European Component Oriented Architecture (ECOA®) Collaboration Programme: Guidance Document: Guidance to develop rhythmic ECOA systems

Date: 30/08/2017

Prepared by
Dassault Aviation

Contents

Figures

0 Executive Summary

This document defines guidance for designing an ECOA system based on only periodic-based execution.

1 Scope

This document is intended to provide guidance on container level checking and time synchronization.

The document is structured as follows:

Section 2 gives a brief introduction to the topic.

Section 3 expands abbreviations used in this report.

Section 4 provides definitions for the key terms used in this report.

Section 5 lists key documents referenced by this report.

Section 6 discusses the guidance to develop rhythmic ECOA systems.

2 Introduction

This document defines guidance for designing an ECOA system based on only periodic-based execution.

Currently ECOA does not allow defining periodic modules. Indeed, an ECOA module is by definition reactive, i.e. a module is executed when it receives an activating operation from another module and when its priority allows it to be scheduled (in case of concurrent access to the computing resource).

However, ECOA defines the notion of trigger, whose purpose is to periodically send one or some ECOA Events to other modules. Only using the trigger does not guarantee a periodic activation, but we will present in this document a way to imitate such activation model, by using this notion of trigger and defining some rules and restrictions.

Note that this document takes place at ECOA level, and does not take into account the technical scheduling level. Then every analysis and methods described here have to be carefully used.

Also note that the purpose is to model a pure periodic system using only the ECOA artefacts. Other methods may exist, such as using external periodic events (e.g. platform events) through driver components to periodically activate the modules.

3 Abbreviations

API	Application Programming Interface
ASC	Application Software Component
DSTL	Defence Science and Technology Laboratory
ECOA	European Component Oriented Architecture
ELI	ECOA Logical Interface
FR	French
IAWG	Industrial Avionics Working Group
I/O	Inputs-Outputs
OS	Operating System
PF	Platform
QoS	Quality of Service
RR	Request-Response
STD	Standard
TR	Technical Report
TRL	Technology Readiness Level
UDP	User Datagram Protocol
UK	United Kingdom
WCET	Worst Case Execution Time
XML	eXtensible Markup Language

4 Definitions

For the purpose of this document, the definitions given in the ECOA Architecture Specification *(ref. [AS])* Part 2 and those given below apply.

Term	***Definition***
(currently none)	

5 References

AS	European Component Oriented Architecture (ECOA) Collaboration Programme: Architecture Specification (Parts 1 to 11) “ECOA” is a registered mark.

6 Guidance to an ECOA pattern to design full periodic systems

6.1 Classical periodic scheduling model

In a classical full periodic scheduling model, each processing unit (typically a thread) is woken up every given period (see Figure 1). Since some threads may wake up at the same time, the computing resource is given to the thread with the highest priority. As for the reactive model, the processing unit has to complete its Worst Case Execution Time before its deadline.

To set such activation model, it is then required, for each processing element, to have the following attributes:

- a period ;
- a priority to handle concurrent access to the computing resource ;
- an estimated WCET ;
- a deadline.

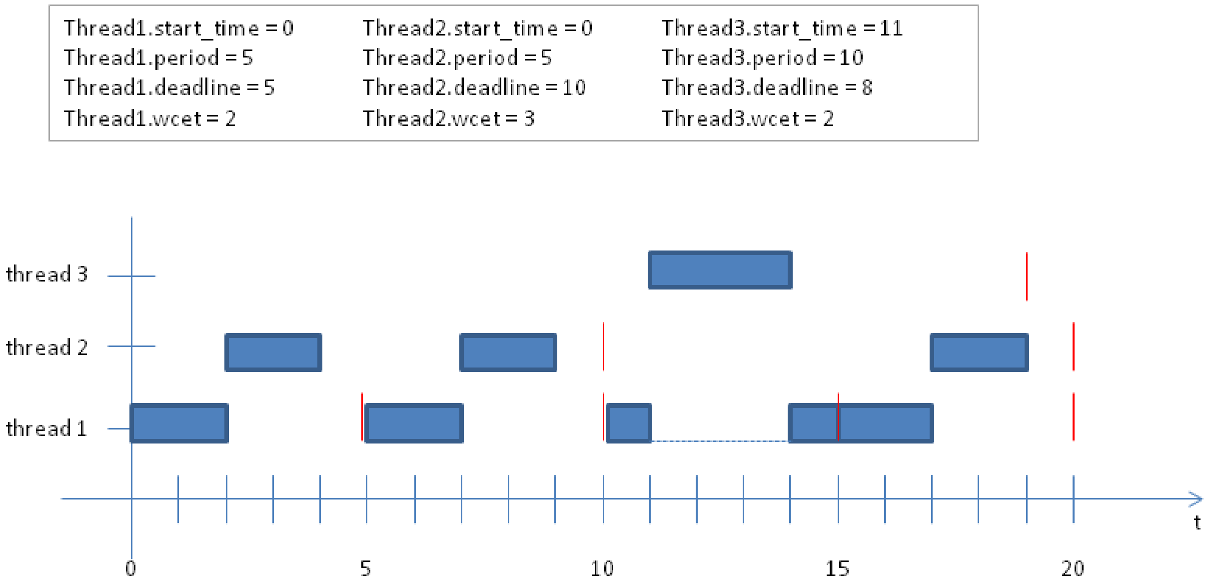

Figure 1: classical periodical scheduling

6.2 ECOA periodic pattern

In ECOA, the processing unit is the application module. By definition in the ECOA standard, the module has only a reactive activation policy, since there is no attribute to set it as periodic or to define an associated period.

The ECOA periodical unit is the trigger. A trigger is periodically woken up by the infrastructure and sends, when it gets the processing resource, an Event to one or several modules it is connected with. Note that if several triggers have to be woken up at the same time, they are executed (i.e. their event are sent) accordingly to their priority, causing a small delay regarding the wakeup time for other triggers that the one with the highest priority (cf. Figure 2).

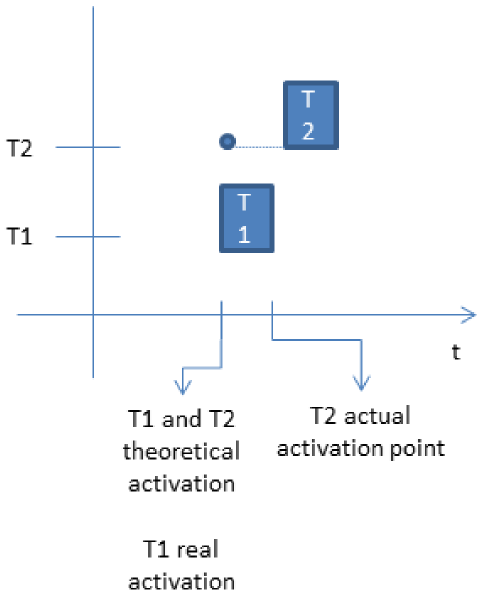

Figure 2: concurrent trigger activation

Consequently, imitating the behaviour of a periodical processing unit of the classical periodic schema required a couple of ECOA elements {module, trigger}, each of them holding a part of the required attributes:

- Module: WCET, priority, deadline ;
- Trigger: priority and period.

The idea of the pattern is then to associate a trigger to each module to periodoically activate this latter, and to define some rules to restrain the use of some ECOA elements and concepts to ensure that the perdiocal schema is followed.

6.3 Guidelines

6.3.1 Module and Trigger Priority

Both module and trigger have a priority that has to be carefully set in order to wake up the module every defined period.

As the triggers are the true periodical unit in charge of activating the corresponding module, their priority has to be higher than their corresponding module priority. More generally, each trigger must have a higher priority than all the modules of its component, or all the modules of the protection domain or the computing node it is deployed on, depending on the chosen scheduling policy (1). Then:

$$\forall\, Ti, \forall\, Mi, Ti.P > Mi.P$$

with Ti any trigger and Mi any module of the [component | *protection domain* | *computing node] (1) and P the priority attribute.*

Moreover, the priority ordering of the modules must be compatible with the priority ordering of the corresponding triggers. For example, for the following couples {M1, T1}, {M2, T2}, {M3, T3},

if T1.P > T2.P > T3.P, then we must have M1.P > M2.P > M3.P

Note (1): the priority defines an order between modules to handle concurrent access to the computing resource. Depending on the scheduling policy chosen, this priority may be related to:

- all the modules of a same component, in case of a hierarchical scheduling component by component ;
- all the modules of a given protection domain, in case of hierarchical scheduling protection domain by protection domain ;
- all the modules of the modules of the computing node in case of RT FIFO scheduling.

Note: using a couple {module, trigger} to imitate a periodical processing unit requires defining a communication for the event sent by the trigger to wake up the module. In some cases, this may lead to spurious behaviour if the communication latency is high, as illustrated on the Figure 3. For example, considering the following couples {M1, T1} and {M2, T2} with:

- T1.P = 1 and M1.P = 10 ;
- T2.P = 2 and M2.P = 12 ;
- T1.period = T2.period ;
- T1.start_time = T2.start_time.

And considering that the event communication latency T1->M1 is higher than T2->M2, then:

1. At the period time, T1 and T2 must be woke up
2. T1.P > T2.P then T1 sends an event E1 to M1
3. Just after, T2 sends an event E2 to M2
4. E2 is received by M2 before E1 arrives to M1
5. M2 starts its execution
6. M1 receives E1 and then pre-empts M2

- T1.P > T2.P
- M1.P > M2.P
- T1.period = T2.period
- T1.start_time = T2.start_time
- communication_latency T1->M1 > communication_latency T2->M2

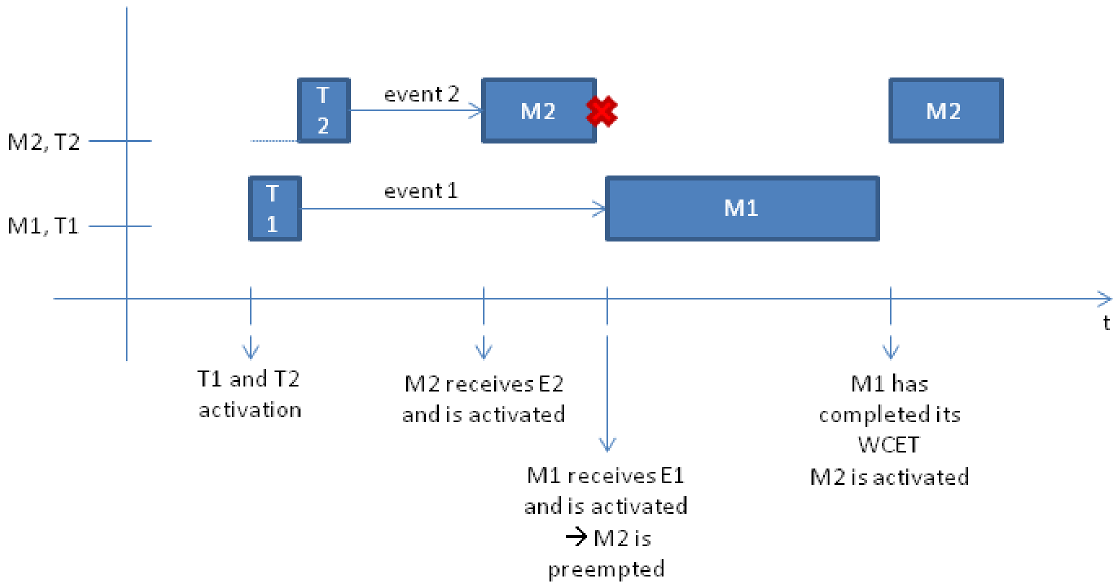

Figure 3: possible spurious behavior

In this case, having a communication between triggers and modules leads to a theoretical spurious behaviour, where M2 starts its execution before M1 even if all has been configured for M1 to start first.

Note: this possible spurious behaviour has been tagged as theoretical, since the analysis based on the ECOA units may not represent the actual scheduling that depends on the way the platform supplier map these ECOA units on the technical processor units (processes and threads). For example, to prevent this kind of behaviour, to reduce the number of threads and to reduce the latencies, the platform supplier may choose to create only one periodic thread from each couple {module, trigger}.

Note: there is a difference with classical scheduling schema on the time available for the module to complete its wcet. Let's define the following values: *CT* for the the computable duration (duration between wakeup time and deadline), *Pre* for the time spent pre-empted by units with highest priority and *C* for the communication between a trigger and its module. As illustrated in the Figure 4, to complete its wcet, in the classical pattern a thread THi has:

$$THi.CT - THi.Pre$$

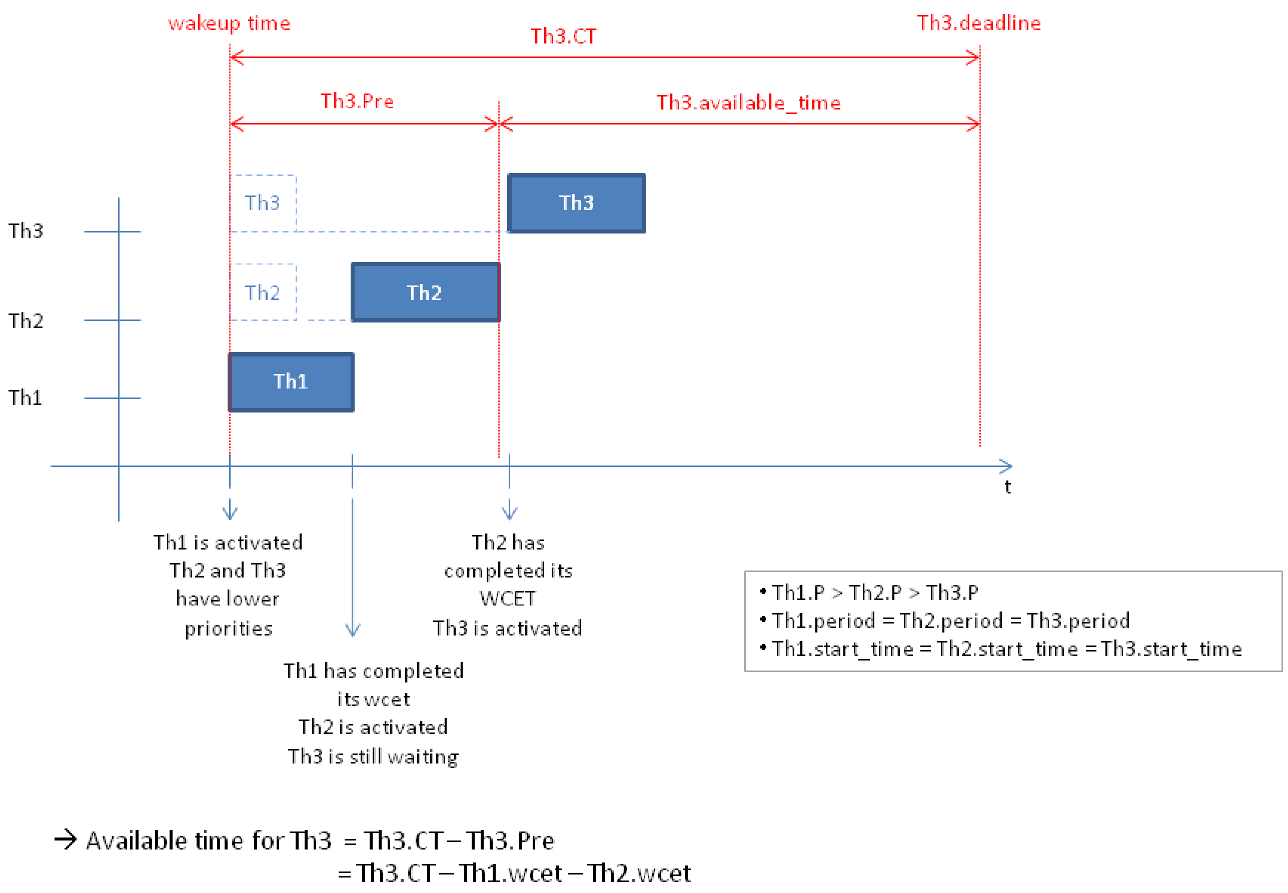

Figure 4: available computing time for a thread in a classical full periodic scheduling

As illustrated in the Figure 5, in the ECOA pattern, assuming all the communications between triggers and modules are equals and assuming all triggers wcet are also equals (as a trigger is not a processing module), a module would have:

$$Mi.CT - Ti.wcet - C(Ti, Mi) - Mi.Pre$$

So a module would have less time available computing time in the ECOA pattern due to the triggers execution and the communications between triggers and modules. In §6.3 we will propose a way to limit that overhead.

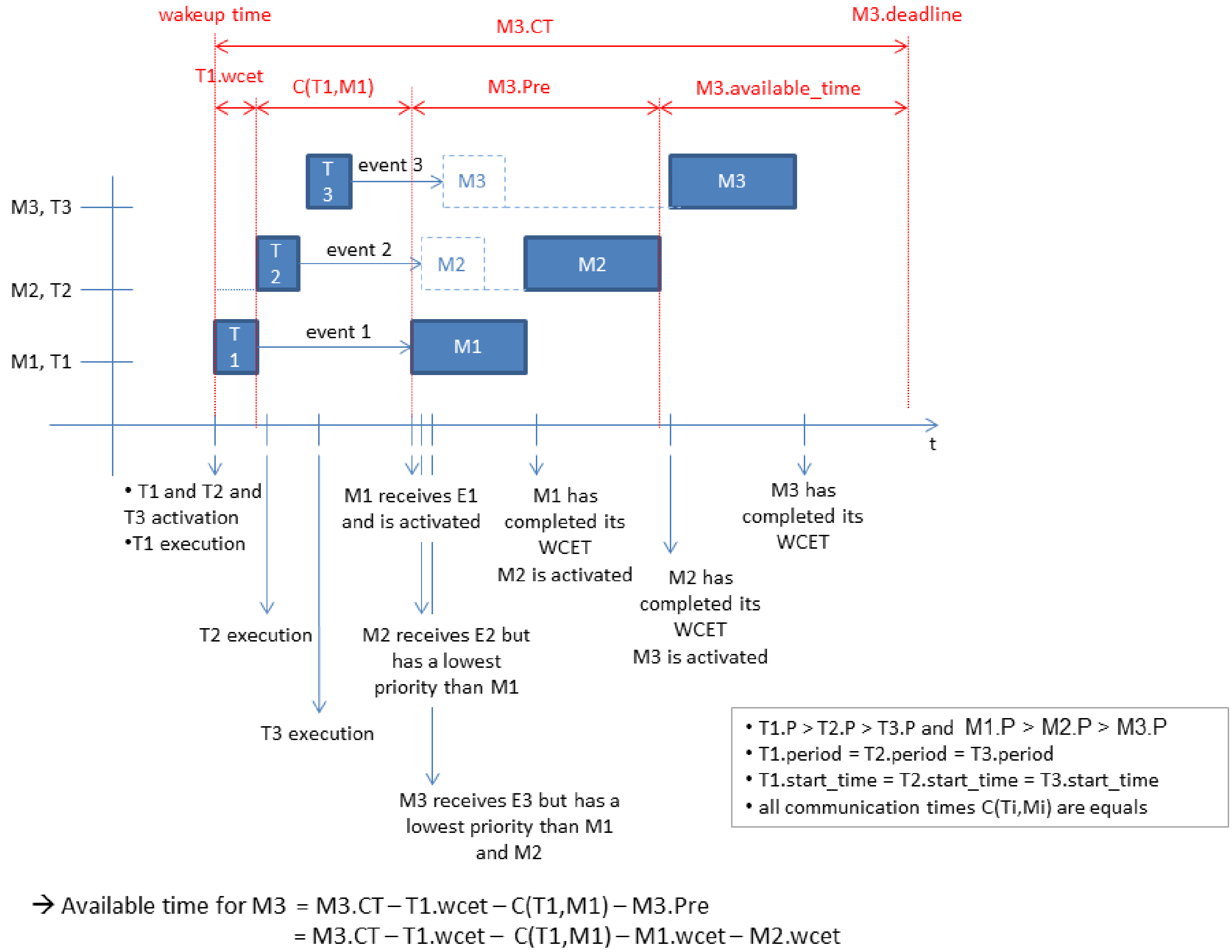

Figure 5: available computing time for a module in the ECOA periodic pattern

6.3.2 Activating and non-activating operations

By now, an assumption has been made on the Event operations sent by the triggers. These operations have been considered as activating.

In order to allow designing a periodic system with ECOA, the Event operations sent by the triggers, that is the only periodic one available in ECOA, must be the only activating operations. All the other operations, tagged as non-activating, are executed when the module gets the processing resource.

In order to avoid a module to indefinitely get the processing resource, only the operations received before the activating one should be executed when the module is scheduled. Operations received during the execution or during the time spent pre-empted are put in the reception FIFO.

6.3.3 Deployment on Protection Domain

Currently, ECOA allows deploying a trigger and its module on two different protection domains, or onto two different computing nodes, which may lead to a situation where the module is activated far after the wakeup time and does not have enough time to complete its execution, as illustrated in the Figure 6.

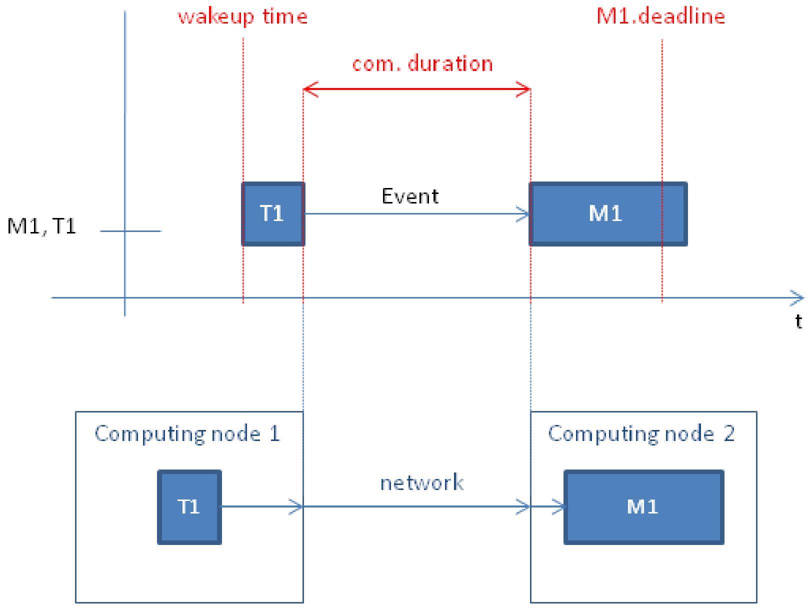

Figure 6: spurious behavior due to communication time

To limit activation overhead and such spurious behavior identified in previous section, the couple {module, trigger} must be deployed onto the same protection domain. Thanks to that rule, the communications between a trigger and its module can be considered as almost immediate, helping solving the identified potential issues.

6.4 Guidance

In order to fulfill the periodic pattern, the dynamic trigger, used to be woken up after a given delay, should not be used since its usage would break the periodic activation pattern (cf. Figure 7).

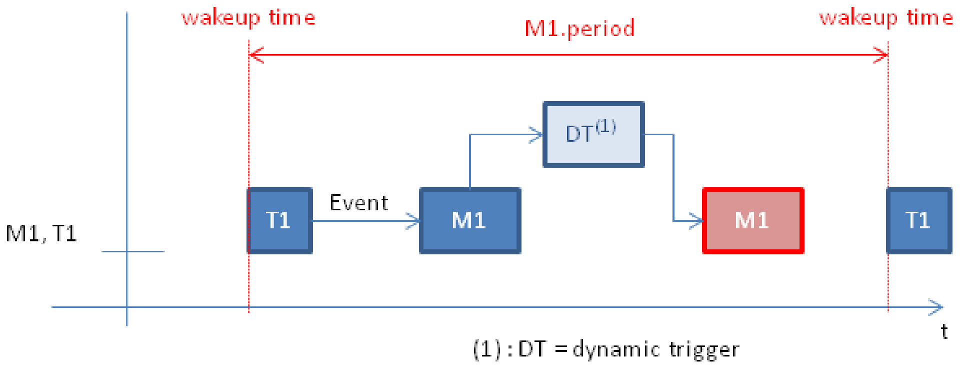

Figure 7: dynamic trigger

6.5 Limitations

The periodic model presented here is a theoretical model applied at ECOA level. Every analysis made on this model should be carefully used, as the actual system execution architecture depends on the way the platform provider map the ECOA artefacts on the system processes and threads.

Again, if the platform supplier chooses to map the couple {trigger, module} on a same periodical thread, then there is not any trigger->module communication time.

Moreover, as in any system (including non-periodic ones), some system processes and threads are executed in the operating system to manage the scheduling, the communications and so on. The overhead related to these system processes and threads can be modeled with the help of the element moduleSwitchTime in the logical system description. This existing modeling of the overhead may be refined in future works (e.g. by modeling the overhead as a function parameterized by the number of modules).

6.6 Summary

In this document we define some guidance to imitate a full periodic system in ECOA. The idea is to base the periodic activation model on a central unit composed of both trigger and its associated module.

These guidances are:

- Triggers and modules priorities have to be set according to the formula detailed in this document ;
- The only activating operation is the Event sent by a trigger to its module. All other operations must be non-activating ;
- The couple {trigger, module} should be deployed on the same protection domain.
- No dynamic trigger.

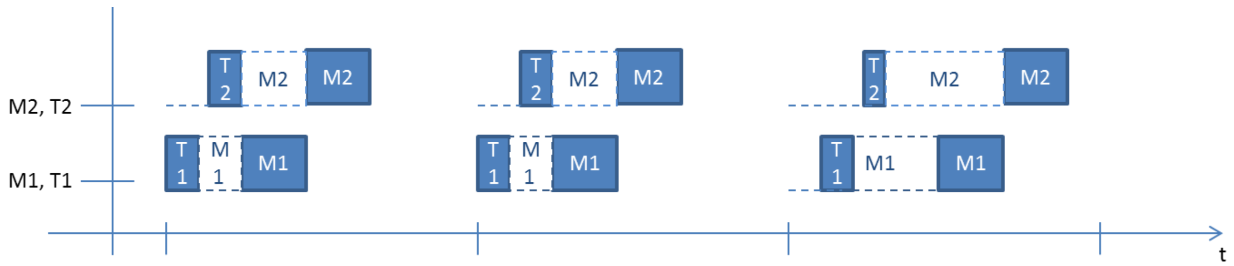

Figure 8: ECOA periodic system with the ECOA pattern

In addition to these aspects, the Technical Insertion Policy Guidance may help to characterize the periodic activation profile of modules implied in a rhythmic system. These profiles may then be used by the System Integrator to define module priorities.