems (DMS) paradigm has become increasingly popular in recent years to implement the middleware layer. There are a lot of middleware systems based on this paradigm; mainly due to the excellent adaptation of the DMS to provide support to the large number of distributed intelligent multi-agent systems (MAS) architectures.

There are many proposed architectures to cover the middleware in a MAS or in a DMS. From the most used architectures, the proposal of the Foundation for Intelligent Physical Agents (FIPA) [2] is the closest to the MAS, Java Message Service (JMS) [3] and Common Object Request Broker Architecture (CORBA) [4] are considered architectures used in MAS but also middleware systems. Finally Data Distribution Service (DDS) [5] is considered mainly a middleware.

Communications in MAS are particularly important; so that, the Quality of Service (QoS) is one aspect that middleware must cover. Because of this, the management of the QoS in a middleware is one of the main objectives in a MAS system.

This paper provides that review of systems above cited. The review focuses on the QoS support offered in each system and emphasizes the consequences of using either system, as well as the features needed in future proposals.

The paper is organized as follow. Section two describes the main features of the systems considered and discusses the role of each system. Section three contextualizes the systems in the communication process between agents and organizes the QoS features. Section four discusses the results of the previous section. Finally the paper reports the conclusions of the study.

2 Messaging Systems used in Multi Agent Systems

The four systems analyzed (Table 1) are the most supported by standardization organizations and commercial companies. FIPA is a proposal of the organization with the same name. Currently the Institute of Electrical and Electronics Engineers (IEEE) continues the standard. FIPA is used in several agent platforms as JADE [6], JACK [7], ZEUS [8] or Grasshopper [9]. JMS is a part of the Java Platform Enterprise Edition (J2EE) [10] and is used by well known commercial products as ActiveMQ [11] or MQSeries [12]. CORBA is a standard defined by the Object Management Group (OMG) [13] an organization aimed at setting standards for distributed systems and model-based standards. CORBA is widely used for academic and commercial purposes. DDS is a specification of a middleware for distributed systems. The aim of DDS is standardize the programming model for distributed systems. Like CORBA, DDS is an OMG specification but DDS is Real Time oriented and allows the user to specify QoS parameters by means a set of policies. DDS is used in critical systems as aerospace or military products and is supported by robust commercial systems as RTI [14] or OpenSplice [15]

Table 1. Systems compared with the corresponding standardization organization, the main scope or use of the system (agents, middleware or both) and the year of the first version.

System	Standardization organism	Scope	1st version
CORBA	OMG	Middleware → Agents architecture	1991
FIPA	IEEE	Agents architecture	1996
JMS	Sun (Oracle)	Agents architecture → Middleware	2001
DDS	OMG	Middleware	2003

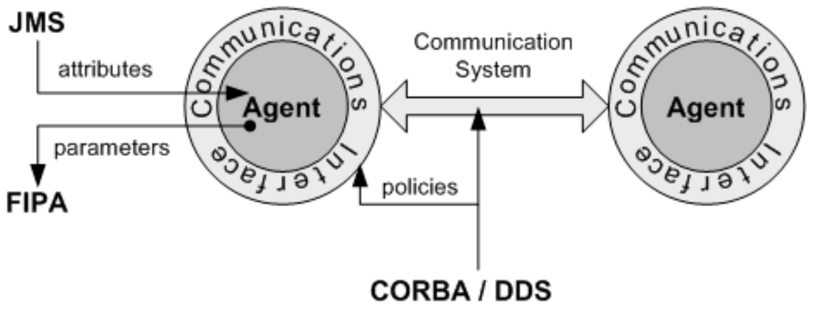

Fig. 1. Location in the MAS communications model of each system analyzed.

CORBA appears in 1991, focused on Object-Oriented Programming (OOP), the programming paradigm used in that time. When the Agent-Oriented Programming model is consolidated appears FIPA. Although is not his main role, CORBA is used as middleware of a great amount of MAS systems, coinciding with the appearance of the agents model. When the concept of middleware is extended, appears JMS. The distributed messaging paradigm is used by JMS and his paradigm is also employed by the DDS model. DDS is the latest model appears and is intentionally a model to be used as middleware.

Figure 1 organizes the scope in the communications process between agents of each system analyzed. JMS is designed to provide to the component the attributes to control the QoS, while FIPA is more focused on the parameters of the agent. CORBA adds the control of the QoS by means the policies, even if the attributes on which it works are more oriented to the communication connection. Finally DCPS is focused on managing the communication of all the distributed system.

3 Quality of Service supported in messaging systems

QoS is a concept that defines a set of parameters to evaluate a service offered. In the field of control architectures, there are many definitions of QoS. From the viewpoint of processing, QoS represents the set of both: quantitative and qualitative characteristics of a distributed system needed to achieve the functionality required by an application [16]. From the communication viewpoint, QoS is defined as all the requirements that a network must meet to message flow transport [17]. In the communication layer, QoS provides temporal parameters like messages delay or easy message flow control like congestion control.

In FIPA, QoS is considered optional, so it is the responsibility of the programmer develops the functions to obtain and manage the QoS parameters. However, [18] proposes 14 relevant QoS parameters. The parameters proposed are similar to the traditionally used in communication systems without taking into account aspects like the message flow or the metadata interchange. Due to FIPA specifies parameters; the model only offers a static idea of the communications.

JMS does not provide QoS parameters. However, different implementations based on the JMS model uses the concept of QoS attributes to manage the communications specified in [19]. In JMS is possible to control 6 attributes, principally specialized in message flow and the temporal characteristics. Nevertheless, JMS don't provide attributes to the handling of the communications faults.

The concept of attributes, offers a dynamic vision of the communications, because some aspects of the message interchange, like the deadline required by the receiver or the deadline that the service can provide, can be configured by the user.

Table 2. QoS areas and parameters included in the systems analyzed.

QoS area	Parameter / Policy	FIPA	JMS	CORBA	DDS
Connection management	Connection delay	Yes			
	Connection errors or liveliness	Yes			Yes
	Connection mode		Yes		
	Connection status	Yes			
	Reconnection			Yes	
Error handling	Error rate / Reliability	Yes			Yes
	Mean up time	Yes			
Message flow	Delivery order			Yes	Yes
	Max hops			Yes	
	Persistence		Yes		Yes
	Priority		Yes	Yes	Yes
	Routing			Yes	
	Synchronization			Yes	
	Topology				Yes
	Transaction type		Yes		
Metadata	Metadata				Yes
	Presentation				Yes
Performance	Bandwidth	Yes			
	Resource limits				Yes
	Throughput	Yes			
Time management	Delay	Yes			
	Delivery mode / Durability		Yes		Yes
	Lifespan				Yes
	Round trip time / Latency	Yes			Yes
	Temporal filter				Yes
	Timeout / Deadline		Yes	Yes	Yes
	Time to live		Yes		
	Timestamp		Yes		

CORBA introduces the concept of QoS policy [20] that works in a communications quality environment. CORBA specifies 13 different policies. The scope of CORBA policies is very wide including policies to manage the message routing. Like CORBA, DDS uses the concept of QoS policy, but DDS defines 22 different QoS policies, covering almost all aspects of the communications. The use of policies by CORBA and DDS offers both visions, static and dynamic, of the communications. A QoS policy can be viewed as a function that returns the state of the communications. Moreover, the QoS policy can be viewed as a function that allows the user to change some aspects of the message interchange.

The middleware isolates the components of the distributed system in time, space and message flow, accordingly, the QoS parameters or policies related with the middleware must contemplate mainly this areas. Table 2 organizes the parameters or policies in the main different QoS areas. Areas are based on the fields covered usually by middleware systems. Detailed definitions of each parameter can be located in the references mentioned above. In a few words, connection management refers to components aspects. Error handling is related with the communications failures. Message flow, performance and time management are classical areas in middleware. Finally, metadata is the additional information that the message syntax can't cover.

4 Analysis and proposals

As expected, the main areas covered by the QoS parameters are time and message flow management. FIPA focuses on communications aspects as the connection management or the communications performance. JMS is centred on the message flow and time management parameters. CORBA works mainly with message flow, and DDS cover all aspects analyzed.

The parameters most used are priority and deadline. With the priority is possible provide a minimal message flow control in a communications system. In the same way, the deadline provides the minimal time control to messages. These two parameters are the minimum requirements to define the QoS in a communications system. Therefore, in real-time systems is common to find only priority and deadline as QoS parameters.

To increase the QoS support is necessary to provide more QoS parameters, especially to manage the connection among the distributed components. As the number and type of QoS parameters is increased, the complexity of the administration of the system, especially in the middleware, grows. A large number of parameters may be difficult to implement and use the QoS, so the concept of *QoS policy* can help to user with to program the communications with a set o performance constraints.

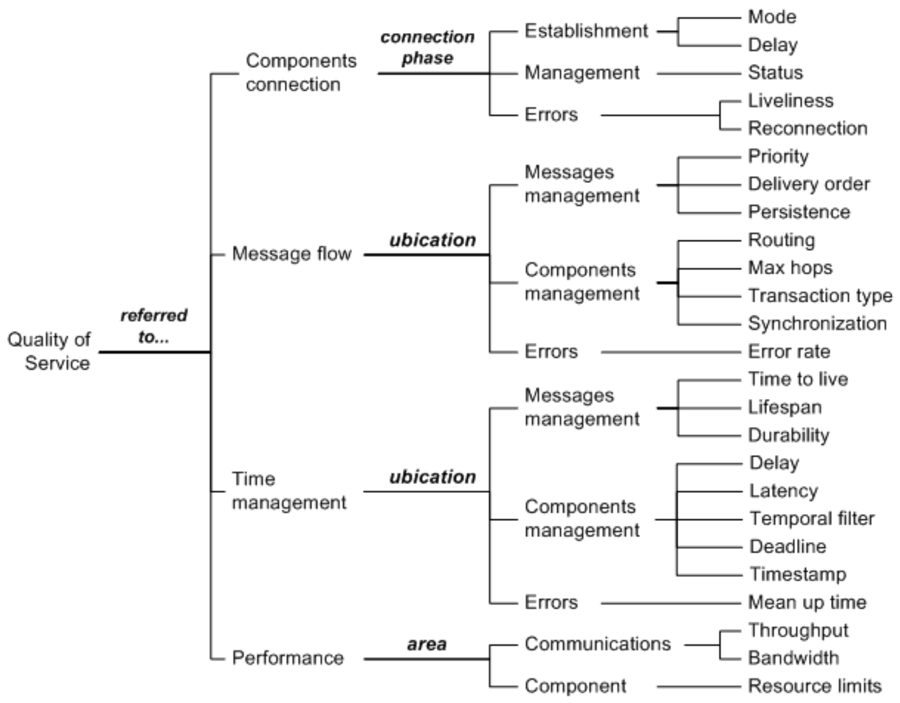

Fig. 2. QoS ontology, where the parameters are organized in function of the system requirements.

It is difficult to determinate the most important or necessary QoS parameters to implement in a distributed MAS; so that, it is necessary organize the parameters. Figure 2 shows an ontology that organizes the parameters in the areas and fields where the QoS parameter is applied. With this ontology, is possible to infer from the type of QoS required which type of QoS parameters are necessary to provide.

The QoS parameters dedicated to manage the components connections, as connection status or liveliness, usually are responsibility of the communications protocol. The performance parameters (throughput, bandwidth or resources management) are highly recommended in all distributed system. Priority, durability, synchronization and deadline, guarantee a minimum QoS in real-time based distributed systems.

The ontology can be organized as a table where the rows describe the main QoS areas (message flow and time management) and columns determine the object to manage (messages or components). Besides, for each QoS area, some parameters can be classified as required while others are only recommender. The table 3 shows the organization described above.

Table 3. QoS parameters recommendations

QoS area	Type of availability	Message management	Component management
Message flow	Required	Priority	Synchronization
	Recommended	Delivery order, Persistence	Transaction type, Routing, Max Hops
Time management	Required	Durability	Deadline
	Recommended	Time to live, Lifespan.	Timestamp, Delay, Latency, Temporal filter

5 Conclusions

On distributed systems, is possible to cover a minimum of the QoS requirements. Depending on the implementation of the system will be more appropriate to use one of the system described in this paper. FIPA covers the requirements of the connection on communications systems. In FIPA, other QoS aspects and parameters are the responsibility of the agents. JMS and CORBA provide more QoS parameters than FIPA, JMS is focused on the message flow management, and CORBA offers more parameters to time management, although CORBA diversifies more his parameters.

Finally, DDS covers all aspects of the QoS parameters, except connection management. DDS is the latest standard proposed based on publish-subscribe paradigm, so has taken into account the relevance of the QoS management in the distributed systems. As the complexity of the distributed system grows, the QoS in the middleware is more necessary.

At the time of implementing the QoS in a distributed system, it is necessary to determine the area of communications to manage: the message flow or the time issues. The ontology presented, in combination with the table 3, can help to designed to choose the most appropriate distributed messaging based middleware. To manage a great amount of QoS policies, is recommended use the concept of policies.

From the analysis offered in this paper, we can deduce that the QoS support offered by the middleware to the MAS it is increasingly necessary to ensure coherence between the agent communications.

Acknowledgments The study described in this paper is a part of the coordinated project COBAMI: Mission-based Hierarchical Control. Education and Science Department, Spanish Government. CICYT: MICINN: DPI2011-28507-C02-01/02

References

1. Gaddah A., and Kunz, T. A survey of middleware paradigms for mobile computing. Technical Report SCE-03-16. Carleton University Systems and Computing Engineering. (2003)
2. Foundation for Intelligent Physical Agents. Website : http://www.fipa.org/
3. Java Message Service Specification. Website : http://java.sun.com/products/jms/docs.html
4. Common Object Request Broker Architecture. Website: http://www.corba.org/
5. Data Distribution Service. Website: http://portals.omg.org/dds/
6. Java Agent DEvelopment Framework. Website: http://jade.tilab.com/
7. Agent Oriented Software Pty Ltd. (1999) "JACK Intelligent Agents: User Guide".
8. H. Nwana, D. Ndumu, L. Lee, and J. Collis. ZEUS: A tool-kit for building distributed multi-agent systems. Applied Artifical Intelligence Journal, 13(1):129–186, 1999.
9. M. K. Perdikeas, F. G. Chatzipapadopoulos, I. S. Venieris, G. Marino: 'Mobile Agent Standards and Available Platforms', Computer Networks Journal, Special Issue on 'Mobile Agents in Intelligent Networks and Mobile Communication Systems', ELSEVIER Publisher, Netherlands, vol. 31, issue 10 (1999)
10. Perrone, Paul J.; Chaganti, Krishna (2003). J2EE Developer's Handbook. Indianapolis, Indiana: Sam's Publishing.
11. Apache ActiveMQ. Website: http://activemq.apache.org/
12. IBM WebSphere MQSeries. Website: http://mqseries.net/
13. Object Management Group. Website: http://www.omg.org/
14. RTI Data Distribution Service. RTI corp.. Website: http://www.rti.com/
15. OpenSplice DDS. PrismTech Ltd. Website: http://www.prismtech.com
16. Vogel, A., Kerherve, B., von Bochmann, G. and Gecsei, J. Distributed Multimedia and QoS: A Survey. IEEE Multimedia.Vol.2, No. 2, pp. 10-19. (1995)
17. Crawley, E., Nair, R., Rajagopalan, B. RFC 2386: A Framework for QoS-based Routing in the Internet. IETF Internet Draft. pp. 1-37. (1998)
18. Foundation for Intelligent Physical Agents. FIPA Quality of Service Ontology Specification. Doc: SC00094A. (2002)
19. Sun Microsystems, Inc. Java(TM) Message Service Specification Final Release 1.1. (2002)
20. Object Management Group (OMG). The Common Object Request Broker Architecture and Specification. CORBA 2.4.2. (2001)