REAL TIME SCHEDULING METHODS REQUIREMENTS IN DISTRIBUTED CONTROL SYSTEMS

Pau Martí*, Ricard Villà*, Josep M. Fuertes*, Gerhard Fohler**

**Dep. of Automatic Control*
Universitat Politècnica de Catalunya
Pau Gargallo 5, 08028 Barcelona SPAIN
Ph.: 34-3-4016971, Fax: 34-3-4017045, Email: {pmarti,villa,pepf}@esaii.upc.es

*** Dep. of Computer Engineering*
Mälardalen Högskola
PO Box 883, SE-721 23 Västerås, SWEDEN
Email: gerhard.fohler@mdh.se

Abstract: Distributed control systems involve three main disciplines: control systems, real time systems, and communication systems. Control systems, due their stringent timing constraints, demand real time computing technology. Distributed control systems need communication systems when distributing sensors, actuators, the control procedures and data messaging. In general, demands of distributed control systems and properties of real time scheduling algorithms differ, for example, for activation patterns of tasks. The aim of this paper is to provide a set of requirements to overcome current limitations of real time scheduling methods to their increase applicability for distributed control systems. *Copyright © 2000 IFAC*

Keywords: Real-time systems, Real-time tasks, Scheduling algorithms, Distributed computer control systems, Control applications, Timing analysis

1. INTRODUCTION

An increasing number of complex systems relying, in part or completely, on control systems, need real time computing. It is known that many control applications constitute real time systems due to their strict timing constraints. Nevertheless, it has been pointed out by Törngren (1995) that control systems have been mostly treated by control engineering methods while real time systems have been mostly treated by computer engineering methods. Therefore, there is a gap between the above disciplines due to their different theoretical and practical backgrounds when dealing with real time control systems.

Real time computing has been widely used in many areas during the last three decades, playing a key role in technology. There are many definitions of real time computing, but basically, the main idea suggested by Stankovic (1988) is that in such a computing system, the correctness of the system depends not only on the logical results of the computations but also on the time at which the results are produced.

In real time systems, scheduling theory addresses, by means of algorithms, the problem of meeting the specified timing requirements in order to have an understandable, predictable and maintainable system timing behaviour. Therefore, as said by

Ramamritham and Stankovic (1994) scheduling involves the allocation of resources and time in such a way that certain performance requirements are met.

Surprisingly, control theory and real-time scheduling theory have been relatively independent research areas. Recently, some works have appeared in the literature which address important issues in real time distributed control systems. Nilson (1998) discusses some problems that can be found in real time control, Törngren (1998) revises fundamental aspects for implementing real time control applications in distributed control systems, and Sandstrom (1999) studies mono-rate and multirate control systems and scheduling issues. Wittenmark, *et al.* (1995) and Wittenmark, *et al.* (1998) investigate the effects of time varying delays on control system stability and performance. It is shown that time varying delays can cause instability and deteriorate performance. Shin and Cui (1996) and Wittenmark, *et al.* (1995) give an interpretation of time varying delays as computer induced disturbances.

Related to scheduling and control systems, task schedulability in control systems has been treated by Seto, *et al.* (1996), where tasks frequencies are optimised in order to make all tasks schedulable and to enhance control system performance. Also, Shin and Meissner (1999), adjusting task periods and task reallocation, adaptation and graceful degradation of control system performance is optimised.

A starting point for studying both real time systems and distributed control systems and trying to overcome the gap between them is testing real time scheduling methods on distributed control systems. From previous similar works by Sandstrom (1999) and Törngren (1998), this paper aims, on one hand, to point out some key issues that can have different interpretations (meanings) for real time engineers and control engineers when facing the suitability of real time scheduling methods in distributed control systems. On the other hand, this paper derives from distributed control systems characteristics a set of real time scheduling methods requirements that will be useful for evaluating the suitability and applicability of such methods in real time distributed control systems.

This paper is structured as follows: in section 2, an overview on distributed control systems timing analysis is given. Section 3 revises well-known real time scheduling concepts. Key new challenging research issues are identified in section 4 when trying to bridge the gap between real time systems and distributed control systems. Section 5 derives a set of suitability indicators from distributed control systems features and section 6 shows an example of an unsuccessful application of two real-time scheduling algorithms on a control application regarding a control requirement. Finally, some conclusions are drawn in section 7.

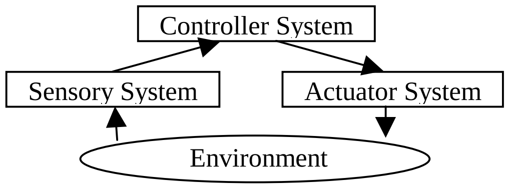


Fig. 1. Conceptual control system

2. CONTROL SYSTEMS TIMING ANALYSIS

A distributed control system has basically three main subsystems: sensory system, controller system and actuator system. Each of theses subsystems can be physically divided into separate units. The aim of these three subsystems working together is to perform some control actions by means of a control procedure, on an real environment or plant (see Fig.1).

The general functionality of a (distributed) control system can be described as follows: firstly, the sensory system collects data from the environment to be controlled. Secondly, the control system, by means of a control law, processes this data and derives the needed action. Finally, the actuator system performs the action on the environment.

The previous functional scheme can be split into more detailed activities that affect with different degrees of strictness the system timing behaviour and its performance.

It is important to stress that strictness in control systems usually means that actions perform at exact time instants. However, in real time systems, strictness means that actions have to complete their computations before a time constraint. In both cases, a common fact is that missing an action result may imply severe consequences to the system.

Moreover, it has to be pointed out that control theory and practice can follow many paradigms such as continuous or discrete control; centralised or distributed control; direct, feedback or feed-forward control; mono-rate or multirate control; classic, adaptive or fuzzy control. One or many of these paradigms are found in any control system, implying different timing behaviours.

Therefore, time is an important issue in control systems regarding timing behaviour and schedulability as well as performance and quality of service. Control theory assumes a highly deterministic timing of an implementation, as described by Åström and Wittenmark (1997). Consequently, few works (Shin and Cui, 1996), (Wittenmark, *et al.*, 1995) and (Wittenmark, *et al.*, 1998) have treated deficiencies in the computer system implementation of the control system with respect to time-variations and time-restrictions.

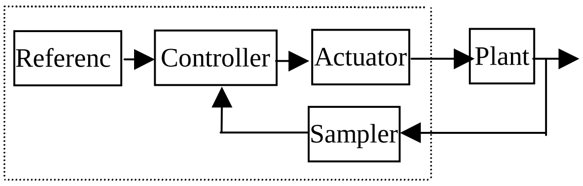


Fig. 2. Computerised distributed control system

In order to provide a set of indicators to study the suitability of real time scheduling methods in distributed control systems, we have to derive and specify timing behaviour of such control systems.

Lets take a closer look at the significant timing behaviour of a simple sampled computerised distributed feedback control system in order to have a deeper understanding about which timing specifications should be derived from control systems (see Figure 2).

It consists of four main nodes, the reference signal generator, the sampler node, the controller node and the actuator node. In the typical control scenario, the following functionality is expected: the sampler samples the plant with a fixed period T_{sample}. The time of the n^{th} sampling is given by

$$t_{sample}(n) = t_{sample}(n-1) + T_{sample} \qquad (1)$$

When the sampler has collected the data it is forwarded to the controller, introducing a communication delay D_{sc}. The n^{th} controller start execution time is given by

$$t_{control}(n) = t_{sample}(n) + D_{sc} \qquad (2)$$

The controller executes the control procedure in order to derive the actuation signal(s), introducing a computing delay T_c. The controller will forward the actuation signals to the actuator, introducing another communication delay D_{ca}. Finally the actuator performs the actuation at time given by

$$t_{actuate}(n) = t_{control}(n) + T_c + D_{ca} \qquad (3)$$

The reference signal generator generates reference values every T_{ref}, which usually is larger than T_{sample}.

In this short timing description, it is shown that the controller executes every T_{sample} and introduces a computing delay T_c. It is supposed that the sampler and the actuator perform without any computation time cost at a given time. It may be necessary to introduce an acceptable deviation (tolerance) from the given time of the sampling (tol_s) and for the actuation (tol_a) in order to be more realistic, modifying (1) and (3) as in (4) and (5)

$$t_{sample}(n) = t_{sample}(n-1) + T_{sample} \pm tol_s \qquad (4)$$

$$t_{actuate}(n) = t_{control}(n) + T_c + D_{ca} \pm tol_a \qquad (5)$$

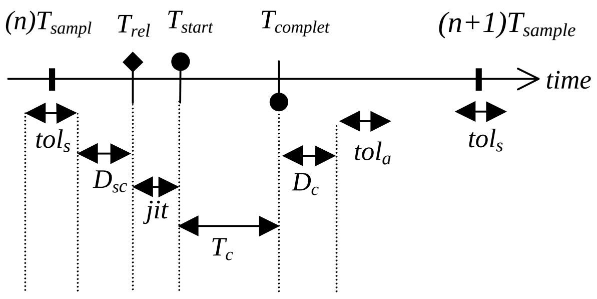


Fig.3. Timing diagram of a typical control loop

Moreover, the controller timing analysis could be more precise as well as complicated if release time (T_{rel}), start time (T_{start}), jitter (varying or fixed difference between T_{rel} and T_{start}, jit) and completion time ($T_{complet}$) of the controller activity were introduced, modifying (2) and obtaining (6).

$$t_{control}(n) = t_{sample}(n) + D_{sc} + jit \qquad (6)$$

The analysed system has two inter-task communication procedures that introduce delays (fixed or varying) at the timing behaviour.

Finally, the whole calculating time, from the sampling to the actuation of any instance of the simple control system under study, can be given by (7).

$$\text{ControlDelay}(n) = t_{actuate}(n) - t_{sample}(n) \qquad (7)$$

A full-specified illustration of the previous analysis can be seen in the Fig.3. Similar detailed analysis was done by Sandstrom (1999) and Lönn and Axelsson (1999).

The previous analysis will become more complex if we add the case of multiple instances for each task. For example, both sampling and actuation tolerances (see e.g. in (8)) can add time variability in one case, but not in other, providing different control delay calculation time for every task instance. Also, the analysed timing constraints will have different degrees of strictness depending on every task instance.

$$t_{sample}(n) = t_{sample}(n-i) + T_{sample} \pm tol_s * i \qquad (8)$$

Another point that has to be considered is that the system timing behaviour sets precedence relations between the different activities (time triggered activity when trigger by period, event triggered actions when trigger by the completion of other activities) as well as resource allocation that will be used when scheduling the system.

3. REAL TIME SCHEDULING BASIC CONCEPTS

Real time systems are characterised by a set of concurrently executing tasks that have deadlines, which represent the time at which the task should complete its execution so as not to cause damage to the system.

Depending on the consequences of a missed deadline (deadline miss tolerance), two types of real time systems can be differentiated. Hard real time systems are those where it is imperative that responses occur within the specified deadline while soft real time systems are those where response times are important but the system will still function if deadlines are occasionally missed. In fact, it has to be pointed out that there is no complete agreement on the previous definitions. Sometimes, if the consequences of missing a deadline are catastrophic, the target system is called critical (or safety critical) real time system instead of hard real time system.

Table 1. Parameters for a real time task τ_i

Parameter	Description
Criticalness (hard/soft)	Parameter related to the consequences of missing deadlines
Worst case computation time (C_i)	An estimation of the maximum time required by the processor for executing the task without interruption
Period (T_i)	Time between consecutive tasks releases
Deadline(D_i)	Time before which the task should be completed
Priority (P_i)	Relative importance (for scheduling purposes) of the task with respect to the other tasks in the system
Starting time (s_i)	Time at which a task starts its execution
Finishing time (f_i)	Time at which a tasks finishes its execution
Worst case blocking time (B_i)	Maximum time during which the task may be blocked by lower priority tasks when accessing a resource
Interference (I_i)	Maximum time that the task will have to wait due to preemption from higher priority tasks
Processor utilisation (U_i)	Processor utilisation of the task (equal to C_i/T_i)
Worst case response time (R_i)	Longest time duration between the invocation of a task and the time that the task completes
Release time (r_i)	Time at which a task becomes ready for its execution
Jitter (J_i)	Time that a task spends from its release until its start time
Lateness (L_i)	Difference between a task completion time and its deadline
Tardiness (E_i)	Time during which a task stays active after its deadline
Laxity (slack time) (X_i)	Maximum time a task can be delayed on its activation to complete within its deadline

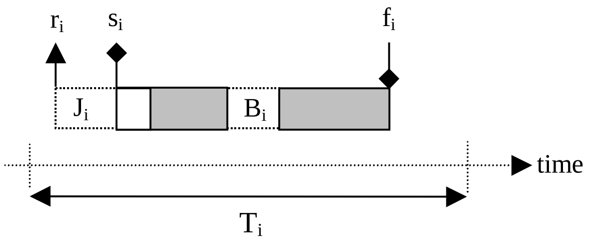


Fig. 4. Some parameters for a real time task τ_i

Another timing characteristic that can be specified on a real time task concerns the regularity of its activation. Depending on it, a task is defined as a periodic or aperiodic task. A task that is released at regular intervals is a periodic task. On the other hand, a task that is not invoked at regular intervals is an aperiodic task. To allow worst-case calculations to be made there is often defined a minimum period between any two aperiodic events (from de same source). If this is the case, the task involved is said to be sporadic.

In general, a real time task τ_i can be characterised by several of the parameters described in the following table, which can be directly task attributes or attributes obtained from current scheduling methods.

Figure 4 shows some of the parameters of a real time task. To characterise all tasks in any distributed real time system, the above parameters should be combined with other models in order to specify resource constraints, concurrency relations, precedence relations, communication patterns and placement constraints that any set of real time tasks may have.

Since the early work by Liu and Layland (1973) on real time scheduling, many scheduling algorithms and methods have been presented.

In Tindell and Hansson (1997), the scheduling approach is broken down into three main activities: off-line configuration, run-time dispatching, and a priori analysis. The off-line configuration generates the static information that will be used for the run-time dispatching. The run-time dispatching deals with the switching between computations for different events at run-time. The a priori analysis examines the system and tells us if all the timing requirements will be met according to the off-line configuration information and the behaviour of the run-time dispatching algorithm. Scheduling approaches can differ on how much effort they spend in every activity.

The a priori analysis is used to determine by means of tests the feasibility of scheduling approaches, i.e. whether the temporal constraints of all processes will be met at run time. The analysis is said to be sufficient and necessary when a task set will meet all its deadlines if, and only if, it passes the test. If passing the test means that the task will meet all its deadlines at run time, but failing the test does not imply that the task will not meet all its deadlines, the analysis is said to be sufficient and not necessary, as Audsley, *et al.* (1995) showed.

Among the great variety of real time scheduling approaches, four main classes are identified by Ramamritham and Stankovic (1994): off-line vs. on-line, preemptive vs. non-preemptive, static vs. dynamic and optimal vs. heuristic scheduling. Obviously, any scheduling algorithms may belong to more than one of the identified classes.

4. NEW CHALLENGES

Real time scheduling has provided general-purpose algorithms that mainly use general task models to express timing requirements. Control systems have specific timing requirements that should fit into these task models.

But control problems increase when control algorithms are implemented in distributed real time systems, as Nilsson (1998) stated. In these cases, timing analysis has to cover a broader set of aspects, such as allocation of tasks to processors, task synchronisation (precedence and resource constraints) and task communication over field-buses or networks.

Also, Stankovic (1998) stressed that further research is also needed on open and non-dedicated real time systems and globally distributed real time systems, as well as on embedded systems. Economic and safety issues must be considered. Therefore, complex control systems designers will have to look close on real time research results.

When studying the suitability of real-time general-purpose algorithms for dealing with control systems timing requirements, some key research issues are identified:

- The jitter problem: most real time scheduling algorithms are concerned with meeting temporal constraints, such as deadlines. An earlier than required completion is acceptable. Control applications, however are concerned with the completion time itself, e.g., by requiring completion within a time interval or bounding the differences in starting or completion times.

- Real time period vs. control period: related to the jitter problem, there is a difference between period interpretation in real time systems and control systems. In the former, the period is the time between consecutive task activations, during which some operation must be performed. Future activation instants are predefined so the period is constant. In control systems, the period is the time between consecutive performed operations, each measured from the last one. Therefore, in control systems, the time lost or gained at each period due to jitter, can go accumulating without bound.

- Interaction between tasks with different periods: in the real time scheduling scenario, it is usually supposed that when two or more task interact, they have the same period. However, it is known that control systems have tasks that perform at different periods and have tight interactions.

- Distributed precedence constraints: to schedule tasks with precedence constraints in a distributed real time system is a problem not yet solved. Moreover, precedence relations have been modelled using directed acyclic graphs; however, control systems have by their nature cyclic precedence relationships between tasks, e.g., in a feedback control loop.

- Communication / precedence relations: real time scheduling has considered that communication between real time tasks implies precedence relations. However, control systems may have communication patterns that do not necessarily imply precedence relations. Therefore, communication and precedence analysis should be carried out separately.

The above-identified points show that the same word (i.e. “period”) can have different meanings and some assumptions may be different between the two systems. Therefore, real time scheduling and the corresponding schedulability analysis should be extended in order to be used in control theory.

5. SET OF SUITABILITY INDICATORS

Prior to test real time scheduling methods in distributed control systems, a set of indicators has to be identified. This set of indicators will be different from requirements of normal real time scheduling algorithms mainly because they have to be included in applications that will schedule real (distributed) control tasks implemented in the available limited computer resources and that will interact with a changeable physical world. Therefore, the new set of requirements will be characterised by the variability of control task periods, by their tolerances and by their different instances. Flexible and reliable timing constraints will have to be handled by new real time scheduling techniques.

Standard scheduling methods cannot handle the new set of indicators because standard scheduling methods are essentially based on a static and limited number of timing constraints where variability can not be expressed. As a consequence, the inherent flexibility of some task timing constraints is abandoned when expressing them in standard static task models. Therefore, in the use of standard methods, flexibility in temporal demands can not be efficiently exploited.

The characteristics of distributed control (DC) systems examined to derive a set of new real time scheduling methods requirements (indicators) are:

- Uninterrupted operation: DC systems operate without stopping using control algorithms discretely expressed and with a changing environment, in an asynchronous or synchronous interaction operation mode.

- Distributed operation: DC systems allow distributed processing, communication and synchronization services, multiple threads of control interaction and non blocking access to shared resources, working with different degrees of intelligence and different degrees of coupling.

- Clock synchronization: a global time is distributed to system components.

- Multi-vendor systems: the components of DC systems are developed and manufactured by different companies. This fact implies that modularity strategies have to be followed when building these systems.

- Reliability: DC systems are reliable. When a distributed control system is implemented, system performance requirements are expected to be always achieved.

- Fault tolerant operation: DC systems have the ability to tolerate faults that occur during system operation. Therefore, failsafe components must be used. Otherwise, degradability (controlled decrease of performance under fault operation) must be guaranteed.

- Stringent demands on the system responsiveness: the reaction of DC systems in front of external or internal stimuli must be in time.

From the above characteristics, the following list presents a set of suitability indicators in order to be able to evaluate real time scheduling methods in distributed control systems. Probably when a real time scheduling method includes these indicators, we can assume that it is well suited for scheduling distributed control systems.

Periodic and aperiodic task management: tasks in a distributed control system have periodic and aperiodic behaviours. Therefore, the scheduling approach has to be able to manage both kinds of tasks.

Event and time triggered task activation: in a distributed control system, time triggered task activation (e.g. sampling) and event triggered task activation (e.g. alarm) are needed. The scheduling approach has to allow the representation of such activation modes.

Quantitative and qualitative priority assignment: in overloaded system conditions some qualitative priority assignment is necessary, while in not overloaded conditions, quantitative priority assignment may be sufficient. The scheduling approach has to provide those types of assignments.

Task allocation: when distributing the control procedures, task allocation is needed. The scheduling approach has to take into account the allocation process, providing a global scheduling which will allow task scheduling over CPUs, networks, I/O modules and embedded devices.

Inter-task relations: a set of concurrently executing tasks in a distributed control system implies that synchronization services, precedence constraints and communication needs must be considered. The scheduling approach has to be able to represent and deal with these interactions.

End to end timing analysis: the scheduling approach has to provide timing analysis of the overall components (and their relations) of a distributed control system.

Resource management: in a distributed control system resources are allocated to tasks. The scheduling approach task model has to include such allocation process, providing schedulable resource constraints information.

Mutual exclusion prevention: task access to shared resources must be in mutual exclusion. The scheduling approach must provide mechanisms to avoid simultaneous access to critical sections.

Deadlock prevention: in distributed control systems inter-task relations between tasks and resource accesses must be deadlock-free. Therefore, the scheduling approach must provide mechanisms to avoid deadlock situations.

Scalability facilities: new requirements for a distributed control system and changes in the controlled environment may suppose changes on the system itself. Composability is needed. The scheduling approach must be reusable and scalable.

Online reconfiguration: online reconfiguration has to be allowed if a distributed control system has to be updated without stopping its operation. The approach must provide flexibility at run time.

Preemption: in distributed control systems preemption between tasks is needed. Therefore, the scheduling approach must allow preemption.

Task-replica facilities: complex distributed control systems should be fault tolerant systems, task-replica facilities should be provided. The scheduling approach has to manage task replicas, or multiple scheduling policies have to be allowed in the same scheduling approach.

Fixed and varying timing constrains: in distributed control systems, some tasks have fixed timing constraints while other tasks can have varying timing constrains. The scheduling approach must work with representations of both timing constrains.

Fulfilling temporal requirements: distributed control systems timing characteristics must be correctly expressed. The scheduling approach task model has to be complete and flexible enough to be able to express such specific timing characteristics.

Predictability analysis: (real time) distributed control systems must be predictable. The scheduling approach has to provide a predictability analysis in order to ensure a correct system behaviour.

Mode change facility: in distributed control systems there are different modes of operation. Therefore, changing scheduling policies must be allowed in the scheduling approach.

Using this set of indicators, a suitability test of any real time scheduling method in distributed control systems is possible. By determining (yes/not) which of the above indicators are included, treated and successfully solved in a real time scheduling method, the suitability of such method in a distributed control system will have been evaluated.

Evaluating current real time scheduling approaches using the presented indicators will provide a useful classification for (real time) distributed control systems engineers as well as a useful set of requirements for new real time scheduling algorithms.

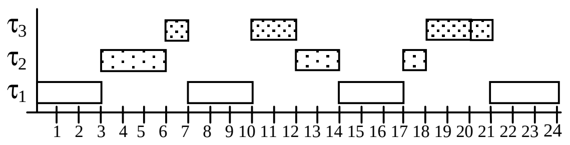
6. AN EXAMPLE

The current task planning algorithms are not enough well suited for control systems as shown in the following example. Let's suppose a control system that includes three tasks. The first one implies a sampling each 12 time units (t.u.) with a computing time of 3 t.u. The other two are other control tasks, one with a period of 7 t.u. and 3 t.u. of computing time, and the other with a period of 20 t.u. and a computing time of 5 t.u.

Applying rate monotonic scheduling and earliest deadline first scheduling algorithms we obtain a feasible scheduling by using the worst case response time analysis and part of the scheduling schemes obtained are shown in figure 5.

Table 2. Set of tasks

	T_i	C_i	Description
τ_3	20	5	Control task
τ_2	12	3	Sampling task
τ_1	7	3	Control task



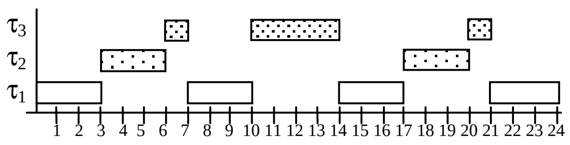


Fig. 5. Partial scheme of the rate monotonic scheduling (top) and earliest deadline first scheduling (bottom) algorithms.

On both schemes (see Fig. 5) the real sampling period, defined by the starting time of task 2, is not constant due to the jitter problem that has been introduced by each algorithm.

This may or may not be acceptable for control purposes. Usually not, because one may get the wrong sample, or no sample at all, implying for example a degradation of the performance of the controlled system, even causing a critical failure of the system.

However, in figure 6 can be seen that there is an alternative way of executing the three tasks while keeping the sampling period constant. For example, fixing the starting time of each instance i of the sampling task (τ_2) at T_{2i}+3 t.u. in every period T_2, the remaining tasks are also schedulable.

The new scheduling has been done heuristically over the hyperperiod (420 t.u.) in order to check the feasibility of the alternative algorithm. Although all the tasks will meet their deadlines (period), no fixed priority assignment has been performed because the main concern was to keep the starting time of the sampling task constant for each period.

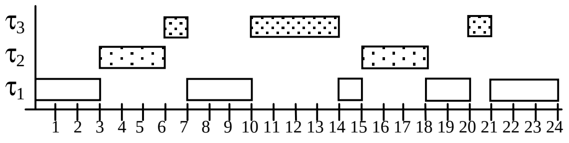


Fig. 6. Alternative planning. The sampling instants (i*period+3 t.u.) of each instance of the sampling task repeat at constant intervals.

7. CONCLUSIONS

In this paper we have identified a a gap between real time systems and control systems. We have given an example showing the kind of problems of scheduling real time control tasks using general purpose scheduling algorithms.

We proposed to model temporal behaviour of control systems and to derive their specific timing requirements in order to be able to assess the suitability of real time scheduling current trends for control systems. A set of suitability indicators has been presented.

These indicators support appropriate scheduling approaches for distributed control systems to handle CPU scheduling, I/O scheduling, embedded devices scheduling and real time communications scheduling. At the same time, the scheduling approach has be local and distributed, and deal with critical and less critical tasks.

Future work will be on evaluating current scheduling approaches using the provided indicators.

REFERENCES

Audsley, N.C., A. Burns, R.I. Davis, K.W. Tindell, and J.Wellings (1995). Fixed Priority Pre-emptive Scheduling: An historical Perspective. *Real-Time Systems, the International Journal of Time-Critical Computing Systems*, **Vol. 8**, Number 2/3

Åström, K. J and B. Wittenmark (1997). *Computer-Controlled Systems. Third edition*. Prentice Hall.

Liu, C. and J. Layland (1973). Scheduling Algorithms for Multiprogramming in a Hard Real-Time Environment. *J.ACM*, **20**, 46-61.

Lönn, H. and J. Axelsson, (1999). A Comparison of Fixed-Priority and Static Cyclic Scheduling for Distributed Automotive Control Applications. *Proc. of the 11th Euromicro Workshop on Real-Time Systems.*

Nilsson, J. (1998). Some Topics in Real-Time Control. *ACC*, June 24-26. Philadelphia.

Ramamritham, K. and J.A. Stankovic (1994). Scheduling Algorithms and Operating Systems Support for Real-Time. *Systems Proceedings of the IEEE*, **Vol. 82**, NO.1, January 1994.

Sandstrom, K. (1999*) Modelling and Scheduling of Control Systems*. Licenciate Tesis TRITA-MMK 1999:5, ISSN 1400-1179, ISRN KTH/MMK/R--99/5-SE. Department of Machine Design, The Royal Institute of Technology, S-100 44 Stockholm. Sweden

Seto, D., J.P. Lehoczky, L. Sha and D.G. Shin (1996). On task Schedulability in Real-Time Control Systems. *RT Systems Symposium, 17th IEEE.* p 13-21

Shin K. and X. Cui (1996). Computing Time Delay and its Effects on Real-Time Control Systems. *IEEE Trans. on Control Systems Technology*. **Vol. 3**, N. 2, p.218-224

Shin, K.G. and C.L Meissner (1999). Adaptation and Graceful Degradation of Control System Performance by Task Reallocation and Period Adjustment. *Real-Time Systems, Proceedings of 11th Euromicro Conference*. P. 29-36.

Stankovic, J.A. (1988). Misconceptions About Real-Time Computing: A Serious Problem for Next Generation Systems. *IEEE Computer*. **21**(10):10-19.

Stankovic, J.A. (1998). Real Time and Embedded Systems. *ACM.*

Tindell K. and H. Hansson (1997). *Real Time Systems by Fixed Priority Scheduling*. Technical report, Departament of computer systems, Uppsala University.

Törngren M. (1995). *Modelling and Design of Distributed Real-Time Control Applications*. PhD thesis, Dept. of Machine Design, The Royal Institute of Technology, Stockholm, Swedwn.

Törngren, M. (1998) Fundamentals of Implementing Real-time Control Applications in Distributed Computer Systems. *J. of Real-Time Systems*, **14**, 219-260, Kluwer Academic Publishers.

Wittenmark B., J. Nilsson and M. Törngren (1995).l Timing Problems in Real-Time Control Systems: Problem Formulation. *Proc. Of the American Control Conference*, Seattle, Washington.

Wittenmark B., B. Bastian and J. Nilsson (1998). Analysis of Time Delays in Sinchronous and Asynchronous Control Loops. *Proc. Of the 37th Conf. On Decision and Control*, Tampa, FL. US.