Scheduling Algorithms to support QoS and Service Integration in Sensor and Actuator Networks

J.M.Cabral[1], J.G.Rocha[1], J.E.Neves[1] and J.Ruela[2]
[1] Industrial Electronics Department, University of Minho, Campus de Azurém, 4800-058 Guimarães, Portugal
[2] School of Engineering, University of Porto (FEUP-DEEC) / INESC Porto, Rua Dr. Roberto Frias, 4200-465 Porto, Portugal
jose.cabral@dei.uminho.pt

***Abstract* - In this paper we analyse and evaluate several Scheduling Algorithms that are candidates to support Quality of Service and Service Integration in Sensor and Actuator Networks. They should satisfy two main goals: to guarantee committed delays for time sensitive services, and to improve the network transmission efficiency. The algorithms are described and some results, obtained by simulation, are presented. The proposed *Traffic Class Oriented Algorithm* proved to be a good solution to meet the proposed objectives as well as to integrate traffic generated by Fieldbus devices and control applications in real communication networks.**

I. INTRODUCTION

Usually, a sensor network is composed of a large number of small devices, whose main objective is to detect and transmit some physical characteristic of the environment [1]. These components or nodes can be used in an efficient way to fulfil a single common objective, even when their number is in the order of thousands. A control system may integrate a large number of sensors, actuators and respective control entities. Therefore, even when the communication between each pair of devices is characterised by a low bit rate and requires moderate or small transmission delays, the aggregate bit rate to be supported by the network can reach very high values and time delays may become unacceptable, if not properly controlled.

This kind of traffic is usually supported by specific networks usually called Fieldbuses; CAN and Profibus are examples of Fieldbus technologies. They have many limitations, mainly in aspects related with the integration of services and systems, bandwidth and coverage area.

On the other hand, until recently, communications networks have been optimized to support specific services (e.g. transmission of voice, video or data files), thus requiring some form of adaptation to support other types of services. The current trend towards integration of services in the same network is usually associated with the need to support differentiated Quality of Service (QoS). Moreover, adapting low bit rate services in such networks is concomitant with the control of time delay in assembling and scheduling packets; these aspects have a significant impact on QoS.

To solve these problems a modular system architecture was studied and specified [2]. Its main component is a Terminal Adapter that allows multiplexing of individual flows generated by low bit rate services into a single aggregate flow.

The specification of a scheduling algorithm is essential to perform statistical multiplexing of different information flows, while taking into account delay requirements as well as other relevant QoS parameters.

The paper is organized as follows. Section II analyses the problem of integrating low bit rate traffic in communication networks. Section III presents some fundamentals of traffic scheduling, while Section IV discusses several algorithms used to perform traffic aggregation. Section V describes simulation results and in Section VI some conclusions are presented.

II. INTEGRATION OF LOW BIT RATE TRAFFIC IN COMMUNICATION NETWORKS

The main objective of this study is to propose and evaluate a solution that aggregates low bit rate traffic, usually associated to Fieldbuses, for transmission over an integrated services communication network. ATM (*Asynchronous Transfer Mode*) technology was selected due to its capability of multiplexing in an efficient way a large number of data flows, while supporting different delay requirements [3].

In order to specify the architecture of the Terminal Adapter, which supports the interconnections of Fieldbus devices into the communication network, it is first necessary to characterize the way different flows, generated by different devices, will be processed by a traffic aggregation system, as well as the time required for data transmission along a network.

A. Traffic Classes

Each traffic class, which aggregates flows with similar QoS requirements, will be treated by the scheduling algorithm such that different target performance levels of the control system are achieved. Three traffic classes were proposed:

• *Maximum Delay (MD)*

The flows associated with this class need a maximum and well-defined time delay guarantee between the sensing device and the control application, and between the control application and the acting device.

• *Data (D)*

In this class, delivery of information has not critical delay requirements. The only requirement is that all data must be delivered without losses, which may require a reliable end-to-end transport protocol to recover from network losses.

• *Minimum Effort (ME)*

This class can be used when occasional loss or high delay in information delivery does not affect the control process.

B. *Transmission delays*

The time required for data transmission along a network includes two components: the delay in processing data packets in terminal and network devices (e.g. packetization and queuing delays) and the propagation delay [4].

The value of 400ms was considered an acceptable limit for network planning purposes, where speech transmission performance was the focus [4].

C. *Terminal Adapter*

The Terminal Adapter is as a set of sending and receiving state machines that work in an independent way. The sending side must multiplex traffic flows from various sources (e.g. sensors) with different delay requirements, into a single flow for transmission over the network, while guaranteeing the specified QoS and maximizing transmission efficiency. The receiving side has to perform channel demultiplexing, in order to deliver individual flows to the actuators of the control system. Fig. 1 shows the Terminal Adapter architecture; it is based on Type 2 ATM Adaptation Layer (AAL-2), which is described in ITU-T Recommendation I.363.2 [5].

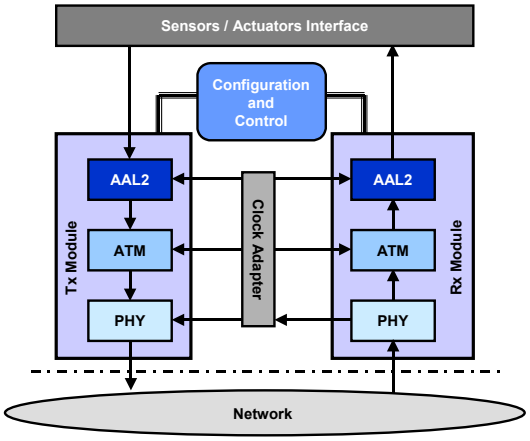

Fig. 1. Terminal Adapter architecture.

The proposed solution for the Terminal Adapter establishes the interface between the network and sensors/actuators and allows identification of devices by the control application. This identification is based on ATM virtual channel and virtual path identifiers (VCI/VPI) at the interface, and on the multiplexing function provided by the adaptation layer. AAL-2 Common Part Sublayer (CPS) packets carry a Channel Identifier (CID) that identifies the channel that is being used. The channels are numbered from 8 to 255, since the values between 0 and 7 are reserved for other purposes. The LI field of CPS packets carries the length of the information field (CPS-INFO) in octets. For each channel, the LI value indicates how many octets have been read from the FIFO (*First In First Out*) of the corresponding input. This value can vary, for each channel, at each reading process, since it depends on the number of octets available in the input FIFO. Moreover, the User to User Indication (UUI) field can serve two functions: to carry specific information in a transparent way, through the CPS sub-layer, and to distinguish between a Service Specific Convergence Sublayer (SSCS) entity (in case its value is between 0 and 27) and the management layer. In this work, the UUI field is used to address the Terminal Adapter and to implement a mechanism for identification of the traffic class associated with the flow carried by the CPS packet.

As soon as they are created, CPS packets are placed in an intermediate FIFO. Here, the scheduling of the input flows had already been made. These packets have already defined the channel identifier (CID), the identification of the Terminal Adapter where they come from (UUI), and the priority associated with the traffic class, which is assigned at the input (UUI) by the configuration module. CPS packets, possibly of different sizes, are concatenated and placed in blocks of 48 octets (ATM_SDUs), which are encapsulated in ATM cells.

In order to organize the transmission scheduling, each packet has a time-stamp associated to allow controlling the delay in the Terminal Adapter. Since some input traffic can have stringent delay requirements, a packet cannot wait more than a well-defined time interval. Thus, if the value of this time interval is too low, the ATM cells will be only partially filled, since the arrival ratio of CPS packets is low compared to the multiplexing clock. Otherwise, ATM cells will be totally filled but the packetization delay will increase.

III. Traffic Scheduling

To organise the aggregation of information flows coming from the different traffic sources that compete for the available transmission capacity, a number of scheduling algorithms were analysed and evaluated. A scheduler establishes the order in which flows are served such that the QoS requirements of each information source are satisfied.

The most basic algorithm consists in placing packets of the different flows into a single FIFO memory structure, such that packets are served in the order of arrival. This FCFS (*First Come First Served*) algorithm is rather simple to implement but does not allow isolation and differentiation among classes, nor even fairness among flows of the same class, that is, it does not provide QoS guarantees.

In order to overcome these limitations, other algorithms that support scheduling of asynchronous traffic have been proposed and are described in the literature. Many of them try to emulate an ideal algorithm known as GPS (*Generalized Processor Sharing*) or FFQ (*Fluid Fair Queuing*). In logical terms, this algorithm assigns a queue to each information flow and on each round simultaneously serves an infinitesimal amount of information from each non empty queue. In this way, in a given finite time interval each queue is visited at least once. A different weight can be associated to each flow, thus allowing the amount of data served to be proportional to its weight.

This algorithm can be implemented, in a simple form, by means of a round-robin (RR) mechanism, where the several

queues are served sequentially and in the same way than in GPS. However, this mechanism serves an information packet at each time instead of an infinitesimal amount. This algorithm is a good approach to GPS when the flows have the same weight and the packets have the same length. WRR (*Weighted Round-Robin*) is a variant of the basic RR mechanism, where the flows are served in the ratio of their weights [6].

WFQ (*Weighted Fair Queuing*) emulates GPS in a more precise way, especially when packets are of variable size [7], but is more complex to implement than WRR. WFQ assigns a time-stamp to each packet, which corresponds to the instant when the packet would have completed service in GPS, and serves packets in the order of their time-stamps. This algorithm is adequate to real-time traffic, but the assigned bandwidth varies inversely with the connection delay, thus becoming less efficient when low delays and high bandwidth utilisation are needed [8].

For scheduling of real-time traffic other algorithms have been proposed, such as EDD (*Earliest Due Date*), also known as EDF (*Earliest Deadline First*). It assigns a deadline to each packet, which is used by the scheduler to define the order of service. A packet whose deadline is more close to the arrival instant has a smaller queuing delay than packets that have been assigned a more distant deadline. Depending on the load, it cannot be possible to serve all the packets before reaching their assigned deadlines.

IV. Scheduling Algorithms

A number of specific Scheduling Algorithms suitable for the Terminal Adapter have been analysed. These algorithms must satisfy two main goals, when performing traffic aggregation: guarantee a bounded delay for services with time critical requirements, and keep high transmission efficiency.

The scheduler is a functional block that belongs to the sending module of the Terminal Adapter and is responsible for scheduling the input information flows, taking into account the delay requirements of each service.

Fig. 2 illustrates the traffic scheduling mechanism, where the multiplexing structure of the AAL-2 CPS packets is used.

The parameters of each traffic class, which depend on the service characteristics, allow the implementation of a priority mechanism to efficiently serve the FIFOs associated with each traffic source. According to the traffic class, the scheduling algorithm must implement a priority mechanism in order to satisfy the delay requirements of each service, and at the same time optimize the efficiency when assembling CPS packets.

At each instant, the scheduler tries to read from the selected FIFO the maximum possible number of octets (45, according to the AAL-2 specification) in order to completely fill a CPS packet, thus keeping the overhead at the minimum. In case all FIFOs are empty, the scheduler will not assemble any CPS packet and the algorithm returns to the starting point.

The guarantee of a maximum delay for a given service is achieved by means of a *Loopclock* value. However, a low *Loopclock* value means poor multiplexing efficiency, due to the creation of a small CPS packet. In this way, the scheduling mechanism must estimate the *Loopclock* value as function of the traffic parameters of the input sources in order to guarantee the maximum delay requirement of each source and to maximize, at the same time, the multiplexing efficiency.

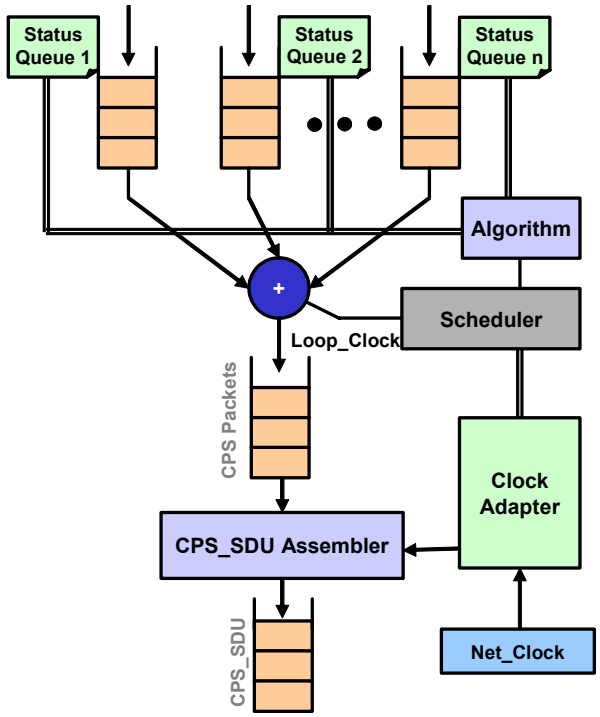

Fig. 2. Terminal Adapter traffic scheduling mechanism.

Due to the fact that low bit rate flows of MD class may produce small amounts of information compared to the flows of other classes, and to the necessity of giving some priority to these services, it might not be possible to fill the corresponding CPS packets with the maximum allowed size. In these cases loss of efficiency can occur, since the need to satisfy the delay requirements of these services imposes the assembly of smaller packets,

Taking into account these considerations, several algorithms were specified and evaluated by means of simulation, in order to select a solution that optimizes the QoS performance of all services supported. The details of the simulator are presented in [9].

The next four sections (A, B, C and D) describe some basic algorithms that help understanding the global functioning of the blocks responsible for scheduling the various information flows. These algorithms do not use QoS parameters to schedule the information. On the other hand, Section E specifies a new algorithm that uses QoS parameters as an input to the scheduler, with the aim of optimizing the multiplexing efficiency and minimizing the delay associated with time critical service flows. This algorithm is called *Traffic Class Oriented Algorithm* (TCOA) and, as will be shown in section

V, is an efficient solution to guarantee QoS to individual service flows as well as to promote service integration; in this way, it allows optimizing the use of bandwidth and thus reducing the associated transmission costs.

A. FIFO Level Criterion Algorithm (FLCA)

At each instant, it calculates the length of each FIFO. The priority is given to the FIFO with the highest occupation level. The scheduler tries to assemble a packet with a maximum size (45 bytes) in order to maximize the multiplexing efficiency. However, it is not always possible to reach this packet length due to the fact the FIFOs may not contain this amount of information. This method does not take into account the deadlines associated with time critical service flows, which may degrade the QoS guarantees required by these services. Despite these characteristics and due to its implementation simplicity, this algorithm may be adequate in many applications.

B. Oldest Byte Criterion Algorithm (OBCA)

This algorithm is an approximation of WFQ introduced in section III. At each instant, it reads the time-stamp of the first byte of each FIFO (byte at the head of the FIFO). It serves the FIFO whose time-stamp is the oldest. Like the previous method, the scheduler tries to read the maximum number of bytes in order to achieve maximum efficiency. Although this algorithm does not take into account the deadlines of each information flow, the scheduler mechanism is based on a time criterion. Due to the fact that the oldest byte does not always belong to a time sensitive flow, this algorithm does not minimize the delays associated with these services.

C. Sequential Criterion Algorithm (SCA)

This is a simple algorithm that is a variant of FFQ introduced in section III. It serves each FIFO in a round-robin fashion and, once again, it tries to read the maximum number of bytes in order to attain maximum efficiency. This method is not suitable for the majority of applications since it does not take into account any QoS related parameter. Nevertheless, it may be adequate for some applications where time related parameters are not relevant for scheduling purposes.

D. Random Criterion Algorithm (RCA)

This algorithm is identical to SCA previously described, but the scheduler is random instead of being sequential. As will be showed in section V, the performance of this algorithm is identical to SCA.

E. Traffic Class Oriented Algorithm (TCOA)

Unlike the previous algorithms, the proposed TCOA uses QoS parameters as an input to the scheduler. Thus, associated with each information flow there is a FIFO and a state table that keeps the following parameters:

- FIFO sizes in octets,
- Traffic class of each flow,
- Time-stamp of the FIFO oldest octet (octet at the head of the FIFO),
- For the MD class - maximum delay that guarantees in time delivery of the information.

Taking into account these parameters, the scheduler performs the following algorithm:

1. It calculates the service instants (deadlines) of MD class FIFOs, as a function of their maximum delays,
2. While the deadlines are not reached, it sequentially serves the largest FIFO (it can be of either class MD or D),
3. When the deadline of one of the MD class FIFOs is reached, it must be served,
4. If there is no information in the MD or D class FIFOs, it transmits the packets of the largest ME class FIFO.

Thus, while the service instants of the MD class packets are not reached, the priority criterion is based on the selection of the FIFO that has, in a given instant, the largest number of octets. The ME class FIFOs are served only when there is no information in the FIFOs of the remaining classes.

V. PERFORMANCE ANALISYS

The system evaluation was based on a simplified test-bed developed for this purpose. It consists of a set of simulation programs [2] written in C Language, which include models of artificial sources that allowed the creation of specific test scenarios, difficult to obtain with real sources. Performance was evaluated in terms of:

- Multiplexing efficiency,
- Maximum delays and queue sizes of the different flows,
- Maximum bandwidth of the output aggregate flow.

The tests carried out for the Terminal Adapter were based on two concrete scenarios that were used to evaluate, respectively, the capability of integrating flows associated with services with different delay requirements (Scenario-1), and the capability to support a high number of low bit rate flows in an efficient way (Scenario-2).

Table I shows the characteristics of the sources used in both simulation scenarios.

TABLE I
SIMULATED TRAFFIC SOURCES.

Source	Bit rate (average)	Type	Class
S1 - Sensor	800bit/s	Constant	MD
S2 - Data	16kbit/s	Variable	D
S3 - Actuator	80bit/s	Random	MD
S4 -Voice	16kbit/s	Constant	MD
S5 - Video	80kbit/s	Variable	ME
Total	112 880 bit/s		

MD class is assigned to Sensor, Actuator and Voice sources, D class to Data sources and ME class to Video sources. Each MD class flow will have an associated target delay, which is a function of its specific service characteristics. Voice sources are the most sensitive to delay, while Video sources load the system with the highest amount of data.

Scenario-1 uses all Sources listed in Table I and Scenario-2 only uses Sensor (S1) and Data (S2) sources, since it is expected that these will be dominant in the control applications addressed in this paper. The target delays defined for S1, S3 and S4 flows were, respectively, 200ms, 50ms and 15ms.

A. Simulation of Basic Algorithms

To evaluate these algorithms scenario-1 was considered, since it is only important to focus on the basic functioning of the mechanism responsible for scheduling the information. Thus, in this scenario the system is loaded with a set of traffic sources that exhibit a broad variety of characteristics in terms of bit-rate and delay requirements.

For each algorithm, simulations with different *Loopclock* values were carried through. As stated before and will be confirmed later on, the value of this parameter influences the multiplexing efficiency and the scheduling delay in such a way that a compromise is required.

Figures 3 to 6 show the variation of the maximum delay as a function of the *Loopclock* parameter for each basic algorithm.

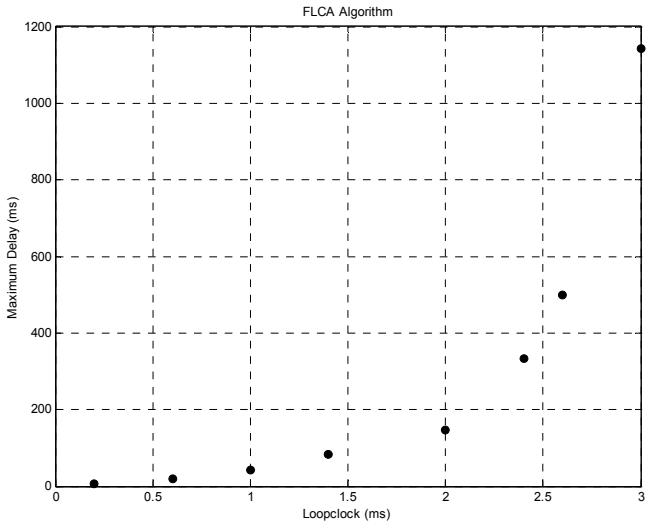

Fig. 3. FLCA simulation: Maximum Delay versus *Loopclock*.

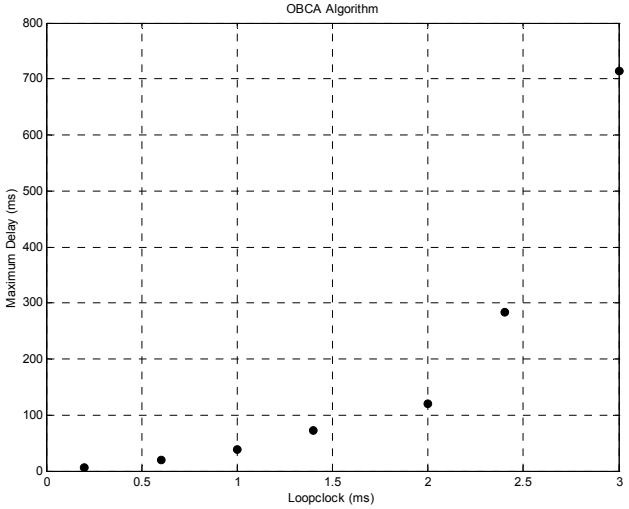

Fig. 4. OBCA simulation: Maximum Delay versus *Loopclock*.

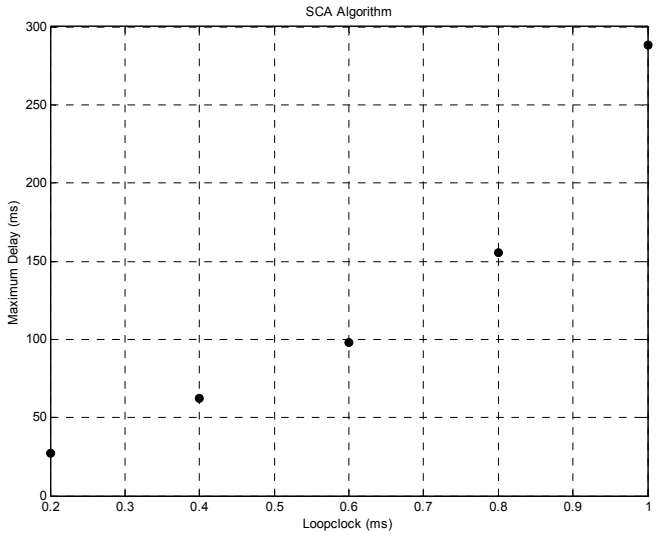

Fig. 5. SCA simulation: Maximum Delay versus *Loopclock*.

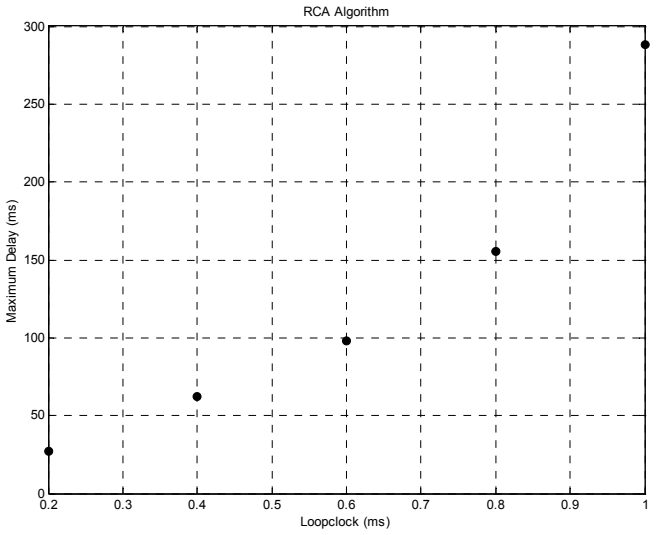

Fig. 6. RCA simulation: Maximum Delay versus *Loopclock*.

The first two algorithms show a better performance than the other two. This fact was expected since they use, as input, parameters that influence the delay. For *Loopclock* values below 3ms, these two algorithms produce very small delays, being FLCA slightly better.

Fig. 7 shows the variation of the Overhead versus *Loopclock* for the four Basic Algorithms. Looking at the figure, it can be observed that the multiplexing efficiency increases with the value of the *Loopclock*. This fact was also expected as well as the better performance of FLCA relatively to OBCA, for values above 2ms.

FLCA has the better performance in terms of multiplexing efficiency and delay. This fact was important for dimensioning TCOA, since FIFO size is one of the key input parameters to the scheduling mechanism.

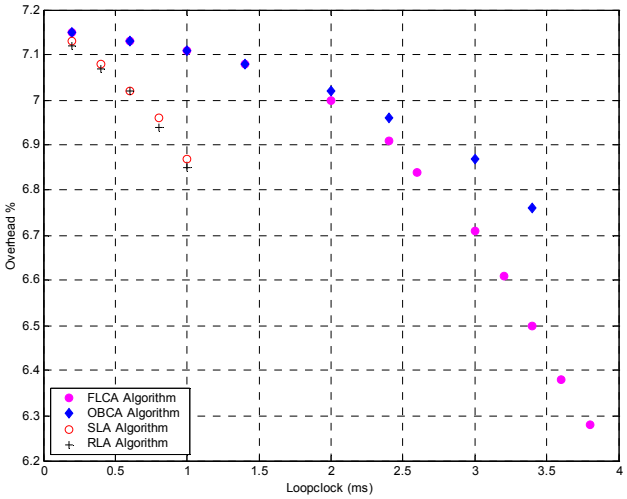

Fig. 7. Variation of the Overhead versus *Loopclock* - Basic Algorithms.

B. Simulation of TCOA

Table II shows some simulation results obtained with TCOA tests in scenario-1.

TABLE II
PERFORMANCE OF THE TCOA ALGORITHM: SCENARIO-1.

	Maximum Delay (ms)					
***Loopclock* (ms)**	**S1**	**S2**	**S3**	**S4**	**S5**	**Overhead**
0.2	0.2	1.6	0.2	0.8	3.4	18.73%
1.0	1.0	9.0	1.0	4.2	21.9	18.54%
2.4	2.4	21.4	2.4	9.8	151.7	18.44%
3.0	3.0	29.9	3.0	14.8	632.4	18.39%

Each row of the table depicts simulation results for a specific *Loopclock* value, represented in the first column. Columns S1 to S5 indicate the maximum delays of the corresponding service flows of Table I. This table shows three important features of the TCOA behaviour:

- MD target delays were not overtaken,
- High multiplexing efficiency (overhead),
- Capability to integrate services.

Table III shows results obtained in scenario-2. The first column indicates the ratio between S1 and S2 sources. Since S2 sources (D class) do not impose time constraints to the scheduler, the multiplexing efficiency will increase with the S2/S1 ratio.

TABLE III
PERFORMANCE OF TCOA ALGORITHM: SCENARIO-2.

		Maximum Delay		
S2/S1	***Loopclock* (ms)**	**S1 (ms)**	**S2 (ms)**	**Overhead**
5 / 5	4.1	20.5	2054	25.73%
10 / 5	2.0	10.5	1047	14.35%
15 / 5	1.4	7.0	691	9,02%
20 / 5	1.1	5.5	964	5.9%

On the other hand, the delay associated with S1 sources decreases when the number of S2 sources increases. This feature is important since the number of S2 sources is usually dominant in these types of applications, helping the scheduling mechanism to improve its performance.

Two of the described basic algorithms (FLCA and OBCA) could be used in some control applications that require the integration of different types of services. However, they are not capable of controlling the delay of flows that will be aggregated for transmission over a network. On the other hand, TCOA not only retains the main qualities of these, but also implements an efficient scheduling mechanism that, at the same time, controls the delay of the services with time constraints and keeps the multiplexing efficiency high.

VI. CONCLUSIONS

In this paper we have analysed and evaluated several Scheduling Algorithms to support QoS and Service Integration in Sensor and Actuator Networks. The Scheduling Algorithm is the core of the statistical multiplexing mechanism that aggregates information flows, exchanged among Fieldbus devices, based on delay requirements and other QoS parameters.

The simulation results showed good performance of the proposed TCOA (*Traffic Class Oriented Algorithm*), both in terms of meeting the target delays of the input sources and multiplexing efficiency.

As a final conclusion, it can be stated that this mechanism is appropriate to aggregate different kinds of traffic. In particular, it allows the integration of low bit rate flows produced by control applications in actual communication networks, such as ATM.

REFERENCES

[1] C. Lu, B. Blum, T. Abdelzaher, J. Stankovic, and T. He, RAP; "A Real-Time Communication Architecture for Large-Scale Wireless Sensor Networks"; Real-Time Technology and Applications Symposium; 2002.
[2] Cabral, J.M.; "A System Architecture for Low Bit Rate Traffic Aggregation in Control Applications", PhD Thesis (http://hdl.handle.net/1822/3325), University of Minho, 2005.
[3] McDysan, D. E., Spohn, D. L., "ATM Theory and Application", McGraw-Hill Series on Computer Communications, 1995.
[4] ITU-T, Rec. G.114, "One-way transmission time", May, 2000.
[5] ITU-T, Rec. I.363.2 – "B-ISDN ATM Adaptation Layer Specification: Type 2 AAL", September 1997.
[6] Keshav, S.; "An Engineering Approach to Computer Networking", pp. 209-246 Addison-Wesley, 1997.
[7] Demers, A., Keshav, S., Shenker, S.; "Design and Analysis of a Fair Queuing Algorithm", Proceedings of ACM SIGCOMM'89, 1989.
[8] Parekh, A., Gallager, R.; "A Generalized Processor Sharing Approach to Flow Control in Integrated Services Networks – The Multiple Node Case", IEEE/ACM Transactions on Networking, pp. 137-150, 1994.
[9] J.M.Cabral, J.G.Rocha, J.E.Neves and J.Ruela; "ATM Terminal Adapter and Concentrator implemented in PC with Linux", Technical Report (http://hdl.handle.net/1822/2830), R&D Centre Algoritmi, University of Minho, 2005.