Document downloaded from:

http://hdl.handle.net/10251/69257

This paper must be cited as:

Poza-Lujan, J.; Posadas-Yagüe, J.; Simó Ten, JE. (2010). Multi-Agent Architecture with Support to Quality of Service and Quality of Control. En Intelligent Data Engineering and Automated Learning – IDEAL 2010. Springer Verlag (Germany). 137-144. doi:10.1007/978-3-642-15381-5_17.

The final publication is available at

http://link.springer.com/chapter/10.1007/978-3-642-15381-5_17

Copyright Springer Verlag (Germany)

Additional Information

The final publication is available at Springer via http://dx.doi.org/10.1007/978-3-642-15381-5_17

Multi-Agent Architecture with Support to Quality of Service and Quality of Control

Jose-Luis Poza-Luján, Juan-Luis Posadas-Yagüe, Jose-Enrique Simó-Ten

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

Abstract. Multi Agent Systems (MAS) are one of the most suitable frameworks for the implementation of intelligent distributed control system. Agents provide suitable flexibility to give support to implied heterogeneity in cyber-physical systems. Quality of Service (QoS) and Quality of Control (QoC) parameters are commonly utilized to evaluate the efficiency of the communications and the control loop. Agents can use the quality measures to take a wide range of decisions, like suitable placement on the control node or to change the workload to save energy. This article describes the architecture of a multi agent system that provides support to QoS and QoC parameters to optimize de system. The architecture uses a Publish-Subscriber model, based on Data Distribution Service (DDS) to send the control messages. Due to the nature of the Publish-Subscribe model, the architecture is suitable to implement event-based control (EBC) systems. The architecture has been called FSACtrl.

Key words: Multi Agent System (MAS). Quality of Service (QoS). Quality of Control (QoC). Publish Subscriber model. Event Based Control (EBC)

1 Introduction

The optimal control of distributed systems has changed from the systems based on bus-oriented communications to the large industrial systems based on computer networks. The current trend joins all aspects of distributed systems with intelligent control in concept known as cyber-physical systems [1].

Systems must be optimum to carry out of the objectives fixed. For optimize a system is necessary to have the suitable information about which are the features that have more influence on throughput. The information about communication performance is known as quality of service (QoS). Information about the compliment of the control requirements is included in the concept of quality of control (QoC). To manage system performance, the architecture must provide to agents all necessary information to build their own quality indicators. One interesting question is, if the quality indicators can be used to take the usual decisions used in the distributed systems, for example move or clone an agent between two control nodes.

There are a lot of agent-based protocols, middleware and architectures. The treatment of the QoS is different depending on the standard used. The Common Object Request Broker Architecture (CORBA) defines the QoS by means the concept of messaging policy. CORBA defines 14 policies to cover the basic time, order and routing aspects [2]. The Foundation for Intelligent Physical Agents (FIPA) defines 14 QoS policies mainly in terms of speed and reliability [3]. The Data Distribution Service model DDS [4] specification proposes 22 different QoS policies that cover all aspects of communications management: message temporal aspects, data flow and metadata.

To use the different points of view of the QoS with the system's QoC is necessary to have a uniform method to obtain the necessary QoS parameters. A multi-agent architecture, called Frame-Sensor-Adapter with Control support (FSACtrl), has been developed to provide to control components the QoS parameters. This architecture is the evolution of the model FSA, widely tested on mobile robot, and home automation systems [5].

The communication system on FSACtrl is based on the DDS model and can supply the DDS QoS policies and, for extension, the FIPA and CORBA policies. In FSACtrl, the control algorithms are implemented on the control components, this control components are very similar than the communications components. FSACtrl is characterized by its simplicity in the implementation of components, the efficient management of QoS and QoC parameters and the minimization of protocol operations. It makes possible to distribute the components in an efficient way.

The rest of the article has been organized as follows. The second section introduces the theoretical concepts of communications that are the base of the architecture shown. The third section describes basic concepts about FSACtrl architecture. The fourth section shows the main idea of the article, based on the cycle of QoS and QoC. Finally, concluding remarks about architecture are shown and possible research lines that emerges from the combination of QoS and QoC.

2 Theoretical concepts

2.1 The DDS model

There are different paradigms of communication with support to quality of service, among them publish-subscribe model is one of the most suitable [6] due that isolates publishers of subscribers, enabling a QoS negotiation based on the information topics. The agents only need to know the topics to send or receive the information, without knowing the current location of the other agents.

Object Management Group (OMG) has proposed DDS, based on the paradigm of publish-subscribe with support to QoS. Data Distribution Service (DDS) provides a platform independent model that is aimed to real-time distributed systems. DDS is based on publish-subscribe communications paradigm. Publish-subscribe components connect information producers (publishers) and consumers (subscribers) and isolate the publishers and the subscribers in time, space and message flow [7].

DDS specifies two areas: Data-Centric Publish-Subscribe (DCPS), which is responsible for data distribution, and Data Local Reconstruction Layer (DLRL) which is responsible for adjusting the data to local level of applications. DLRL area is optional due to the DCPS components can work directly with the control objects without data translations.

When a producer (component, agent or application) wants to publish some information, should write it in a Topic by means of a component called Data Writer which is managed by another component called Publisher. Both components, Data Writer and Publisher, are included in another component called Domain Participant. On the other hand, a Topic cans delivery messages to both components: Data Readers and Listeners by means of a Subscriber. Data Reader provides the messages when the application requires. Instead of a Listened sends the messages without waiting for the application.

Quality of Service is defined as the collective effect of service performance, which determines the degree of satisfaction of a user of the service [8]. The concept of QoS is used to measure all relevant characteristics of a system. Generally, QoS is associated with a set of measurable parameters. In DDS model, QoS policy can be defined as the dynamic management of the QoS parameters whit negotiated values. For example, by means the "Deadline" policy, that determines the maximum time for the message arrival, and the "Time-Based-Filter" policy, that determines the minimum time between two messages, a component can establish a temporal window to receive messages from other components.

2.2 Event-Based Control and Quality of Control

As control system complexity grows, timing requirement became difficult to be complied; therefore, the time-driven based control approach (TBC) is not the most efficient model. The Event Based Control (EBC) [9] complements the TBC model decreasing the messages needed to receive data and send control commands.

In the EBC model, messages between sensors, controllers and actuators, are only sent when an important condition is fulfilled. A wide range of conditions can generate events. The most common condition is related to error between the control action and the value obtained, and related to connection maintenance (keep-alive messages). EBC model requires a communications infrastructure based on events. From agent's point of view, DDS model provides a system based on events, implemented by means of Publish-Subscribe paradigm in the communications components.

In the same way that to evaluate the efficiency of communications uses QoS parameters; control must provide the corresponding parameters (QoC). It's considered a good control when the signal is sent to the actuator causes the signal measured by the sensor is identical to a reference signal, therefore there is no error between the measured signal and reference signal. Control error is used to modify the signal sent to actuator. The most commonly used QoC parameters are the value of the Integral Absolute value of Error (IAE) and the Integral of the Time and the Absolute value of Error (ITAE). Both parameters allow the system to know how evolve the error, and predict the new action. In the EBC model, the QoC should include the aspects related with the event management [10] which implies the existence of common parameters with the QoS such as throughput, delay and delay jitter.

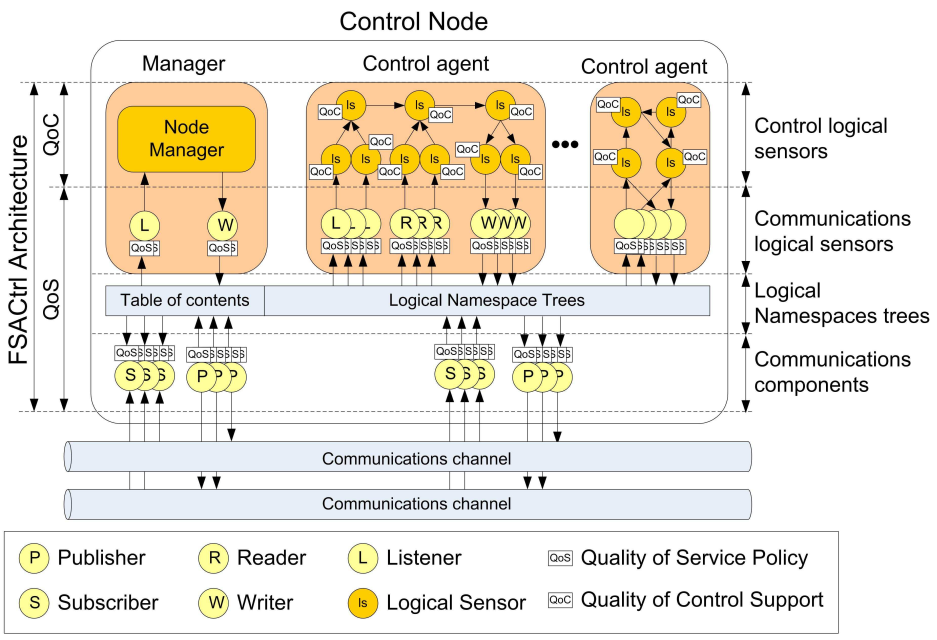

Fig. 1. This figure shows the overview of the architecture FSACtrl with the main communications and control components and connections between them.

3 FSACtrl Architecture

Figure 1 shows the main components of the architecture FSACtrl [11]. Each control node has a manager agent, the necessary control agents, the communications components to provide support at control agents, and a set of topics to connect the control agents with the communications components, Topics are organized in an ontology called Logical Namespace Tree.

Each control node contains a special ontology called Table of Contents (TOC). TOC describes the control node to the other control nodes of the system. The communications components are the components proposed by the DCPS model of the DDS standard. Publishers and Subscribers are common to all control agents, whereas Data Writers, Data Readers and Listeners are exclusive to each control agent.

Control algorithms are implemented by the components called Control Logical Sensors that provides the QoC parameters. The logical sensors are grouped hierarchically by means a component called Control Component. Thus the architecture provides the support to hierarchical control.

Manager agent provides support to all functions of the MAS: processes the request of the control agents, both from within and outside the control node, manage the ontology to connect successfully communications components and control components and mediates in the negotiation based on the QoS and QoC parameters.

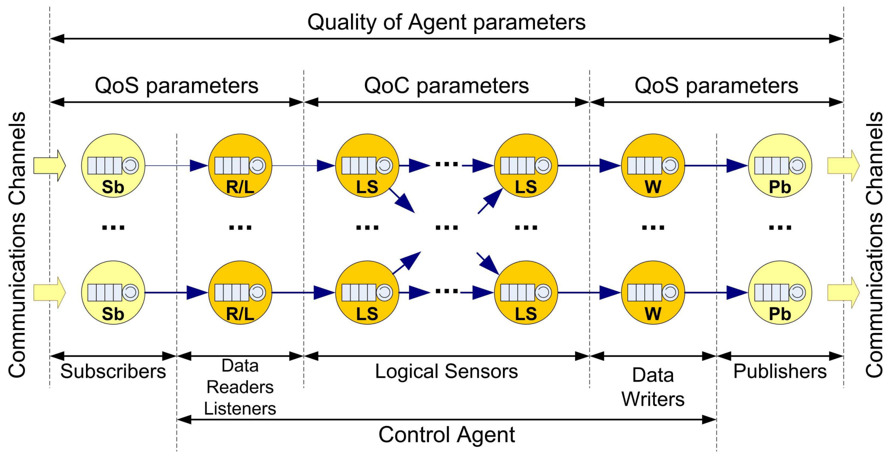

Fig. 2. QoS and QoC location within the FSACtrl components.

4. Joining the QoS and the QoC

4.1 QoS and QoC within the control nodes

The events of the system arrive to agent by means of the Subscribers. Since the message arrives to agent, it generates a chain of messages that follow a path between components. Finally, if is necessary, the agent sends the action by means of the Publishers (figure 2). The QoS parameters are obtained from the connections between the communications elements of the control node (Publishers and Subscribers) and the communications elements from the agent (Data Writers, Data Readers and Listeners). The QoC parameters are obtained from the control components (Logical Sensors) of the agent and its connections. To avoid a deadlock, cycles are not allowed between Logical Sensors.

All communications and control components of the FSACtrl architecture have a unique message queue to manage the incoming messages. Besides, all components have a unique control thread. The control thread contains the control algorithm in the case of Logical Sensor or the communications code in the case of communications components. Incoming and outgoing connections must work according with the QoS and QoC values agreed with the others components. Wearing a unique queue for every atomic component can make the component a bottleneck. To avoid it, the manager agent controls the QoS parameters and can duplicate some components and propose the agent movement to other control node with better conditions.

Since each component can provide parameters of both types (QoS and QoC) combining single parameters is possible to obtain general parameters to measure the QoS or the QoC about the Control Agent or Control Node. These parameters are known as Quality of Agent (QoA) parameters.

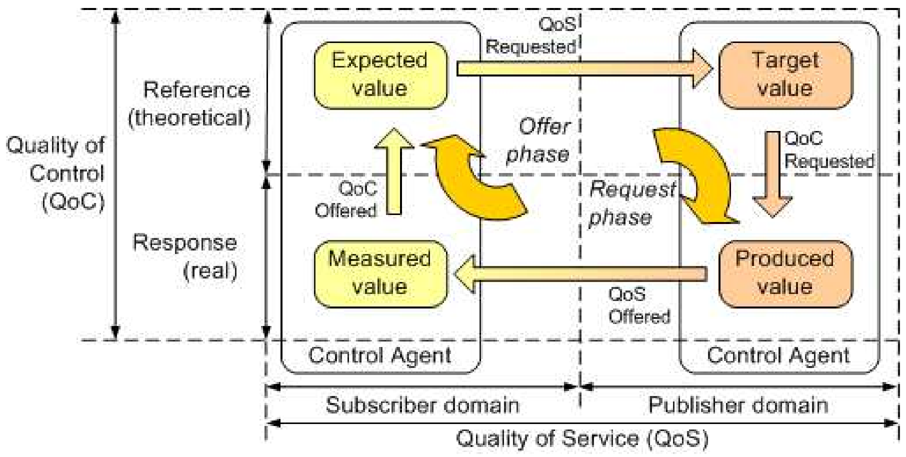

Fig. 3. The QoS and QoC cycle with the request phase and the offered phase.

4.2 The Quality Request-Offer Cycle

In a Multi-Agent System (MAS) when the communications between the agents is based on a middleware, every agent can works as a server, to offer their services, and as client, to use the services of other agents. Usually, in distributed systems when an agent needs an interchange of information with other agent, is necessary an initial phase to negotiate the values of QoS parameters. Additionally, in EBC systems when an agent needs to change the control action, is necessary a phase to negotiate the new values. Consequently, the values of QoC parameters need to be adjusted.

DDS offers a protocol for negotiating the QoS parameters by means the QoS policies. However, when the QoC is included in the negotiation process, it is necessary that the control agent can provide the appropriate protocol and parameters to measure the optimization level. In FSACtrl, the protocol is based in both types: QoS and QoC parameters.

Figure 3, shows the location of the parameters in the control loop. Based on the QoS cycle [12], FSACtrl architecture adds the QoC in the protocol. The QoS parameters are associated with the publish-subscriber communication and the QoC parameters characterize the relationship between the theoretical expectations and the real results of the control actions, both the client and the server side.

As seen in figure 3, the QoS and QoC have a request phase to communicate the value needed of the parameters and the next phase to communicate the values offered. Consequently the cycle consists in two cyclic phases: the QoS and QoC request, and the QoS and QoC offer. This cycle is known as “quality request-offer cycle”. Since a single component, for example a Logical Sensor, to an agent, the cycle runs as a service. The service allows the user to know the values of the formulas that estimates the relation between the QoS and the QoC. By correlating QoS parameters with QoC parameters, the agent is able to decide to prioritize the communications in front of the control or vice versa.

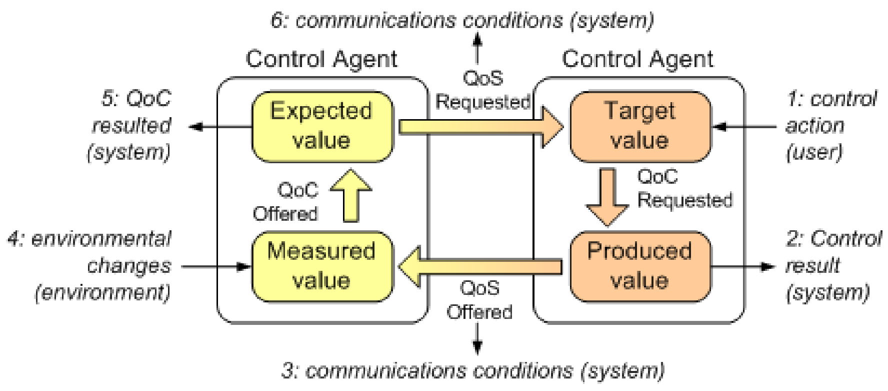

Fig. 4. Events that trigger the quality

There are two ways to start the cycle: by manager agent's initiative and by event. When the Manager Agent wants to monitor a specific situation, starts the service in the control agents.

Whenever an agent performs an operation over a specific Control Node (for example, move or clone), the Manager Agent verifies if there is any Control Node in the distributed system that provides the QoS and QoC requested. The quality request-offer cycle is used to determine which Control Nodes are suitable to locate an agent, in the case of insert and move operations.

Figure 4 shows the events that can trigger the quality request-offer cycle. There are two external events, one and four, and three internal events, the rest. The external triggering can be started by the user when changes the control action (Figure 4, case 1), or by the environmental (Figure 4, case 4), for example when a robot changes the path from the prefixed. The internal events can be located on the control action agent (Figure 4, case 2), on the agent controlled (Figure 4, case 5) and in the communications interchange (Figure 4, case 3 and 6). The cycle defines the actions to take in function of the event triggered. When an event occurs, the adjacent elements related with the event are altered. For example when the user changes the action control, the expected and the produced value changes, so that, the QoC requested and the QoS requested parameters are reviewed. The QoS and QoC changes due to the new control action can need other features of the same data, like an increase in the samples from a sensor, or new information from other control agents, for example new sensors samples.

5. Concluding remarks and future work

This article has shown the need to have QoS and QoC parameters in distributed intelligent control systems. A control paradigm based on events needs the synergy between communications and control. FSACtrl architecture allows QoS and QoC parameters, whatever type of components that implements the agent. Architecture is suitable to implement a distributed intelligent control system based on events.

Currently, the software developed based on FSACtrl architecture is not suitable to implement hard real-time systems because it works in TCP/IP based networks. However, it is very suitable for simulate the behaviour of specific agents topologies in the distributed system in order to optimize global system.

Some research lines related with the FSACtrl architecture are the study of the relations of the QoS and QoC parameters. Other field to work is the development the QoC policies to manage the QoC parameters, in the same way that DDS implements QoS policies to manage the performance of the communications.

Acknowledgements. The architecture described in this article is a part of the coordinated project SIDIRELI: Distributed Systems with Limited Resources. Control Kernel and Coordination. Education and Science Department, Spanish Government. CICYT: MICINN: DPI2008-06737-C02-01/02.

References

1. Lee E.A.: Cyber Physical Systems: Design Challenges. In: 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing. pp.363-369 (2008)
2. Siegel, J., "CORBA 3: Fundamentals and Programming," OMG (2000)
3. FIPA. FIPA-QoS web site, http://www.fipa.org/specs/fipa00094 (2002)
4. Object Management Group (OMG): Data Distribution Service for Real-Time Systems, v1.1. Document formal / 2005-12-04. (2005)
5. Posadas, J.L., Poza J.L., Simó J.E., Benet G., Blanes, F.: Agent Based Distributed Architecture for Mobile Robot Control. Engineering Applications of Artificial Intelligence. Pergamon Press Ltd. Vol.: 21 N. 6. p.p. 805-823 (2008)
6. Aurrecoechea, C., Campbell, A.T. and L. Hauw.: A Survey of QoS Architectures. ACM/Springer Verlag Multimedia Systems Journal, Special Issue on QoS Architecture, Vol. 6 No. 3, pg. 138-151 (1998)
7. Pardo-Castellote, G. OMG Data-Distribution Service: architectural overview. Proceedings of 23rd International Conference on Distributed Computing Systems Workshops. Providence, USA. Vol. 19-22, pp. 200-206 (2003)
8. International Telecommunication Union (ITU). Terms and Definitions Related to Quality of Service and Network Performance Including Dependability. ITU-T Recommendation E.800 (0894) (1994)
9. Sánchez, J., Guarnes, M.Á., Dormido, S.: On the Application of Different Event-Based Sampling Strategies to the Control of a Simple Industrial Process. Sensors. 9, 6795-6818. (2009)
10. R.C. Dorf and R.H. Bishop, Modern Control Systems, 11th Edition, Prentice Hall (2008)
11. Poza, J.L., Posadas, J.L., Simó, J.E.: Middleware with QoS Support to Control Intelligent Systems. 2th International Conference on Advanced Engineering Computing and Applications in Sciences, ADVCOMP, pp.211-216 (2008)
12. Poza, J.L., Posadas, J.L., Simó, J.E.: From the Queue to the Quality of Service Policy: A Middleware Implementation. In: S. Omatu et al. (Eds.): IWANN 2009, Part II, LNCS 5518, pp. 432–437 (2009)