Quality of Control and Quality of Service in Mobile Robot Navigation

José-Luis Poza-Luján, Juan-Luis Posadas-Yagüe and **José-Enrique Simó-Ten**[1]

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

ABSTRACT

This article presents the experimental work developed to test the viability and to measure the efficiency of the intelligent control distributed architectures. To do this, a simulated navigation scenario of Braitenberg vehicles has been developed. To control the vehicles, system uses a distributed control architecture that provides support to QoS and QoC parameters to optimize de system. The architecture uses a Publish-Subscribe model, based on Data Distribution Service to send the control messages. Due to the nature of the Publish-Subscribe model, the architecture is suitable to implement event-based control systems. The architecture has been called FSACtrl. To test the efficiency, the architecture provides a set of Quality of Service parameters. In the experiment described in this paper, the performance is used as a reference. The measuring of the quality of the navigation is done through the Integrated Time Average Error as Quality of Control parameter. Tested scenarios are: an environment without quality parameter managing, an environment with a relevant message filtering and an environment with a predictive filtering determined by the type of scenario used. Results obtained show that some of the processing performed in the control nodes can be moved to the middleware to optimize the robot navigation.

Keywords: Distributed Systems, Control Architectures, Quality of Service, Quality of Control.

Mathematics Subject Classification: 68M14, 68M20, 68T42, 90B22

Computing Classification System: I.2.8, I.2.9, I.2.11

1. INTRODUCTION

In mobile robot navigation architectures, different components work at different control nodes that are connected through the communications channels.

To measure the efficiency of the communications, and the quality of component's services offered, system uses the concept of Quality of Service (QoS) (Vogel et al., 1995), that measures the degree of services compliance, through the QoS parameters (Crawley et al., 1998). The communications management functions that are oriented to optimize the QoS parameters are known as QoS policies (Bradner, 1996). Among standards to manage distributed communications systems, the Data Distribution Service for Real-Time Systems (DDS) standard, proposed by the Object Management Group (OMG) (OMG, 2005), implements a large type of QoS policies. DDS is based on publish-subscribe paradigm, extended with some elements that connect the application synchronously (readers and writers) and asynchronous (listeners). Therefore, DDS is well suited for implementing distributed intelligent control architectures (Poza et al., 2011).

To measure the control efficiency, currently is used the concept of Quality of Control (QoC) (Dorf and Bishop, 2008). The QoC measures the quality of the control action through equations, generally using the difference between the input signal and the reference signal. Sometimes the QoC parameters are used as feedback of control action; thus, the QoC measures the control efficiency and it makes easier the control processing. To measure the QoC, the system must provide separated control nodes that process functionally independent control loops. Among the diverse proposals, the model Sensor Web Enablement (SWE), proposed by the Open Geospatial Consortium (OGC), monitors and processes sensor data from multiple nodes distributed in space. Besides, SWE includes the organization of the processes.

The control efficiency does not depend exclusively on the algorithms used; the communications efficiency also affects the control action (Soucek and Sauter, 2004). To prove the relationship between QoS and QoC, an architecture called Frame Sensor Adapter to Control (FSACtrl) has been developed (Poza et al., 2010). FSACtrl allows measuring QoC and QoS parameters in control nodes. The architecture is based on DDS standard, and it uses the DDS QoS policies to manage the communications. In addition, FSACtrl offers in its control components, the IAE and ITAE parameters.

This paper describes the architecture, parameters and tests performed in a simulated mobile robot environment. It shows results obtained by using QoS and QoC to measure the efficiency of control node depending on the communications configuration.

The paper is organized as follows: the following section review the related work about the subject of the article. Next, the third section presents the theoretical concepts used to design the FSACtrl architecture, and a description of the components of this architecture. The fourth section describes the environment used to perform tests of the architecture: simulation environment and simulated robots and describes the QoC and QoS parameters that have been considered in the described environment. The fifth section describes the tests performed and results. Finally, the paper ends with conclusions of experiments done and the future work to be developed.

2. RELATED WORK

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 (Lee, 2008).

Systems must be optimum to carry out of the objectives fixed. In distributed control systems, it is necessary to optimize the performance, for example the energy consumed (Gusrialdi, 2013). To optimize a system it 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 components 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 moving or cloning a component between two control nodes.

There are a lot of network 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 (Siegel, J., 2000). The Foundation for Intelligent Physical Agents (FIPA) defines 14 QoS policies mainly in terms of speed and reliability (FIPA, 2002). The Data Distribution Service model DDS 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.

As control system complexity grows, timing requirement becomes difficult to be complied; therefore, the efficiency of the time-driven based control approach (TBC) depends on the context. The Event Based Control (EBC) (Sánchez et al., 2009) 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). When the EBC model is applied in a Networked Control System (NCS), the core of the system is a communications infrastructure based on events with support to distribute efficiently the system information.

There are a great amount of NCS based on events. (Yan et al., 2011) highlights the importance of the network architecture and the protocol used, in addition, (Tang and Yu, 2007) emphasizes the importance of measuring the QoC in the NCS. Systems that work in NCS environments must cover these features.

3. FSACTRL ARCHITECTURE: CONCEPTS AND DESCRIPTION

3.1. DDS and Quality of Service

There are different paradigms of communication with support to quality of service, among them publish-subscribe model is one of the most suitable (Aurrecoechea et al., 1998) due that isolates publishers of subscribers, enabling a QoS negotiation based on the information topics. The components only need to know the topics to send or receive the information, without knowing the current location of the other components.

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.

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. Figure 1 shows the main components of the DCPS layer from the DDS model.

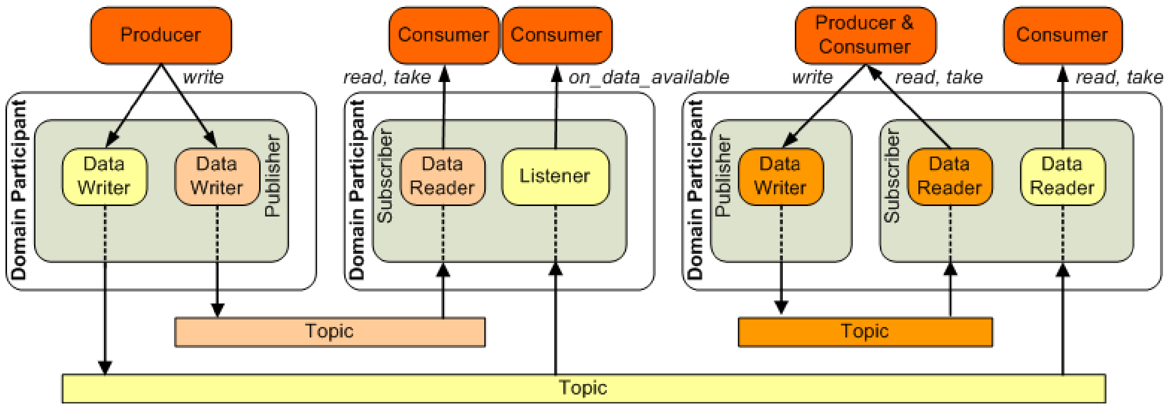

Figure 1. Overview DCPS components from DDS model.

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 and a Listener 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 (ITU, 1994). 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.

3.2. SWE and Quality of Control

The main objective of Sensor Web Enablement (SWE) is providing a unique and revolutionary framework of open standards for exploiting Web-connected sensors and sensor systems of all types (Botts et al., 2006). SWE was developed in 2004 as part of an initiative by the OpenGIS Consortium

(OGC). At present SWE is used especially for monitoring and management of sensor networks. The proposed model is currently used by many organizations, like NASA and computer weather systems. SWE assign control functions to several interconnected elements.

The components of SWE are divided in two groups: information models and services. Information models are standard specifications in XML, processes interchanges messages with these specifications. Services are control components that process the information models. Control processes are based on components interconnected, those receive information models from other components, and send the results to connected components.

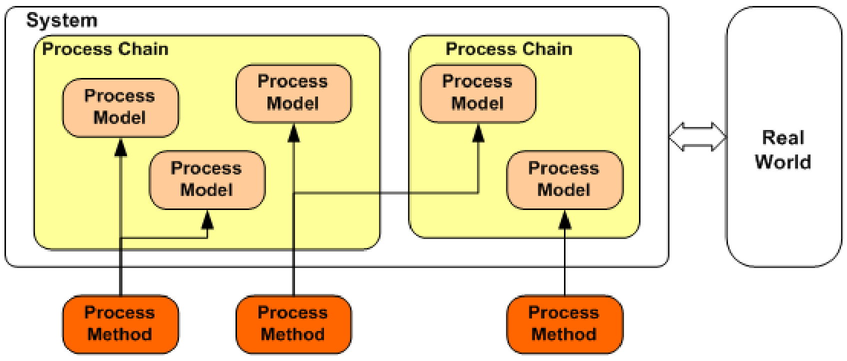

Figure 2. SWE control architecture components overview.

From SWE viewpoint, a component is a particular physical process that transforms information. Simple examples of SWE components are sensors, effectors or physical process filters. Complex examples of SWE components are control kernels or sensor data fusion algorithms.

As shown in the Figure 2, a “Process Model” is a single component, used into a more complex structure, called “Process Chain”. Moreover, a “Process Model” is based on a “Process Method” which acts as a “Process Model” template. A “Process Method” specifies the interface and how to implement the “Process Model”, also define inputs, outputs and the operating parameters. The model proposed by SWE is very interesting because allows to specify reusable process patterns. This scheme provides a highly scalable control system based on singles control kernels.

Anyway, it should take some precautions when using this scheme. The highly interconnected model increases redundant information because the model hides the data sources. Also, repetition in control patterns can lead to control actions repeated. Finally, the interconnection of control models can generate undesirable control cycles. Any SWE based architecture must prevent these aspects.

In the same way that QoS parameters are used to evaluate the efficiency of communications; control must provide the corresponding parameters (QoC). It’s considered a good control when the signal sent to the actuator causes that 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 to evolve the error, and predict the new action.

The SWE model is very suitable to provide the QoC parameters. This is because a general control action corresponds to a SWE Control Method component, and the error of the control action is obtained directly from the Control Method.

3.3. FSACtrl Architecture

In the EBC model, the QoC should include the aspects related with the event management which implies the existence of common parameters with the QoS such as throughput, delay and delay jitter. To provide QoS and QoC support, the FSACtrl architecture is inspired by DDS and SWE models. FSACtrl components arise from the viewpoint of agents. This is because the components, besides offering services, make their own decisions based on the QoS and QoC parameters

FSACtrl is an evolution of an architecture called Frame Sensor Adapter (FSA) (Posadas et al., 2008) developed by the authors research group. The architecture has two distinct areas: communication and control. QoS Policies connects both areas. Figure 3 shows the details of the architecture.

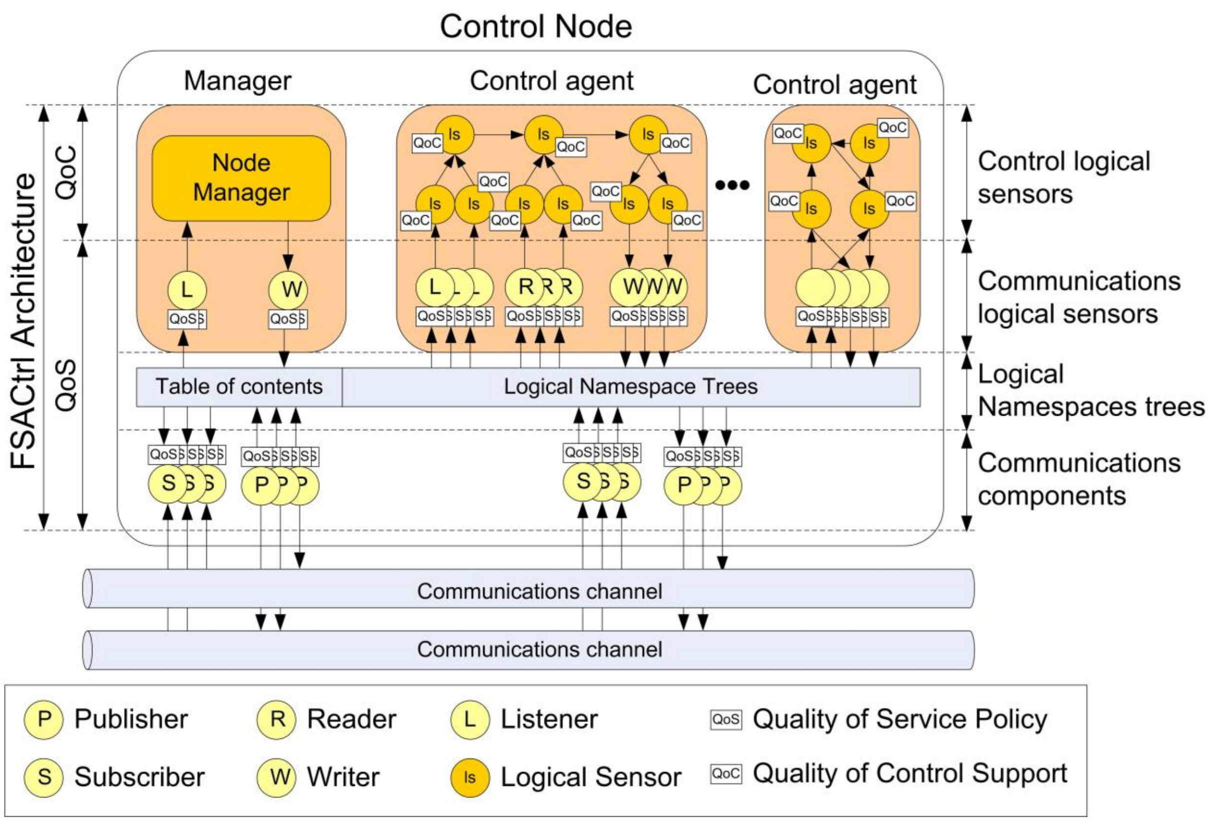

Figure 3. Overview of the architecture FSACtrl with the main communications and control components and connections between them.

The “Frame” component of the FSA architecture takes the same role of the “DomainPaticipant” component of the DDS architecture. The “Adapter” component takes the role of both DDS components, “Publisher” and “Subscriber”. A specialization of “Sensor” component takes the role of the “DataWriter”, “DataReader” and “Listener” DDS components. The function of the “Topic” DDS component is performed by the “LogicalData” component of the FSACtrl architecture. The communication layer organizes the “LogicalData” in a hierarchical structure to hide any type of communication channel like the TCP/IP protocol, EIB or CAN bus. The structure is a symbolic tree called Logical Namespace Tree (LNT).

Control layer organizes the “Sensors” on a graph, called Logical Sensor Graph (LSG). This model is based on SWE “Process Chain". The process units are known as “Logical Sensors”, and some of this “Logical Sensors” takes the role of some communication components. A “Logical Sensor” can receive, or send, messages from, or to, another “Logical Sensors”.

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.

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.

The logical sensors are grouped hierarchically by means a Control Agent that implements the control algorithm and provides the QoC parameters. Thus the architecture provides the support to hierarchical control.

The Manager Agent processes requests of control agents. These requests can come from within and outside the control node. Besides, the Manager Agent manages the ontology to connect successfully communications components and control components and mediates in the negotiation based on the QoS and QoC parameters.

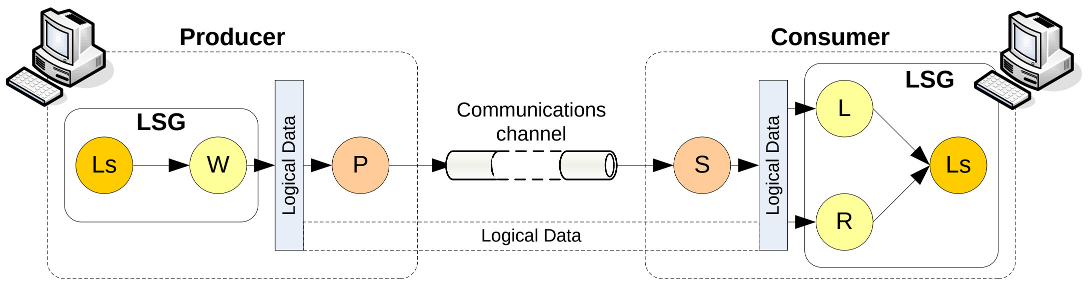

Figure 4. The FSACtrl architecture components and connections involved in the information and control distribution.

When a control service, provided by a producer, has to send information to consumers, it generates a message and sends it to a DataWriter. The DataWriter uses the Logical Data to send the message to selected consumers. The Logical Data connects DataWriters with corresponding communications channels through Publishers. Publishers send messages to the Subscribers according to the characteristics of the communications channel. Messages arrive at consumers by means of the Subscribers (figure 4).

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 (Poza et al., 2008). 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.

As 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

4. EXPERIMENTAL ENVIRONMENT

To facilitate the design and implementation of control algorithms based on FSACtrl architecture, it has been developed an editor that provides a developer graphical environment with predefined agent or component schemas.

To test the architecture, the control of first five Braitenberg vehicles (Braitenberg, 1984) has been implemented. The first three vehicles are characterized by the lack of advanced control functions; so that, these vehicles are suitable for evaluating the performance of the communications because messages are processed principally in the middleware. The interest of Braitenberg vehicles is in the simplicity of control, based on the simple functions that connect sensors and actuators. In addition, the possibility to have different types of sensors that react to different sources provides a lot of messages that are used to test the effect of communications configuration in the control efficiency.

4.1. Simulation infrastructure

A simulator of mobile robots has been implemented to test proposal control algorithms. Figure 5 shows the topology of the distributed system used to test the architecture.

The simulator allows user to create a 2D environment and insert any number of robots. For each robot, the simulator has twelve different types of sensors. All robots are circular and have two motors. This configuration allows robots to move in any direction in the simulated environment.

The robot simulation environment is composed of a space with different signal sources and rectangular and circular obstacles. The simulator sends via TCP clients the data from the sensors of each robot, and it receives, via a TCP server, speeds assigned to each robot motor.

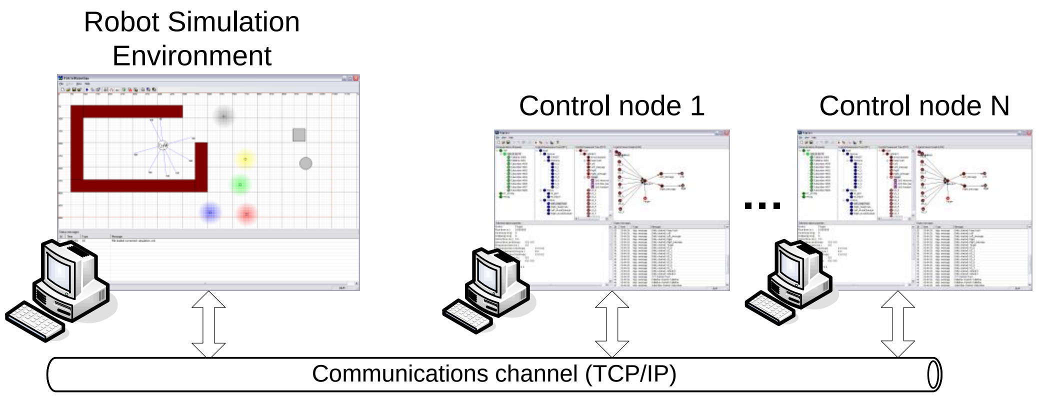

Figure 5. Experimental environment used to test the FSACtrl architecture.

4.2. Architecture implementation

Control nodes are composed of an FSACtrl elements editor that launches the control processes. The editor allows insert, modify and configure QoS policies and QoS parameters to each FSACtrl architecture element. The system implements the control node over personal computers on a TCP/IP based network. The accuracy of the measurements in the control nodes is nanoseconds; the computation time of the control nodes has been simulated in order to obtain comparable results

First three Braitenberg vehicles are based on connections between sensors and actuators without the intermediation of control processes. The Braitenberg vehicle 3.c (figure 6) combines the features of previous vehicles, increasing the number of type sensors to four: light, temperature, oxygen and organic.

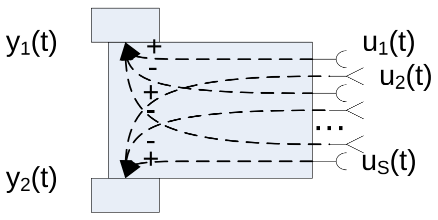

Figure 6. Braitenberg vehicle 3.c with the different input: sensor sources $U_x(t)$ and the two outputs: motors $Y_1(t)$ and $Y_2(t)$.

The Braitenberg vehicle 3.c is used to test the FSACtrl architecture as a middleware and how the middleware can be used to optimize the system. FSACtrl elements used to implement the Braitenberg vehicle 3.c are shown in figure 7.

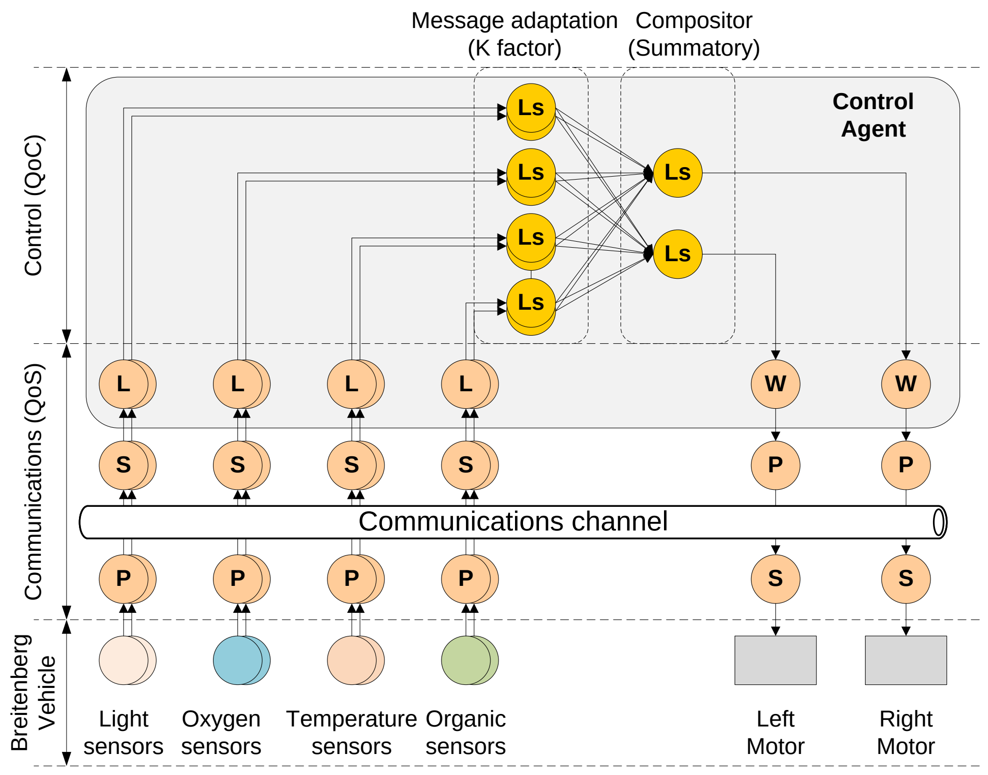

Figure 7. FSACtrl architecture implementation of the Braitenberg 3.c vehicle used to make experiments.

Braitenberg vehicle sends sensors values to Subscribers through a specific Publisher for each sensor. Every sensor needs its own QoS parameters (i.e. .frequency sampling) and the value of these parameters can change throughout the navigation time. Subscribers send sensor values to the Agent Control Listeners.

The output of each of the composers is calculated from the contribution of each input of the N sensors of the vehicle, weighted by a specific K factor for each sensor. The vehicle calculates the direction that should be taken based on information obtained from the four types of sensors available, using the next equation.

$$Output_{compositor} = \sum_{i=1}^{N} K_i \cdot Input_i$$

The implementation of the behavior described on the equation above with an FSACtrl Agent Control needs two steps. The first step performed by the control agent Logical Sensors is the message adaptation by means a weighting factor K. As a result of this step, messages of different sensors have a specific weight to the control action. The second step generates a single control signal to every motor from each sensor input.

4.3. Quality of Control in robot navigation

To obtain the QoC parameters it is necessary to define the control error. In the 3.c Braitenberg vehicle the control error is measured by means of the angle that the vehicle deviates from the planned angle in the theoretical analysis of the vehicle mission (figure 8).

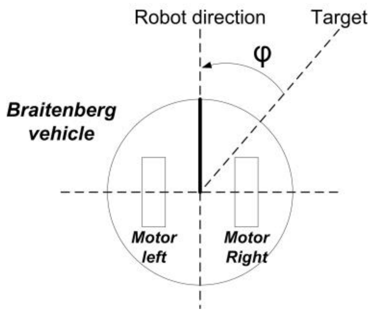

Figure 1. Path error in the Braitenberg 3.c vehicle used to calculate the ITAE parameter.

The equations to obtain the quality of control parameter can be very different, because of the quality parameter is directly associated to the characteristics of the robot on which it is applied (Gabel and Litz, 2004). In the case of vehicle 3.c the quality of control is directly calculated with the parameter ITAE shown below

$$ITAE = \int_{t_0}^{t_{END}} t \cdot \left| \varphi_y(t) - \varphi_r(t) \right| dt$$

In the previous equation, φy(t) is the value of the desired angle for a time t, while φr(t) is the real angle obtained in the same instant of time. ITAE parameter considers the navigation error with the same weight during all the navigation time, so that it is very suitable to make global comparisons. The smaller ITAE value, the better quality of navigation of the vehicle is. This is because of the angle obtained from the course is closer than expected angle

4.4. Quality of Service in robot navigation

Middleware manages the QoS. In the case of FSACtrl architecture, QoS is managed by the QoS policies of the DDS standard. In the tests performed the QoS parameters that have been measured are the control component load and the rate of useful messages. The control component load (ρ) is calculated as the rate between the service demand and the service rate of the component. Due to the architecture elements are made by messages queues, global load is obtained through the pondered rate of each element load (next equation). The K factor is used to balance the most important control components.

$$\rho_{global} = \sum_{i=1}^{N} \left(K_i \cdot \rho_i \right) \Big/ N$$

To calculate the load ρ of each component is used the next equation, where λ is the demand for the services requested from the vehicle control and μ is the rate of service provided by the control component. Both of these parameters are expressed in messages per second so that the load is a dimensionless parameter. Closer load to zero better is the control component load.

$$\rho = \lambda / \mu$$

The useful messages rate (UM) is obtained using the following equation. The concept of utility of a message can be quite large. In the experimental environment a useful message is considered when the message produces a change in vehicle navigation. The variation of navigation is produced when the control action calculated for a measurement is different from the control action calculated for the previous measurement. Closer to one are, better the parameter is. The control action in Braitenberg vehicles is performed on the speed of the motors.

$$UM = N_{output\ (i) \neq output\ (i-1)} / N_{total}$$

From the two previous equations, the performance (η) of the control can be obtained with the equation shown below. Performance is defined as the satisfactory results obtained in relation to the cost in resources used. The control performance is obtained through the parameters from the equations 4 and 5.

$$\eta = UM \cdot \left(1 - \rho_{global}\right)$$

Through the performance equation, can be verified the effectiveness of the control messages related to the resources consumed from the control service. If the performance value is close to 1, the control action, viewed as a service, is optimized. So that, the value of ITAE indicates how to the service improvement affects the vehicle navigation. If the performance value increases, and the ITAE value remains in the same ranges for all cases, the system is optimized without affecting the vehicle navigation.

5. EXPERIMENTAL TESTS AND RESULTS

Three scenarios on the architecture with the Braitenberg vehicle 3.c have been tested. In the first scenario (scenario one) the control action is obtained without filtered messages optimization and without messages selection optimization: not QoS and not QoC management. The scenario two the control action is obtained with filtered messages optimization and without messages selection optimization: QoS managed but not QoC management. Finally, the scenario three obtains the control action with filtered messages optimization and with messages selection optimization: QoS and QoC managed.

Message filtering consists of transmit through the middleware only those messages whose content is different, compared with the preceding message. The message filtering is one of the characteristics specified in DDS standard recommendations for a middleware. Publishers are the components responsible for this optimization.

Messages selection produces the improvement of control optimization. This selection is performed by inserting control components that predict changes in the control action. The prediction is made by comparison between messages from different sensors involved in the calculation of control action.

The environment is a system without obstacle with the four types of sources associated with the four types of sensors of the vehicle 3.c of Braitenberg (figure 6). The vehicle is configured to be attracted by light and organic matter sources, and to be rejected by heat and oxygen sources. The vehicle follows a path that depends on the location of the sources in the environment (figure 9).

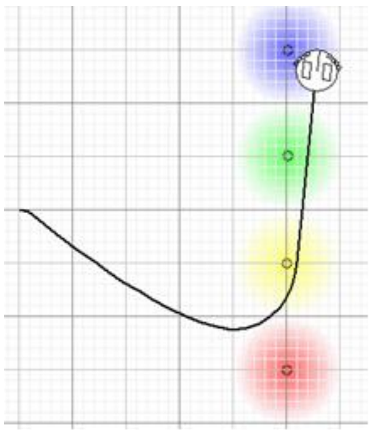

Figure 9. Example of 3.c Braitenberg vehicle navigation in a multi-source environment with the composition of different types of sensors sources.

Tests have been performed starting the vehicle in the same position and the sources placed in the same location and changing the middleware according to each scenario described. Table 1 shows experimental values for each of the scenarios described at the beginning of the paragraph. Columns show the average values of the control load, the usefulness of messages rate, the performance of the control element and the value of ITAE. Each row contains the data for each of the scenarios described above.

Table 1: Experimental results based on different scenarios (average values)

Scenarios	*ρ*	*UM*	*η*	*ITAE*
One: Without QoS and QoC management	0.184	0.212	0.173	0.252
Two: With QoS management and without QoC management	0.121	0.323	0.284	0.261
Three: With QoS and QoC management	0.119	0.683	0.602	0.284

Due to the response time of control service is the same in all scenarios tested; the variation of the control load depends on the message arrival frequency. Because of the scenarios two and three include a message filtering phase the control load decreases significantly respect the scenario one.

UM rate changes progressively among the three different scenarios. In the scenario two, UM value rises respect the scenario one because the middleware has filtered some messages that do not generate a control action. However, the most significant improvement of useful message index is produced in the scenario three. In the scenario three, the control receives only messages that haven't been filtered in the middleware and in the control prediction. For this reason the message utility rate increases considerably compared with the previous two scenarios. Figure 10 shows the comparison between the service performance index (η) and the control index (ITAE).

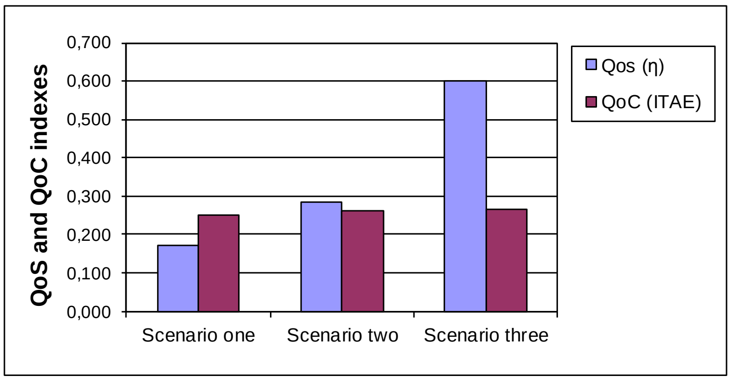

Figure 10. Comparison chart between the η values (QoS) and the ITAE values (QoC).

The service performance describes the common contributions of the two parameters analyzed and it is a good measure of the quality of service that the control component provides. The figure shows how performance is directly related to the optimizations used in each scenario. ITAE parameter is used to check the efficiency of the control service optimizations of the vehicle navigation. In this case, ITAE parameter increases very slightly in relation with the optimized scenario, so that improvements implemented on every scenario scarily affects the quality of the robot navigation.

6. DISCUSSION AND CONCLUSION

Currently, in the field of the NCS, the increasing requirements of intelligent distributed control, increases the communications requirements. A distributed control paradigm needs the synergy between communications and control. To measure the impact of the relation into communications and control, the system should provide the tools to measure the performance.

FSACtrl architecture allows QoS and QoC parameters in all components that implement every agent. Architecture is suitable to implement a distributed intelligent control system based on events. It is based in two standards architectures, DDS and SWE, and takes the benefits of a QoS-based communication. DDS, based on publish-subscribe paradigm, is a standard supported by OMG. SWE standard is endorsed by OGC. The FSA-Ctrl architecture focuses especially on the use of QoS policies and QoC parameters.

FSACtrl architecture has been tested in a simulated mobile robot environment by using the first three Braitenberg vehicles in three scenarios with different optimizations that are measured by means of QoS and QoC parameters.

Results of the experimental work carried out are satisfactory. Results show that the FSACtrl architecture is viable as a middleware with support to simple control actions. It is also proves as manager of the communications layer allows to optimize the control layer that affects overall system optimization.

In the specific field of robot navigation, the experiment demonstrated that using a middleware to preprocess sensor messages increases the system performance. However, the quality of the control action is affected, but in lesser proportion. Determine when it is necessary to apply Communications or Control optimizations is a potentially line of research.

As future work, several studies related with the relation between QoS and QoC can be performed. One of the most interesting questions, to develop, is the dynamic adjustment, through QoS policies, of the robot navigation. The concept of the dynamic variation can be extended to the QoC with the QoC policies. The objective is determine the convenience to adjust the communications and control characteristics, as the sampling frequency, according to certain environmental and design constraints such as energy consumption or the time to complete the mission of the vehicle.

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

7. REFERENCES

Aurrecoechea, C., Campbell, A.T. and L. Hauw., 1998. A Survey of QoS Architectures. ACM/Springer Verlag Multimedia Systems Journal, Special Issue on QoS Architecture, Vol. **6** No. 3, pg. 138-151

Botts M., Percivall G., Reed C. and Davidson J, 2006, OGC. *Sensor Web Enablement: Overview and High Level Architecture*, OpenGIS Consortium Inc.

Bradner, S., 1996, RFC 2026: *The Internet Standards Process*. IETF Internet Draft. section. **10**.

Braitenberg, V., 1984. *Vehicles: Experiments on Synthetic Psychology*. MIT Press, Cambridge, Massachusetts.

Crawley, E., Nair, R., Rajagopalan, B., 1998, RFC 2386: *A Framework for QoS-based Routing in the Internet*. IETF Internet Draft. pp. 1-37.

Dorf, R.C. and Bishop, R.H., 2008, *Modern Control Systems*, 11th Edition, Prentice Hall.

FIPA. 2002, FIPA-QoS web site, http://www.fipa.org/specs/fipa00094

Gabel O. and Litz L., 2004, QoS-adaptive Control in NCS with Variable Delays and Packet Losses – A Heuristic Approach, 43rd IEEE Conference on Decision and Control, Atlantis, Paradise Island, Bahamas. Vol. **1**, pp. 1586-1591.

Gusrialdi, A. R. Dirza, T. Hatanaka and M. Fujita, 2013, Improved Distributed Coverage Control for Robotic Visual Sensor Network under Limited Energy Storage International Journal of Imaging and Robotics, Vol. 10, No. 2, pp. 58–74.

ITU (International Telecommunication Union), 1994, *Terms and Definitions Related to Quality of Service and Network Performance Including Dependability*. ITU-T Recommendation E.800 (0894)

Lee E.A., 2008, Cyber Physical Systems: Design Challenges. In: 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing. Orlando, Florida, USA. Vol. **1**, pp.363-369

Object Management Group (OMG), 2005 , Data Distribution Service for Real-Time Systems, v**1.1**. Document formal / 2005-12-04

Pardo-Castellote, G., 2003, OMG Data-Distribution Service: architectural overview. Proceedings of 23rd International Conference on Distributed Computing Systems Workshops. Providence, USA. Vol. **19-22**, pp. 200-206

Posadas, J.L., Poza J.L., Simó J.E., Benet G., Blanes, F., 2008, Agent Based Distributed Architecture for Mobile Robot Control. Engineering Applications of Artificial Intelligence. Pergamon Press Ltd. Vol.: **21** N. 6. p.p. 805-823

Poza, J.L, Posadas, J.L. Simó, J.E., 2010, Multi-Agent Architecture with Support to Quality of Service and Quality of Control. 11 th International Conference on Intelligent Data Engineering and Automated Learning. Paisley, UK. Vol. **6283**, pp.137-144

Poza, J.L, Posadas, J.L. Simó, J.E., 2011, A Survey on Quality of Service Support on Middleware-Based Distributed Messaging Systems Used in Multi Agent Systems. In proceeding of: International Symposium on Distributed Computing and Artificial Intelligence, DCAI 2011, Salamanca, Spain. AICS Vol. **91**, pp. 77-84.

Poza, J.L., Posadas, J.L., Simó, J.E., 2009, From the Queue to the Quality of Service Policy: A Middleware Implementation. In proceeding of: Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living, 10th International Work-Conference on Artificial Neural Networks, IWANN 2009 Workshops, Salamanca, Spain, Part II, LNCS **5518**, pp. 432–437

Sánchez, J., Guarnes, M.Á., Dormido, S., 2009, On the Application of Different Event-Based Sampling Strategies to the Control of a Simple Industrial Process. Sensors. Vol.**9**, pp. 6795-6818.

Siegel, J., 2000, *CORBA 3: Fundamentals and Programming*. OMG.

Soucek S. and Sauter T., 2004, Quality of Service Concerns in IPBased Control Systems, IEEE Transactions on Industrial Electronics, Vol.: 51, Issue: 6, pp. 1249-1258.

Tang, X. and Yu, J., 2007, Networked control system: survey and directions. Lecture Notes in Computer. Science, Vol.: **4688**, pp. 473–481.

Vogel, A., Kerherve, B., von Bochmann, G. and Gecsei, J., 1995, Distributed Multimedia and QoS: A Survey. IEEE Multimedia.Vol.**2**, No. **2**, pp. 10-19.

Yan, H., Wan, J., Li, D., Yuqing, T., Zhang, P., 2011, Codesign of Networked Control Systems: A Review from Different Perspectives. In Proceedings of IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems. Kunming, China. Vo.**1**, pp. 84-90