Formal specification of satellite on-board networks requirements

Pierre-Julien Chaine, Marc Boyer, Claire Pagetti, Franck Wartel

Formal specification of satellite on-board networks requirements

Pierre-Julien CHAINE[1], Marc BOYER[2], Claire PAGETTI[2], and Franck WARTEL[1]

1 **Airbus Defence and Space**
Toulouse, France
pierre-julien.chaine@airbus.com, franck.wartel@airbus.com

2 **ONERA**
Toulouse, France
marc.boyer@onera.fr, claire.pagetti@onera.fr

Abstract

This report proposes a formal introduction to the satellite on-board network requirements. These requirements have been sorted into several categories: *Input Contracts* which describe constraints on the injection of the messages of the flows in the system (network), *Generic Constraints* which describe generic constraints as well as medium specific constraints, *Quality of Service Constraints* which describe constraints on the reception of messages of the flows in the system and *Precedence Constraints* which describe temporal relationships between flows. One category of constraints have been left behind : *Fault Tolerance Constraints*, they will not be described in this document.

Keywords and phrases Time Sensitive Networking, Ethernet, Embedded Networks, Requirements, Satellite

1 System Definition

This section will define the system we are considering. These definitions are separated into three parts: *Devices and applications*, *Flows* and *Time Sensitive Networking Network*

1.1 Devices and applications

The system we consider, a satellite, is composed of two parts: *Payload* and *Platform*. The *Payload* is the purpose of the satellite, it is the part of the satellite that generates added value for our clients, it is specific to each and every mission. The *platform* is what allows the satellite to achieve its mission. If this part of the satellite does not work properly, the satellite is useless or might even be lost. The platform is composed of all the systems and subsystems that ensure a nominal behaviour of the satellite. This separation *Payload/Platform* is also applied to the satellite network. The network constraints applied to *Payload* flows are soft real time whereas constraints applied to *Platform* flows are hard real time constraints. Let us now describe the devices and applications of our system.

1.1.1 Devices

The first elements of our system are the devices. We start with a known set of devices. Let us name that set $\mathscr{D}$. The devices ($Dev \in \mathscr{D}$) can be part of four different families:

- *Platform Computing*: The computing devices are where, you name it, the processing of data from sensing devices is made and where commands directed to actuating devices are generated.
- *Sensing*: The sensing devices collect data through analog or digital sensors.
- *Actuating*: The actuating devices execute commands received from a computing device so as to control the attitude and the orbit of the satellite.

- *Payload (Data and Computing)*: the payload devices collect useful data for the mission, for example: antennas and transponders for telecommunication satellites, telescopes and cameras for Earth/Space observation satellites or basically any type of scientific instruments for scientific missions.

In their behaviour, both *Actuating* and *Sensing* devices act as slaves of the *Computing* devices.

Some of the sensing or actuating devices are analog devices and hence are connected to a *RIU* for *Remote Interface Unit* that acts as controller for these devices. This RIU will also be considered as an *Actuating* or a *Sensing* node.

1.1.2 Applications

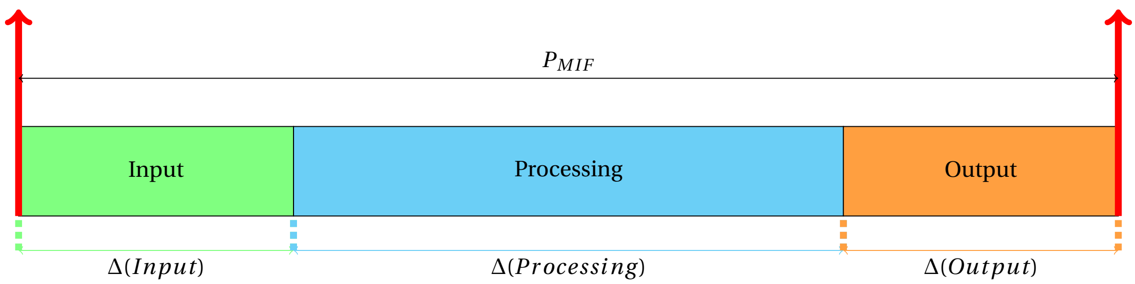

Figure 1 Typical application pattern

Several time constants are used to define the behaviour of the applications (*Command & Control*, *Vison Based Navigation*, *Payload Processing*, etc.) running on the computing devices. In fact the application follows a pattern, named a *cycle*, shown in Fig.1, where a reserved time quantum (δt_1) of its time is dedicated to gathering input data from sensing devices, then another quantum (δt_2) of its time is dedicated to processing this data and one last quantum (δt_3) is dedicated to sending output data to actuating devices according to the output of the processing. Let us name $\Delta(Input)$, the duration of the *input* part, $\Delta(Processing)$, the duration of the *processing* part, $\Delta(Output)$, the duration of the *output* part. We also denote $Start(Part)$ and $End(Part)$, the time instant where the part starts (resp. ends). We can now define the *MIF Period* P_{MIF} such as :

$$\begin{cases} \Delta(Input) + \Delta(Processing) + \Delta(Output) = P_{MIF} \\ \Delta(Input) = \delta t_1 * P_{MIF} \\ \Delta(Processing) = \delta t_2 * P_{MIF} \\ \Delta(Output) = \delta t_3 * P_{MIF} \\ End(Input) = Start(Processing) \\ End(Processing) = Start(Output) \end{cases} \tag{1}$$

▶ **Rule** 1. Starting now, all values for periods in this document will be expressed in milliseconds.

Usual values of P_{MIF} are $125ms$, $62.5ms$ or even $31.25ms$.
The scheme of duration P_{MIF} is repeated k times $\in \mathbb{N}$ leading to P_{MAF} so that:

$$P_{MAF} = k * P_{MIF} \tag{2}$$

▶ **Hypothesis** 1. In our system, $P_{MAF} = 1000$ hence $k = \frac{1000}{P_{MIF}}$.

This hypothesis is illustrated in Fig. 2. Tab. 1 gives the value of k for the usual values of P_{MIF}.

k	P_{MIF}
8	$125ms$
16	$62.5ms$
32	$31.25ms$

Table 1 Usual values of k and P_{MIF}

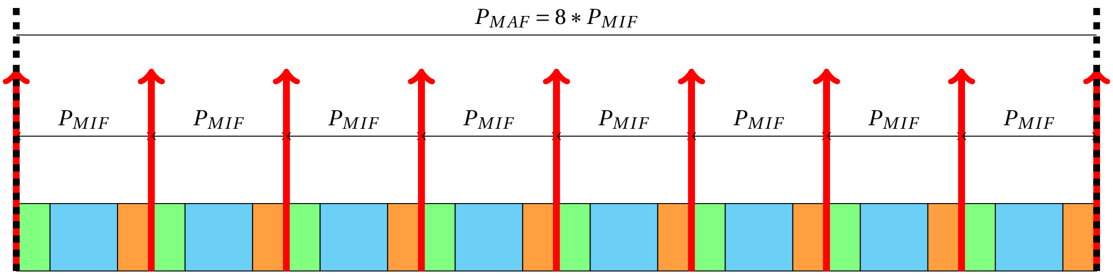

Figure 2 Link P_{MIF} - P_{MAF} with $k = 8$

1.1.3 Motivating Example

All along this report, we will use a motivating example to illustrate the given definitions, constraints, etc. Every now and then, we will use this example or add to it so as to ease the understanding of the concepts of this document. This example, for now, is composed of three devices:

- One Computing Device that we will name *OBC*.
- One Sensing Device that we will call *High Performance Sensor*
- One Actuating Device that we will call *High Performance Actuator*, that is, in reality, connected through a *RIU - Remote Interface Unit*

The connexions between devices are illustrated in Fig.3. Two communication links are illustrated in the figure. Communication links will be introduced later in this technical note.

1.2 Flows

The applications running on the computing devices communicate with the sensing and actuating devices with messages. These messages are gathered under the concept of flows.

▶ **Definition** 1. Flow
A flow is a unidirectional sequence of messages from one sender to one or several receivers. Let us denote f a flow, and $\mathbb{F}$ the set of flows of our system. A flow is characterized with the following tuple:

$$\forall f \in \mathbb{F} < f.Src_f, f.Dest_f, f.MaxDataSize, f.r_f > \quad (3)$$

Where :

- $f.Src_f$ is the device from which the messages are generated and emitted
- $f.Dest_f$ are the device(s) to whom the messages are sent.
- f.MaxDataSize is the maximum data size the data sent by the flow in one applicative message. We express it in bytes.

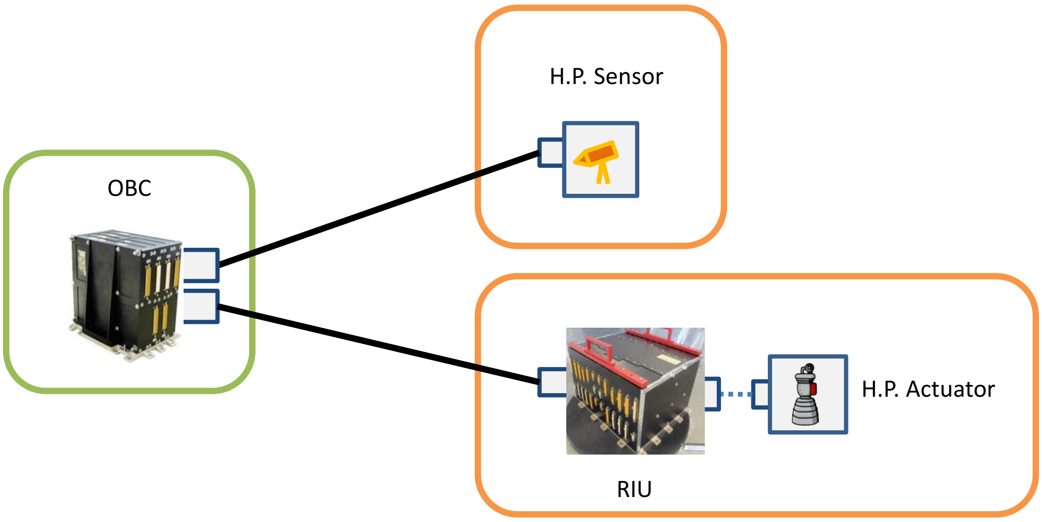

Figure 3 Motivating Example with three devices

- $f.r_f$ is our way to express the periodicity of the flow (see Eqn. 5), $r_f \in \mathbb{Z}$.

▶ **Hypothesis** 2. In our system, we will use an Ethernet medium and we will consider that :

$$\forall f \in \mathbb{F}, f.MaxDataSize \leq MTU_{Ethernet} \tag{4}$$

This means, that one applicative message of a flow will lead to one message in the network. The definitions, properties and constraints in the rest of the document apply this hypothesis. If this hypothesis is later relaxed, the definitions, properties and constraints of the document will have to be slightly redefined.

The period P_f of flow f, is linked with its ratio r_f with the following equation:

$$\forall f \in \mathbb{F}, \begin{cases} r_f < -1 & \implies P_f = |r_f| * P_{MIF} \\ r_f > 0 & \implies P_f \iff r_f \ messages \ per \ P_{MIF} \\ r_f = 0 & \implies P_f = NA \ (f \text{ is non-periodic}) \end{cases} \tag{5}$$

Figuratively, if $r_f > 0$, flow f sends r_f messages during one P_{MIF}, hence $k * r_f$ messages during P_{MAF}. If $r_f < -1$, flow f sends one message during $|r_f| * P_{MIF}$, hence $\frac{k}{|r_f|}$ messages during P_{MAF}. The concept of ratio is illustrated in Fig. 4 with, in blue a flow with $r_f = -4$ and in green a flow with $r_f = 2$.

▶ **Property** 1. Link r_f - k

$$\begin{aligned} & r_f < -1 \implies r_f \mid k \\ & \text{hence } r_f < -1 \implies lcm(|r_f|, k) = k \end{aligned} \tag{6}$$

▶ **Hypothesis** 3. In our system we only consider periodic flows and a flow shall at least send one message every P_{MAF} i.e. :

$$r_f \in [-k, -1] \cup [1, +\infty[\tag{7}$$

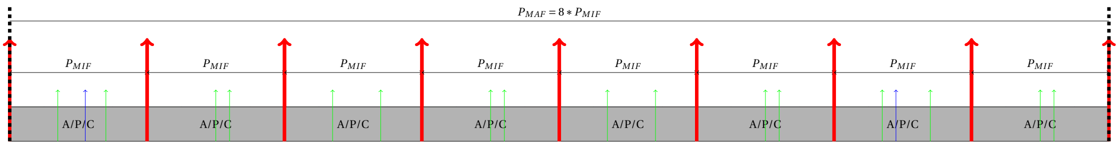

Figure 4 Ratio concept with two flows ($r_{f1} = 2$, $r_{f2} = -4$) and $k = 8$

1.2.1 Motivating Example: Adding Flows

In our motivating example (cf. 1.1.3), the applications running on the devices communicate with three flows. Starting now, the convention used for naming flows is the following:

▶ **Rule** 2. Naming Flows
A flow originating from Src_f and going to $Dest_f$ will be named:

$$f\_Src_f\_Dest_f\_ID \tag{8}$$

Where *ID* is a user defined identifier used to distinguish several flows with the same senders and receivers.

According to R. 2, the flows, represented in Fig. 5, are the following:

- *f_OBC_HPActuator_1*
- *f_OBC_HPSensor_1*
- *f_HPSensor_OBC_1*

For each flow, we give below the tuple that characterizes them:

- *f_OBC_HPActuator_1* : < *OBC*, *HPActuator*, 1500, 1 >,
- *f_OBC_HPSensor_1* : < *OBC*, *HPSensor*, 1500, 1 >,
- *f_HPSensor_OBC_1* : < *HPSensor*, *OBC*, 1500, 1 >,

The following subsections will introduce the different contracts identified for the flows, grouped in four categories: *Input Contracts*, *Generic Constraints*, *QoS Constraints*, *Precedence Constraints.*. This constraints will often be called *flow constraints*. For every constraint, an example will illustrate the constraint being satisfied. In these figures, the colors representing the Input/Processing/Output parts have been removed for the sake of readability of the flow constraints.

1.2.2 Input Contract

In this paragraph, we will introduce the *Input contracts* identified for our system. Three *Input contracts* are identified for our system: the *Periodic Message Production* constraint, the *Injection Zone* constraint and the *Controlled Input Traffic Shape* constraint.

▶ **Rule** 3. Emission and Reception dates
In this document, all the Emission and Reception dates are at applicative level.

▶ **Rule** 4. Emission and Reception dates in figures
Starting now, in this document, we will materialize:

- The emission date of a message by an upward facing arrow
- The reception date of a message by a downward facing arrow

This rule will help readability of figures in particular if several messages are present in a single cycle.

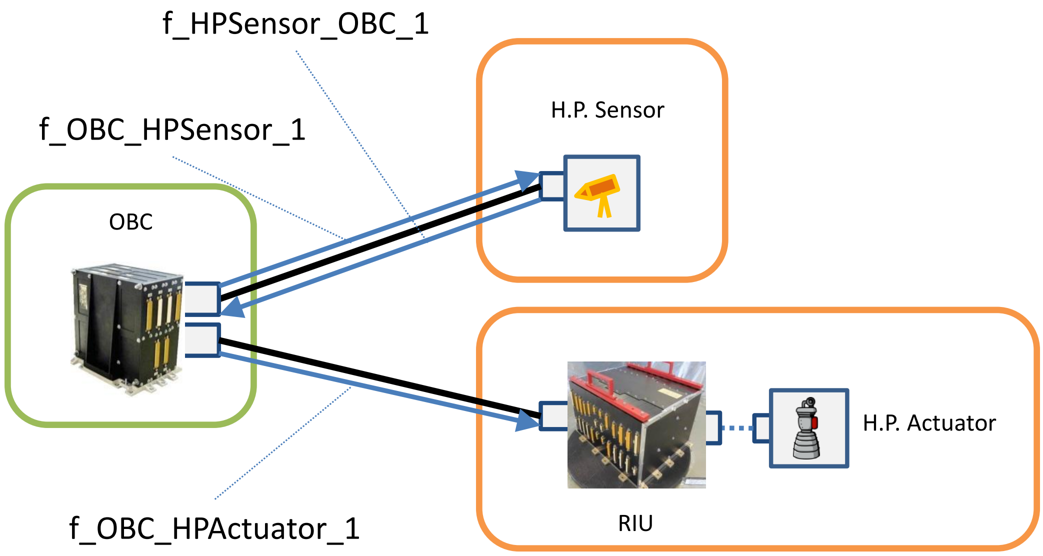

Figure 5 Adding three flows to the motivating example

▶ **Constraint Template** 1. Periodic message production
Let $f \in \mathbb{F}$ be a flow such that $r_f \neq 0$. Let f_l be the lth message of flow f (from device Src_f). Let us denote $\forall l \in \mathbb{N}, T_p(f_l)$ the emission date of f_l.

$$f \models PeriodicMessageProduction$$
$$\stackrel{def}{\equiv}$$
$$\forall l \in \mathbb{N}, \quad \begin{cases} -k \leq r_f < -1 & \Longrightarrow l * P_f \leq T_p(f_l) < (l+1) * P_f \\ r_f > 0 & \Longrightarrow \lfloor \frac{l}{r_f} \rfloor * P_{MIF} \leq T_p(f_l) < (1 + \lfloor \frac{l}{r_f} \rfloor) * P_{MIF} \end{cases} \tag{9}$$

This first constraint defines a window for every message of a periodic flow (i.e. $r_f \neq 0$ in which the message of the flow shall be emitted. It is illustrated in Fig. 6. In blue, we have represented a flow with $r = -4$, and in green, a flow with $r = 2$.
We define *Refcycle*(f_l), a reference date for f_l such as:

▶ **Definition** 2. Reference Date

$$\forall f \in \mathbb{F} \ with \ r_l \neq 0, \forall l \in \mathbb{N}, \quad \begin{cases} -k \leq r_f < -1 & \Longrightarrow Refcycle(f_l) = l * P_f \\ r_f > 0 & \Longrightarrow Refcycle(f_l) = \lceil \frac{l}{r_f} \rceil * P_{MIF} \end{cases} \tag{10}$$

The concept of *reference date* is also illustrated in Fig. 6 by the blue and green brackets.

In our system, one may want to reduce even more the emission window of the flows. That is why a second constraint is introduced: the *Injection Zone* constraint.

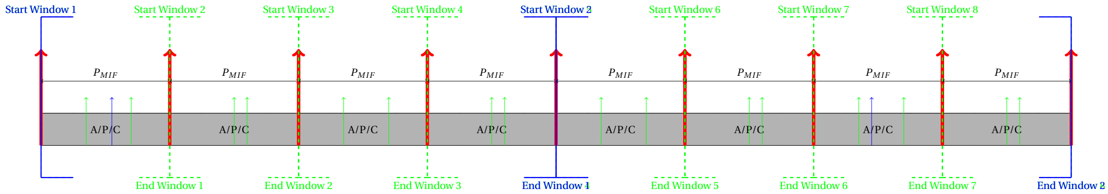

Figure 6 Periodic Message Production Constraints, with 2 flows ($r_{f1} = 2$, $r_{f2} = -4$) and $k = 8$

▶ **Constraint Template** 2. Injection Zone($EarliestProdOffset$, $LatestProdOffset$)
Let $f \in \mathbb{F}$ be a flow with $r_f \neq 0$. Let f_l be the lth message of flow f (from device Src_f). Let us denote $\forall l \in \mathbb{N}, T_p(f_l)$ the emission date of f_l. Flow f satisfies Injection Zone constraints if all messages of the flows are emitted between their earliest and latest possible emission dates i.e.:

$$f \models InjectionZone(EarliestProdOffset, LatestProdOffset) \overset{def}{\equiv} \forall l \in \mathbb{N}, \mathbf{Prod_{early}(f_l)} \leq T_p(f_l) \leq \mathbf{Prod_{late}(f_l)} \quad Where: \begin{cases} EarliestProdOffset \geq 0 \\ EarliestProdOffset \leq LatestProdOffset < max(P_{MIF}, Refcycle(f_{l+1}) - Refcycle(f_l)) \\ \mathbf{Prod_{early}(f_l)} = Refcycle(f_l) + EarliestProdOffset \\ \mathbf{Prod_{late}(f_l)} = Refcycle(f_l) + LatestProdOffset \end{cases} \tag{11}$$

▶ **Property** 2. In all cases:

$$Refcycle(f_l) \leq \mathbf{Prod_{early}(f_l)} \leq T_p(f_l) \leq \mathbf{Prod_{late}(f_l)} < max(Refcycle(f_l) + 1, Refcycle(f_{l+1})) \tag{12}$$

This constraint is illustrated in Fig. 7.

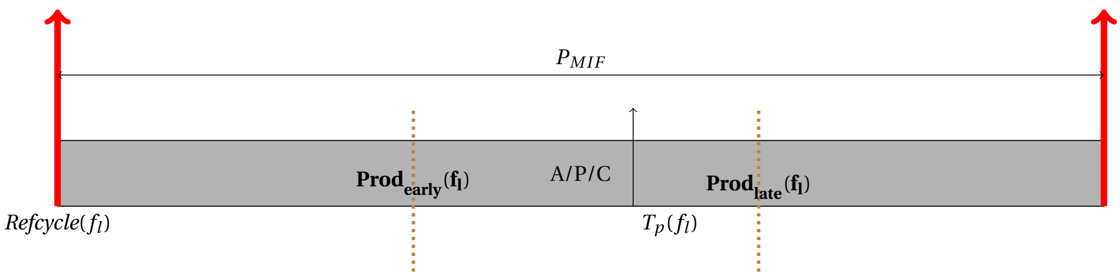

Figure 7 Injection Zone constraint with $EarliestProdOffset = \frac{P_{MIF}}{3}$ and $LatestProdOffset = \frac{2*P_{MIF}}{3}$

We raise the readers' awareness on the fact that in Fig. 7, $EarliestProdOffset$ and $LatestProdOffset$ have been represented in the same P_{MIF} cycle. However, as Constraint 2 states, $EarliestProdOffset$ and $LatestProdOffset$ can be in separate cycles (if $LatestProdOffset >= P_{MIF}$).

One constraint shall also be mentioned: in our system we shall ensure an ordered emission of the messages within a flow. This translates in the *Ordered Emission* constraint.

▶ **Constraint Template** 3. Ordered Emission
Let $f \in \mathbb{F}$ be a flow. Let f_l, f_m two messages of f.

$$\begin{aligned} & f \models OrderedEmission \\ & \overset{def}{\equiv} \\ & \forall l, m \in \mathbb{N},\ l < m, T_p(f_l) < T_p(f_m) \end{aligned} \tag{13}$$

The last constraint that we have identified is motivated by applications running on payload nodes.

▶ **Constraint Template** 4. Controlled Input Traffic Shape
Let $f \in \mathbb{F}$ be a flow. Let R be a rate, in Mbits/s. Let B be a burst, in bytes. Flow f satisfies the Controlled Input Traffic Shape if and only if f input traffic is within a token bucket of parameters (R,B) i.e.:

$$\begin{aligned} & f \models ControlledInputTrafficShape(R, B) \\ & \overset{def}{\equiv} \\ & \forall l, m \in \mathbb{N}, \sum_{i=l}^{m} size(f_i) \leq R * (T_p(f_m) - T_p(f_l)) + B \end{aligned} \tag{14}$$

1.2.3 Generic Constraints

In this paragraph, we will introduce the *Generic Constraints* for our system. Two constraints are identified, for now, for our system: the *Reception after Emission* constraint and the *Ethernet Frame Size* constraint.

▶ **Constraint Template** 5. Reception after Emission
Let $f (\in \mathbb{F})$ be a flow. Let $T_p(f_l)$, $T_r(f_l)$ be the production (resp. reception) date of the lth message of f (i.e. f_l). f respects the *Reception after Emission* constraints, if and only if, for all its messages, the reception date is greater than the emission date i.e.:

$$\begin{aligned} & \forall f \in \mathbb{F}, \\ & f \models ReceptionAfterEmission \\ & \overset{def}{\equiv} \\ & \forall l \in \mathbb{N}, T_p(f_l) < T_r(f_l); \end{aligned} \tag{15}$$

This constraint is naturally extended to multicast flows as follows:

$$\begin{aligned} & \forall f \in \mathbb{F}, \\ & f \models ReceptionAfterEmission \\ & \overset{def}{\equiv} \\ & \forall d \in Dest_f, \forall l \in \mathbb{N}, T_p(f_l) < T_r(f_l, d) \end{aligned} \tag{16}$$

In Fig. 8, we have represented $T_p(f_l)$ and $T_r(f_l)$ respecting the previous constraint.

▶ **Constraint Template** 6. Ethernet Frame Size
Let $f (\in \mathbb{F})$ be a flow. Let f_f be the lth message of f. Let *Frame(f_l)* be the Ethernet frame corresponding to the lth message of f. f respects the *Ethernet Frame Size* constraint, if and only if, for all its messages, the size of their associated frames is lower or equal to the Ethernet MTU i.e.:

$$\begin{aligned} & \forall f \in \mathbb{F}, \\ & f \models EthernetFrameSize \\ & \overset{def}{\equiv} \\ & \forall f_l, size(Frame(f_l)) < Ethernet_{MTU} \end{aligned} \tag{17}$$

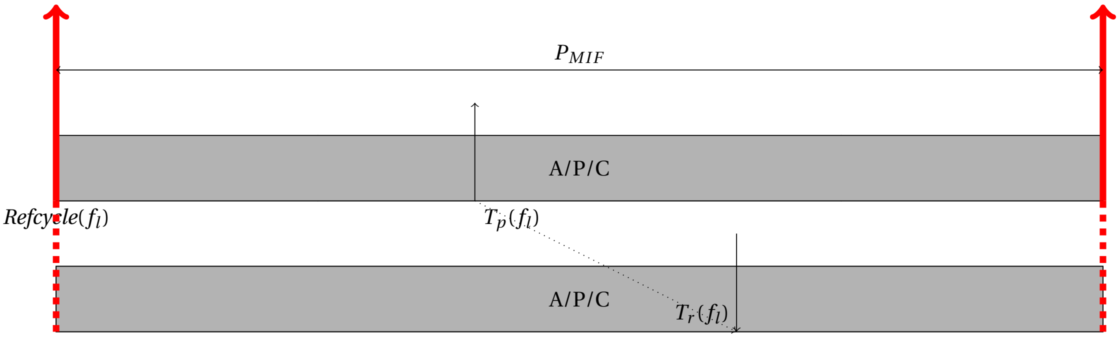

Figure 8 Reception After Emission constraints on f_l

1.2.4 Quality of Service Constraints

Several *Quality of Service constraints* are identified for our system. One flow can be subject to any combination of this *Quality of Service constraints.*

As well as the order of emission of the messages of the flows was constrained, the reception of said messages is also constrained. This translates into *Ordered Delivery constraint.*

▶ **Constraint Template** 7. Ordered Delivery
Let $f \in \mathbb{F}$ be a flow. Let f_l, f_m two messages of f.

$$\begin{aligned} & f \models OrderedDelivery \\ & \stackrel{def}{\equiv} \\ & \forall l, m \in \mathbb{N},\ l < m, T_r(f_l) < T_r(f_m) \end{aligned} \tag{18}$$

▶ **Constraint Template** 8. Implicit Deadlines
Let $f \in \mathbb{F}$ be a flow such that $r_f \neq 0$.

$$\begin{gathered} f \models ImplicitDeadlines \\ \stackrel{def}{\equiv} \\ \forall l \in \mathbb{N},\ \begin{cases} -k \leq r_f < -1 & \implies l * P_f \leq T_r(f_l) < (l+1) * P_f \\ r_f > 0 & \implies \lfloor \frac{l}{r_f} \rfloor * P_{MIF} \leq T_r(f_l) < (1 + \lfloor \frac{l}{r_f} \rfloor) * P_{MIF} \end{cases} \end{gathered} \tag{19}$$

This constraints implies that the latest reception date of f_l of flow f must be received before the beginning of the emission window of f_{l+1}. It is illustrated in Fig. 9.

Again, in our system, one may want to reduce even more the reception window for the messages of a flow. This translates into *TimeZone constraints.*

▶ **Constraint Template** 9. TimeZone($EarliestRecOffset$,$LatestRecOffset$)
Let $f \in \mathbb{F}$ be a flow such that $r_f \neq 0$. Let f_l be the lth message of f. Let $T_r(f_l, d)$ be the reception date of

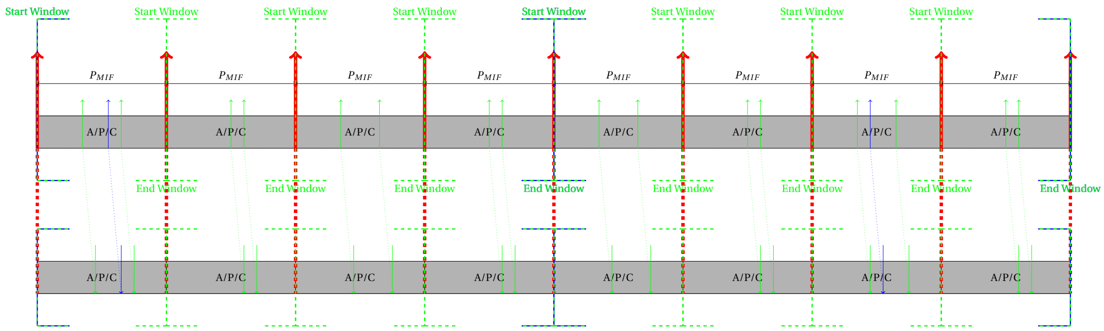

Figure 9 ImplicitDeadlines constraint, with 2 flows ($r_{f1} = 2$, $r_{f2} = -4$) and $k = 8$

f_l at destination d.

$$\begin{gathered} f \models TimeZone(EarliestRecOffset, LatestRecOffset) \\ \stackrel{def}{\equiv} \\ \forall d \in Dest_f, \forall l \in \mathbf{N}, \mathbf{Rec_{early}(f_l)} \leq T_r(f_l, d) \leq \mathbf{Rec_{late}(f_l)} \\ Where: \\ \begin{cases} EarliestRecOffset \geq 0 \\ EarliestRecOffset \leq LatestRecOffset < max(P_{MIF}, Refcycle(f_{l+1}) - Refcycle(f_l)) \\ \mathbf{Rec_{early}(f_l)} = Refcycle(f_l) + EarliestRecOffset \\ \mathbf{Rec_{late}(f_l)} = Refcycle(f_l) + LatestRecOffset \end{cases} \end{gathered} \tag{20}$$

Remark: In all cases:

$$Refcycle(f_l) \leq \mathbf{Rec_{early}(f_l)} \leq T_r(f_l, d) \leq \mathbf{Rec_{late}(f_l)} < max(Refcycle(f_l) + 1, Refcycle(f_{l+1})) \tag{21}$$

This constraint is illustrated in Fig. 10.

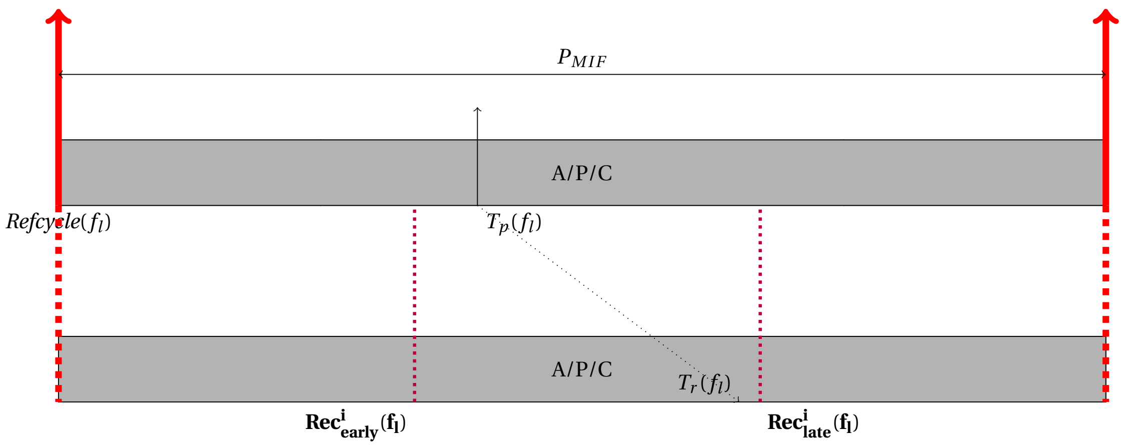

Figure 10 Timezone constraint with $EarliestRecOffset = \frac{P_{MIF}}{3}$ and $LatestRecOffset = \frac{2*P_{MIF}}{3}$

Now that the emission and reception date of the lth message of flow f have been defined (respectively $T_p(f_l)$ and $T_r(f_l, d)$), the concept of latency can be defined. Usually, the network end-to-end latency of the lth message of flow f would be defined as the difference between the reception date and the emission date of f_l. However, we define the latency of f_l as follows:

▶ **Definition 3.** Latency

$$\forall d \in Dest_f, \forall l \in \mathbb{N}, Lat_{f_l,d} = |T_r(f_l,d) - Refcycle(f_l)| \tag{22}$$

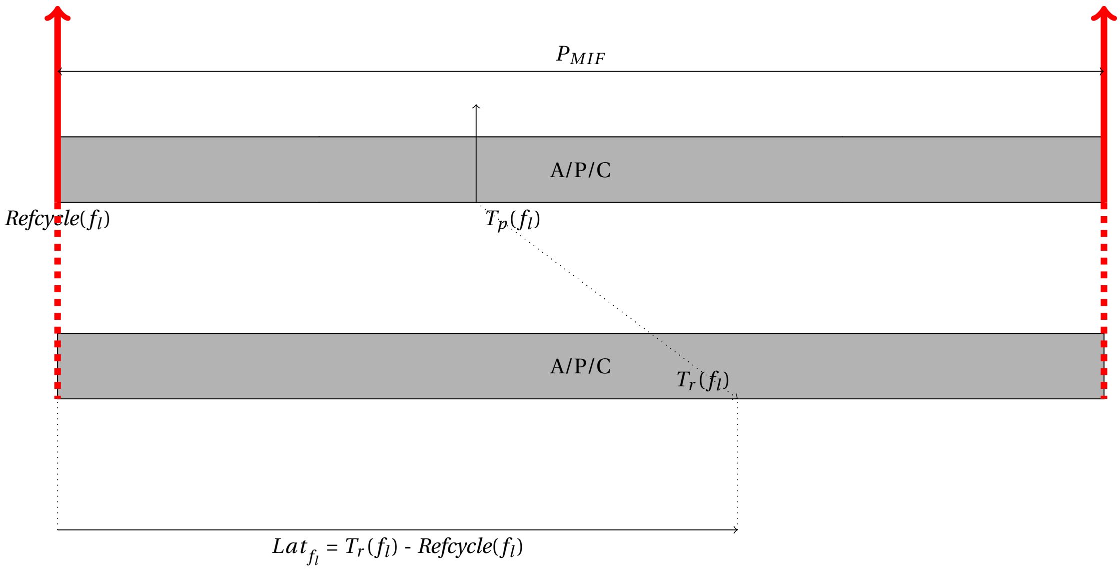

Figure 11 Latency concept in our system

This definition is illustrated in Fig. 11. With this new definition, instead of considering the time spend in the network by f_l, we consider the age of f_l compared to its reference date *Refcycle*(f_l). The *Jitter constraint* can now be defined.

▶ **Constraint Template 10.** Jitter(α)
Let $f \in \mathbb{F}$ be a flow, let f_l, f_m be two messages of f. Let α be a positive constant.

$$f \models Jitter(\alpha) \stackrel{def}{\equiv} \begin{cases} -k \le r_f < -1 & \implies \forall l, m \in \mathbb{N}, |Lat_{f_l} - Lat_{f_m}| \le \alpha \\ r_f > 0 & \implies \forall l, m \in \mathbb{N}, l \equiv m \ (mod\ r_f), |Lat_{f_l} - Lat_{f_m}| \le \alpha \end{cases} \tag{23}$$

Hence, in our system, jitter is understood (like in classic networking system) as the variability of the latency of the frames of the flow. However, the definition of latency is different.
This constraint is illustrated in Fig. 12.

As flows may have more than one destination (gathered in $Dest_f$), we added one constraint to add a relation between the reception dates on the different devices in $Dest_f$. This constraint is the *Sync constraint*.

▶ **Constraint Template 11.** Sync(ϵ)
Let $f \in \mathbb{F}$ be a flow, let f_l be the lth message of f, let $Dest_f$ be the list of recipients of f.

$$\begin{gathered} f \models Sync(\epsilon) \\ \stackrel{def}{\equiv} \\ \forall l \in \mathbb{N}, \forall x, y \in Dest_f, |T_r(f_l,x) = T_r(f_l,y)| < \epsilon \end{gathered} \tag{24}$$

This constraint is illustrated in Fig. 13. The two dotted red bars represented the acceptance window ϵ.

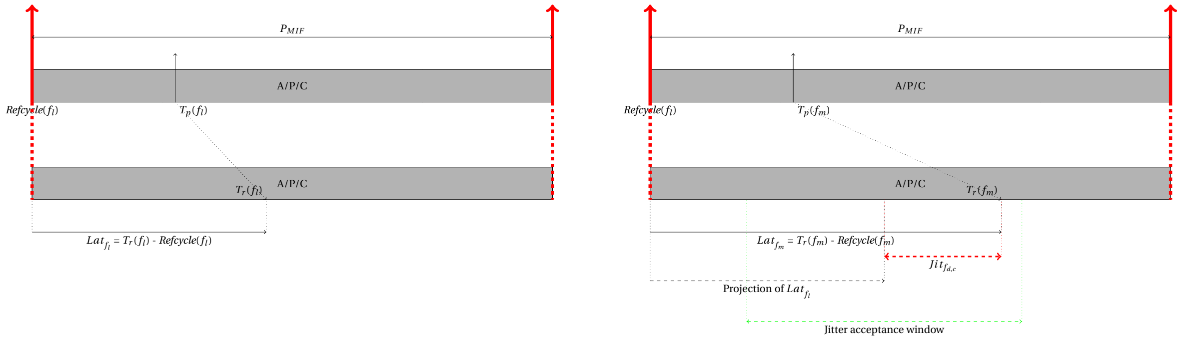

Figure 12 Jitter visualisation with two messages f_l and f_m

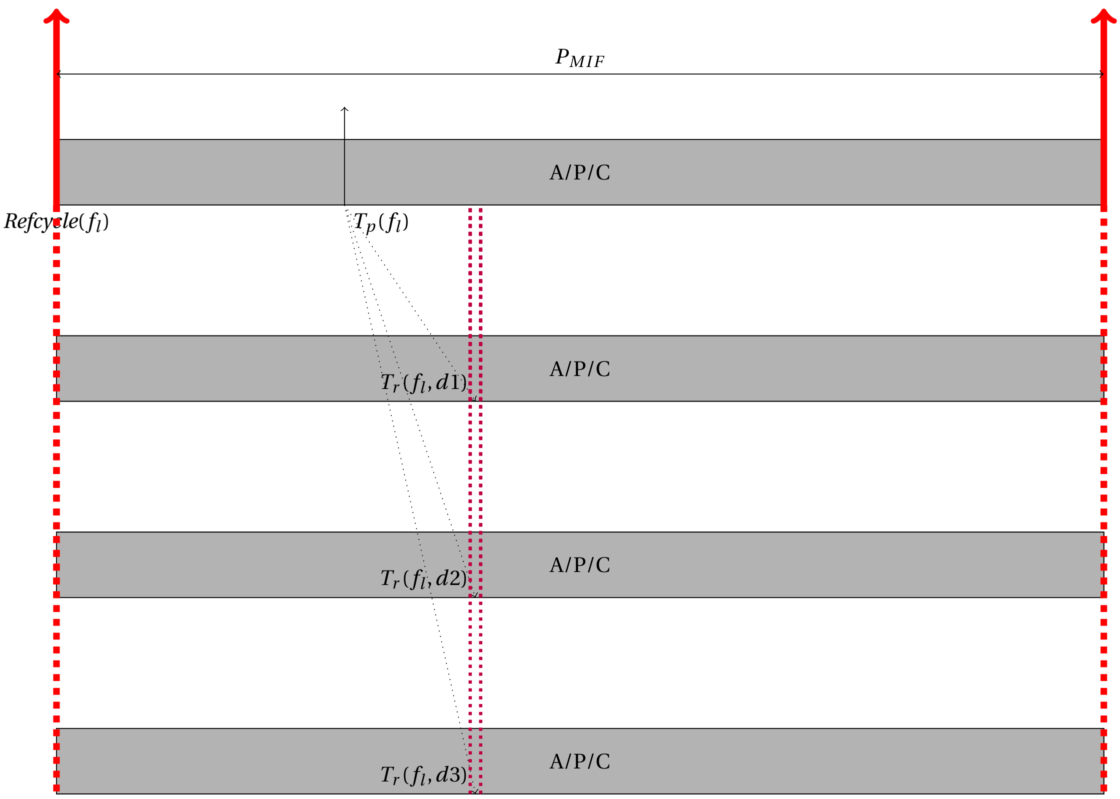

Figure 13 Sync constraint with 3 recipients for flow f

The *Quality of Service constraints* identified up to now in this document describe constraints that would be qualified of *real-time* constraints. The last constraint of this paragraph introduces a relation between two consecutive messages of a flow.

▶ **Constraint Template** 12. Minimum Space(γ)
Let $f \in \mathbb{F}$, be a flow, let f_l, f_{l+1} be two consecutive messages of f. Let γ be a positive constant.

$$f \models MinimumSpace(\gamma) \overset{def}{\equiv} \forall l \in \mathbb{N}, T_r(f_{l+1}) - T_r(f_l) \geq \gamma \tag{25}$$

This constraint is illustrated in Fig. 14.

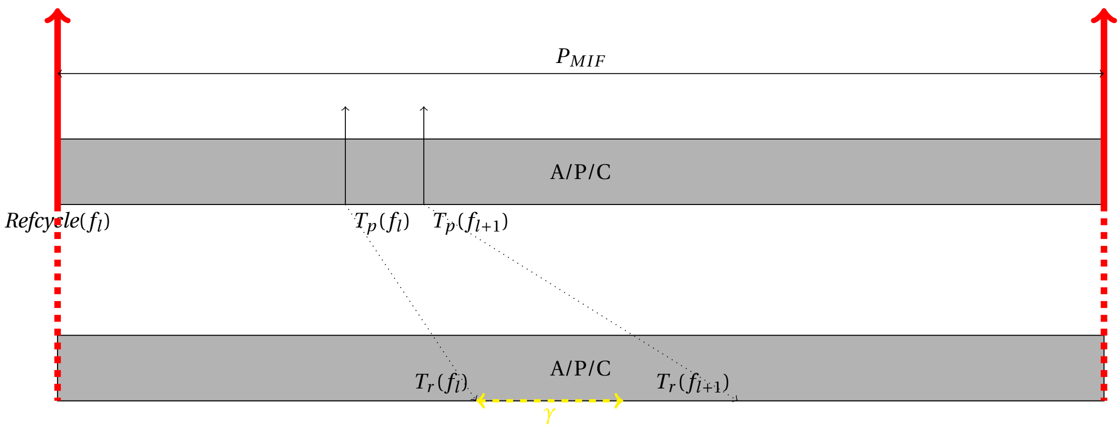

Figure 14 Constraint Minimum Space(γ) for flow f

This constraint allows to define a minimum space between the reception of two consecutive messages of a flow regardless of their emission date.

▶ **Constraint Template** 13. [*Option 1*] Bounded Arrival For Processing (α_d)
Let $f \in \mathbb{F}$ be a flow. Let α_d be the expected number of messages of flow f to be received during duration d. Let us assume that when the lth message of f, f_l, is received, it is immediately processed during duration $\delta_{process}(f_l)$. Flow f satisfies the Bounded Arrival For Processing constraint if and only if the last message of the set of received messages is received early enough to be processed before the end of the cycle i.e.:

$$f \models BoundedArrivalForProcessing(\alpha_{P_{MAF}}) \; \forall l \in \mathbb{N}, \; T_r(f_{(l+1)*\alpha_{P_{MAF}}-1}) <= Refcycle(f_{l*\alpha_{P_{MAF}}}) + P_{MAF} - \delta_{process}(l * \alpha_{P_{MAF}}) \tag{26}$$

Remark: In order for this constraint to hold, the network designer shall ensure that the reception of the l+1th message is after the reception of the lth message plus its processing time i.e.:

$$\forall l, T_r(f_{l+1}) <= T_r(f_l) + \delta_{process}(f_l) \tag{27}$$

▶ **Constraint Template** 14. [*Option 2*] Bounded Arrival For Processing (α_d)
Let $f \in \mathbb{F}$ be a flow. Let α_d be the expected number of messages of flow f to be received during duration d. Let us assume that when all the α_d messages are received, their processing starts. We denote $\delta_{process}(f_l)$,

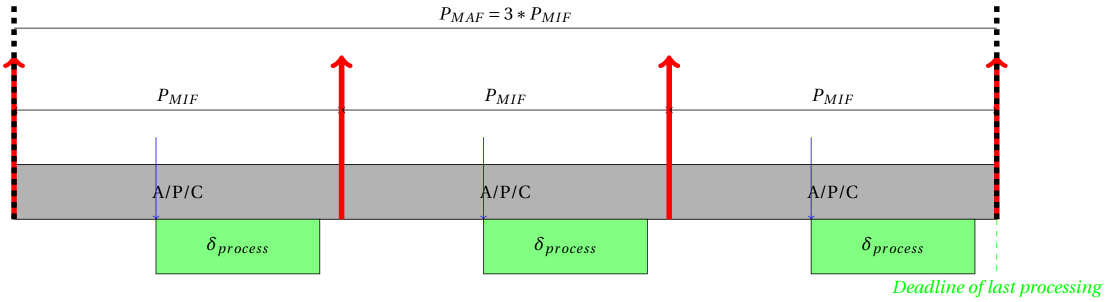

Figure 15 Bounded Arrival For Processing with $\alpha_{P_{MAF}} = 3$ - Option 1

the required processing time for the lth message of f. Flow f satisfies the Bounded Arrival For Processing constraint if and only if the last message of the set of received messages is received early enough so that all the messages are processed before the end of the cycle i.e.:

$$\begin{gathered} f \models BoundedArrivalForProcessing(\alpha_{P_{MAF}}) \\ \forall l \in \mathbb{N}, \\ T_r(f_{(l+1)*\alpha_{P_{MAF}}-1}) <= Refcycle(f_{l*\alpha_{P_{MAF}}}) + P_{MAF} - \sum_{i=0}^{\alpha_{P_{MAF}}-1} \delta_{process}(f_{l*i}) \end{gathered} \tag{28}$$

Remark: In order for this constraint to hold, the network designer shall ensure that the necessary time to receive and process all α_d frames is lower than duration d.

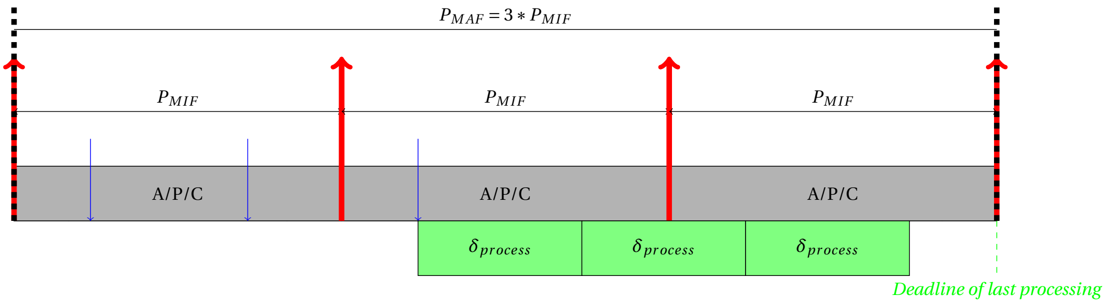

Figure 16 Bounded Arrival For Processing with $d = P_{MAF}$ and $N_{Frames} = 3$ - Option 2

1.2.5 Precedence Constraints

This last paragraph will introduce constraints that link flows together. We have identified one constraint in this category: the *Reaction constraint*.

▶ **Constraint Template** 15. Reaction(ω, f)
Let $f, g \in \mathbb{F}$ be two flows, let f_l (resp. q_l) the lth message of f (resp. g). Let ω a positive constant.

$$\begin{gathered} f \models Reaction(\omega, f) \\ \stackrel{def}{\equiv} \\ \forall l \in \mathbb{N}, (0 \leq)\ \omega \leq (T_p(g_l) - T_r(f_l)) \end{gathered} \tag{29}$$

This constraint specifies that the lth message of g shall be emitted after the emission of the lth message of f plus a duration ω. It is illustrated in Fig. 17.

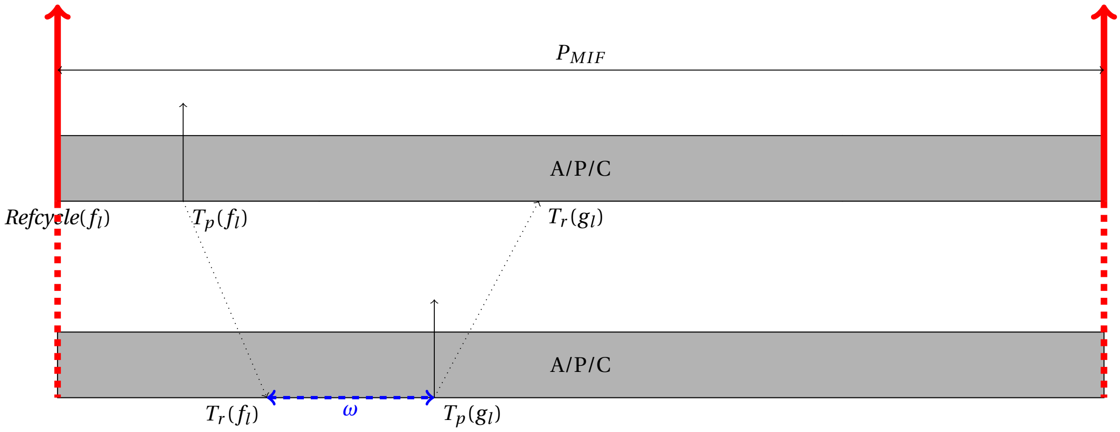

Figure 17 Reaction constraints with two flows f and g

1.2.6 Motivating Example: Adding Flow Constraints

Three flows have been identified in the motivating example (cf. 1.2.1). Let us now assign them flow constraints according to the previously defined constraints. We denote $\Upsilon(f)$ the function that, given a flow, returns its flow constraints. In the motivating example:

$$\Upsilon(f\_OBC\_HPActuator\_1) = \begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(\frac{3}{4} * P_{MIF}, P_{MIF}) \\ TimeZone(\frac{3}{4} * P_{MIF}, P_{MIF}) \end{cases} \tag{30}$$

$$\Upsilon(f\_OBC\_HPSensor\_1) = \begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \frac{1}{4} * P_{MIF}) \\ TimeZone(0, \frac{1}{4} * P_{MIF}) \\ Jitter(500\mu s) \end{cases} \tag{31}$$

$$\Upsilon(f\_HPSensor\_OBC\_1) = \begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \frac{1}{4}) * P_{MIF}) \\ TimeZone(0, \frac{1}{4} * P_{MIF}) \\ Reaction(\omega, f\_OBC\_HPSensor\_1) \end{cases} \tag{32}$$

1.3 Relaxing Hypothesis 2

In a previous section of this document, we have chosen through *Hypothesis* 2 that the maximum size of any message of any flow could not be greater than the Ethernet MTU. In fact, this allows us to state that one applicative message correspond to one Ethernet Frame. Relaxing Hyp. 2 will lead to having one applicative message being split (or fragmented) into several Ethernet frames. In that case, the concepts of *Emission Date* and *Reception Date* of a message shall be redefined.

▶ **Definition** 4. Emission Date
Let f be a flow ($\in \mathbb{F}$), where f does not satisfies Hyp. 2. Let f_l be the lth message of f. Let us denote *Frames*(f_l) the set of Ethernet frames corresponding to f_l. Let *First*(*Frames*(f_l) (resp. *Last*(*Frames*(f_l))) the first (resp. last) frame corresponding to f_l. The emission date of f_l, i.e. $T_p(f_l)$, correspond to the emission date (at applicative level) of the first bit of *First*(*Frames*(f_l)).

▶ **Definition** 5. Reception Date
Let f be a flow ($\in \mathbb{F}$), where f does not satisfies Hyp.2. Let f_l be the lth message of f. Let us denote *Frames*(f_l) the set of Ethernet frames corresponding to f_l. Let *First*(*Frames*(f_l) (resp. *Last*(*Frames*(f_l))) the first (resp. last) frame corresponding to f_l. The reception date of f_l, i.e. $T_r(f_l)$, correspond to the reception date (at applicative level) of the last bit of *Last*(*Frames*(f_l)).

With these newly adapted definitions, Hyp. 2 can be relaxed. The constraints previously introduced in this technical note are based on the concept of production date and reception date. Since they have just been redefined, the size of applicative message is no longer limited to the Ethernet MTU.

1.4 TSN Network

In our system, up to now, we have identified the devices and the flows. These flows travel through their messages between devices thanks to a TSN network. Let us define what this network is.

1.4.1 Definitions

A Time Sensitive Network is composed of End-Systems, switches and links. The End-systems are the devices defined in a previous section. The switches connect a set of links to another set of links. In a network topology, the end-systems can be connected with each other with a given number of links and switches.

▶ **Hypothesis** 4. In our system, we chose a topology with 2 switches, as represented in Fig.18. In this document, we denote this network by *Net*.

The elements that are left to be defined are the network links.

▶ **Definition** 6. Links
A link is a unidirectional path between a source node (i.e. end-system or switch) and a destination network node. We denote $\mathscr{L}$ the set of links in *Net*(the network we consider, defined in Hyp. 4). A link is characterized by the following tuple:

$$\forall v \in \mathscr{L}, < v.Start, v.End, v.Speed, v.PropagationDelay, v.Macrotick > \quad (33)$$

Where :

- $v.Start$ is a node (switch or end-system) at the start of the link,
- $v.End$ is a node (switch or end-system) at the end of the link,
- $v.Speed$ is the maximum speed on the link. We express it in Megabits per seconds,
- $v.PropagationDelay$ is a medium specific delay. It varies between physical mediums and links length. We express it in nanoseconds.
- $v.Macrotick$ is the length of a discrete time unit defining the granularity of time events for the given link. We express it in nanoseconds.

▶ **Hypothesis** 5. In our system, we only consider 1Gbit/s links i.e.:

$$\forall v \in \mathscr{L}, v.Speed = 1000 \quad (34)$$

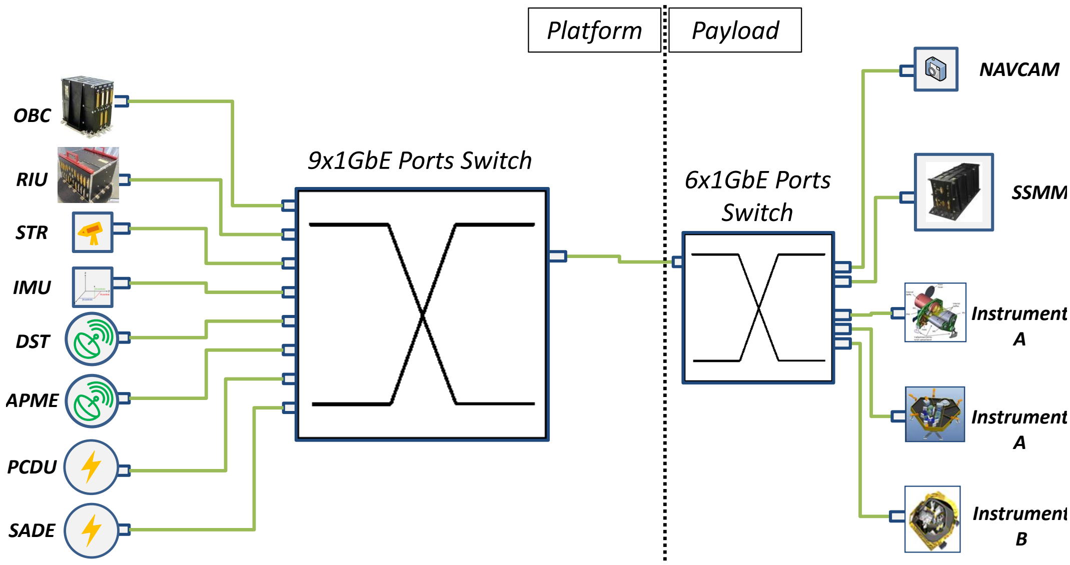

Figure 18 Network topology considered in our hypothesis

▶ **Hypothesis** 6. In our system, we consider that all links have the same *Macrotick* i.e.:

$$\forall va, vb \in \mathcal{L}, va.Macrotick = vb.Macrotick \tag{35}$$

1.4.2 Fault Tolerance Requirements

In this report, we have only formalized the network performance requirements. However, these are not the only requirements of the satellites on-board networks. Another category of requirements exits : the fault tolerance requirements. We will not detail these requirements in this report.

2 TSN Configuration Concept

In the prospect of using Time Sensitive Networking as the on-board network solution able to satisfy the previously introduced requirements, this section introduces the concept of *Configuration* of a TSN Network as well as some TSN vocabulary.

2.1 Vocabulary

In this section, we introduce some TSN vocabulary needed to understand the TSN configuration concept. These definitions are not introduced in IEEE TSN standards but reflect our understanding of Time Sensitive Networking.

▶ **Definition 1** (Feature). A *feature* refers to a TSN standard, amendment or on-going project. For instance, 802.1Qbv is a feature.

▶ **Definition 2** (Mechanism). A *mechanism* is a part of a feature. Thus a feature is a union of mechanisms. For instance, Time Aware Shaper is a mechanism of 802.1Qbv feature.

▶ **Definition 3** (Parameter). A *parameter* is a variable in a specific mechanism of a specific feature that can have several values. Thus a mechanism is a union of parameters. A mechanism has at least one parameter. For instance, the value and length of a Gate Control List (GCL) is a parameter of Time Aware Shaper mechanism.

2.2 Configuration

In this section, we introduce the concept of *Configuration* for a TSN Network.

▶ **Definition** 7. Configuration of a Network
Let *Net* be a network, let $\mathcal{N}$ be the set of nodes, of size n, in *Net*. A configuration of a network corresponds to a configuration for each node in the network. Let us denote $\Theta(Net)$ (resp. $\Theta(n)$) a configuration of the network *Net*(resp. of node d). Hence:

$$\forall n_1, ..., n_n \in \mathcal{N}, \Theta(Net) =< \Theta(n_1), ..., \Theta(n_n) > \quad (36)$$

▶ **Definition** 8. Configuration of a device
Let $d \in \mathcal{D}$ a node. Let *Features* be the set of all Time Sensitive Networking features. Let $Feat_1, ..., Feat_n$ be a subset of size n ($\in \mathbb{N}$) of *Features*. We denote $\Theta(Feat)$ a configuration of TSN feature *Feat*. A configuration of device d is characterized by the following tuple:

$$\forall Feat_1, ..., Feat_n \in Features, \Theta(n) =< \Theta(Feat_1), ..., \Theta(Feat_n) > \quad (37)$$

According to this definition, if a feature does not appear in the configuration of a device, this means that this feature will not be used in this device. Accordingly, if a feature appears in the configuration of the device, this means that it will have to be configured (see the following definition).

▶ **Definition** 9. Configuration of a feature
Let $Feat \in Features$ be a feature in the set of all TSN features. Let *Mechanisms(Feat)* be the set of all mechanisms of *Feat*. Let $Mechanism_1, ..., Mechanism_n$ be a subset of size n of *Mechanisms(Feat)*. We denote $\Theta(Mechanism)$ a configuration of a mechanism of *Feat*. A configuration of TSN feature *Feat* is characterized by the following tuple:

$$\begin{aligned} &\forall Mechanism_1, ..., Mechanism_n \in Mechanisms(Feat), \\ &\quad \Theta(Feat) =< \Theta(Mechanism_1), ..., \Theta(Mechanism_n) > \end{aligned} \quad (38)$$

According to this definition, if a mechanism does not appear in the configuration of a feature, this means that this mechanism will not be used in this feature. Accordingly, if a mechanism appears in the configuration of the feature, this means that it will have to be configured (see the following definition).

▶ **Definition** 10. Configuration of a mechanism
Let $Feat$ be a TSN feature, let $Mechanism$ be a mechanism of $Feat$. Let *Parameters(Mechanism)* be the set of parameters of mechanism. Let $param_1, ..., param_n$ be a subset of size n of *Parameters(Mechanism)*. We denote $\Theta(param)$ the value of parameter param. A configuration of TSN mechanism *Mechanism* of feature *Feat* is characterized by the following tuple:

$$\forall param_1, ..., param_n \in Parameters, \Theta(Mechaninsm) =< \Theta(param_1, ..., param_n) > \quad (39)$$

According to this definition, if a parameter does not appear in the configuration of a mechanism, this means that this parameter will not be used in this mechanism. Accordingly, if a parameter appears in the configuration of the mechanism, this means that it will have to be given a value.

▶ **Definition** 11. Flow to Traffic Class mapping
Let f be a flow in the set of flows $\mathbb{F}$ of *Net*, let $\mathcal{N}$ be the set of nodes of *Net*. Let $n_1, ..., n_n$ be a subset of

$\mathcal{N}$ corresponding to the nodes in the path of f, let us denote it $Path(f)$. We denote TC(f, d) the traffic class of flow f in node *N*. The Flow to Traffic Class mapping of *Net* is characterized by the following tuple:

$$\forall f \in \mathbb{F}, \forall d \in Path(f), Mapping(Net) = (< TC(f,d) >) \tag{40}$$

With this definition, we introduce the possibility for a flow to change of traffic classes between nodes. In all cases (802.Q-2018 property, see chapter 3):

$$\forall f \in \mathbb{F}, \forall d \in Path(f), 1 \geq TC(f,d) \leq 8 \tag{41}$$

▶ **Definition** 12. Configuration of a system
Let S be the system we consider. We denote $\Theta(S)$ a configuration of S. Let *Net* be the network of your system. A configuration of S is characterized by the following tuple:

$$\Theta(S) =< \Theta(Net), Mapping(Net) > \tag{42}$$

This means that a configuration of the system is composed of a network configuration and a flows to traffic class mapping.

3 Satellite On-board Network Use Case

In this section, we instantiate the model and constraints introduced above to a Satellite On-Board Network use case.

3.1 Network topology

The system we consider is a network composed of:

- One computing device, named *On-Board Computer* (OBC)
- Two sensing devices, named *Star Tracker* (STR) and *Navigation Camera* (NAVCAM)
- One actuating device connected in the network through a *Remote Interface Unit* (RIU)
- Three payload devices, named *Mass Memory* (SSMM), *Instrument 1* (Instr. 1) and *Instrument 2* (Instr. 2).

These devices are connected together through a set of links and two switches, according to the topology of Fig. 19.

3.2 Applications

In this satellite use case, we consider three applications:

- One *Command & Control* application
- One *Vision Based Navigation (VBN)* application
- One *Payload* application

This section will describe the parameters of these applications. Note that we now denote the *input* part as the Acquisition part and the *output* part as the Command part.

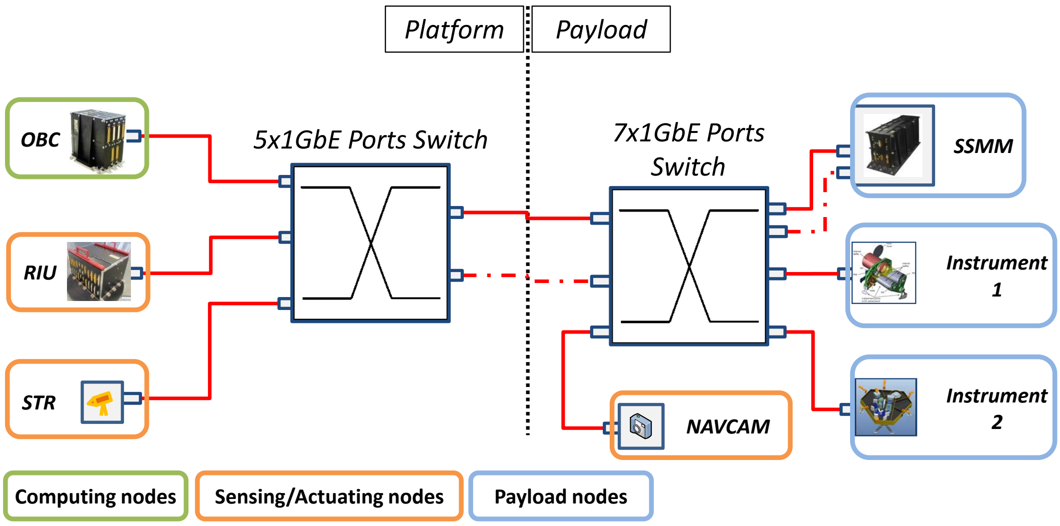

Figure 19 Network topology and devices considered in the Satellite use case

3.2.1 Command & Control application use case

This C & C application is running on the OBC (Platform Computing Device). Flows runs between the OBC, the STR and the RIU. For this application, we choose:

$$k = 8 \text{ hence } P_{MIF} = 125ms \quad (43)$$

Within one cycle (of length P_{MIF}), we specify the duration of the Acquisition, Processing and Command part. We choose:

$$\delta t_1 = \frac{1}{4}, \delta t_2 = \frac{1}{2}, \delta t_3 = \frac{1}{4} \quad (44)$$

3.2.2 VBN application use case

This VBN application is running on the OBC (Platform Computing Device). Flows runs between the OBC and the NAVCAM. For this application, we choose:

$$k = 30 \text{ hence } P_{MIF} = 33{,}33ms \quad (45)$$

Within one cycle (of length P_{MIF}), we specify the duration of the Acquisition, Processing and Command part. We choose:

$$\delta t_1 = \frac{1}{3}, \delta t_2 = \frac{1}{3}, \delta t_3 = \frac{1}{3} \quad (46)$$

3.2.3 Payload application use case

This C & C application is running on the OBC (Platform Computing Device) and on the Payload Devices. Flows runs between the OBC, the INSTR1, the INSTR2 and the SSMM. For this application, we choose:

$$k = 1000 \text{ hence } P_{MIF} = 1ms \quad (47)$$

Within one cycle (of length P_{MIF}), we specify the duration of the Acquisition, Processing and Command part. We choose:

$$\delta t_1 = \frac{P_{MIF}}{2}, \delta t_2 = \frac{P_{MIF}}{2}, \delta t_3 = 0 \tag{48}$$

3.3 Flows

In this satellite use case, we consider that the applications running on the different devices send their messages through several flows. Let us now define these flows and their constraints with the formalism of Section 1. For easier readability, the flow's definition are gathered in Tables 2 to 8.

3.3.1 C & C application use case : adding flows

Name	Source	Dest	Max. Data Size	r_f
f_RIU_OBC_HK_1	RIU	OBC	64	-8
f_RIU_OBC_HK_2	RIU	OBC	64	-8
f_RIU_OBC_HK_3	RIU	OBC	64	-8
f_RIU_OBC_HK_4	RIU	OBC	64	-8
f_RIU_OBC_HK_5	RIU	OBC	64	-8
f_RIU_OBC_HK_6	RIU	OBC	64	-8
f_RIU_OBC_HK_7	RIU	OBC	64	-8
f_RIU_OBC_HK_8	RIU	OBC	64	-8
f_RIU_OBC_HK_9	RIU	OBC	64	-8
f_RIU_OBC_HK_10	RIU	OBC	64	-8
f_RIU_OBC_HK_11	RIU	OBC	64	-1
f_RIU_OBC_HK_12	RIU	OBC	64	-1
f_RIU_OBC_FDIR_1	RIU	OBC	64	-8
f_RIU_OBC_FDIR_2	RIU	OBC	64	-8
f_RIU_OBC_FDIR_3	RIU	OBC	64	-8
f_RIU_OBC_FDIR_4	RIU	OBC	64	-8
f_RIU_OBC_FDIR_5	RIU	OBC	64	-8
f_RIU_OBC_FDIR_6	RIU	OBC	64	-8
f_RIU_OBC_FDIR_7	RIU	OBC	64	-1
f_RIU_OBC_FDIR_8	RIU	OBC	64	-1
f_RIU_OBC_FDIR_9	RIU	OBC	64	-8
f_RIU_OBC_FDIR_10	RIU	OBC	64	-8
f_RIU_OBC_FDIR_11	RIU	OBC	64	-8
f_RIU_OBC_FDIR_12	RIU	OBC	64	-8
f_RIU_OBC_MEO_1	RIU	OBC	64	-1
f_RIU_OBC_MEO_2	RIU	OBC	64	-1
f_RIU_OBC_MEO_3	RIU	OBC	64	-1
f_RIU_OBC_MEO_4	RIU	OBC	64	-1
f_RIU_OBC_MEO_5	RIU	OBC	64	-1
f_RIU_OBC_MEO_6	RIU	OBC	64	-1
f_RIU_OBC_MEO_7	RIU	OBC	64	-1
f_RIU_OBC_MEO_8	RIU	OBC	64	-1
f_RIU_OBC_ACK_1	RIU	OBC	64	-1
f_RIU_OBC_ACK_2	RIU	OBC	64	-1
f_RIU_OBC_ACK_3	RIU	OBC	64	-1
f_RIU_OBC_ACK_4	RIU	OBC	64	-1
f_RIU_OBC_ACK_5	RIU	OBC	64	-1
f_RIU_OBC_ACK_6	RIU	OBC	64	-1
f_RIU_OBC_ACK_7	RIU	OBC	64	-1
f_RIU_OBC_ACK_8	RIU	OBC	64	-1
f_RIU_OBC_ACK_9	RIU	OBC	64	-1
f_RIU_OBC_ACK_10	RIU	OBC	64	-1
f_RIU_OBC_DATA_1	RIU	OBC	64	-1
f_RIU_OBC_DATA_2	RIU	OBC	64	-1

Table 2 Flow definition for the satellite C&C use case, device RIU

Name	Source	Dest	Max. Data Size	r_f
f_OBC_RIU_HK_1	OBC	RIU	64	-8
f_OBC_RIU_HK_2	OBC	RIU	64	-8
f_OBC_RIU_HK_3	OBC	RIU	64	-8
f_OBC_RIU_HK_4	OBC	RIU	64	-8
f_OBC_RIU_HK_5	OBC	RIU	64	-8
f_OBC_RIU_HK_6	OBC	RIU	64	-8
f_OBC_RIU_HK_7	OBC	RIU	64	-8
f_OBC_RIU_HK_8	OBC	RIU	64	-8
f_OBC_RIU_HK_9	OBC	RIU	64	-8
f_OBC_RIU_HK_10	OBC	RIU	64	-8
f_OBC_RIU_HK_11	OBC	RIU	64	-1
f_OBC_RIU_HK_12	OBC	RIU	64	-1
f_OBC_RIU_FDIR_1	OBC	RIU	64	-8
f_OBC_RIU_FDIR_2	OBC	RIU	64	-8
f_OBC_RIU_FDIR_3	OBC	RIU	64	-8
f_OBC_RIU_FDIR_4	OBC	RIU	64	-8
f_OBC_RIU_FDIR_5	OBC	RIU	64	-8
f_OBC_RIU_FDIR_6	OBC	RIU	64	-8
f_OBC_RIU_FDIR_7	OBC	RIU	64	-1
f_OBC_RIU_FDIR_8	OBC	RIU	64	-1
f_OBC_RIU_FDIR_9	OBC	RIU	64	-8
f_OBC_RIU_FDIR_10	OBC	RIU	64	-8
f_OBC_RIU_FDIR_11	OBC	RIU	64	-8
f_OBC_RIU_FDIR_12	OBC	RIU	64	-8
f_OBC_RIU_MEO_1	OBC	RIU	64	-1
f_OBC_RIU_MEO_2	OBC	RIU	64	-1
f_OBC_RIU_MEO_3	OBC	RIU	64	-1
f_OBC_RIU_MEO_4	OBC	RIU	64	-1
f_OBC_RIU_MEO_5	OBC	RIU	64	-1
f_OBC_RIU_MEO_6	OBC	RIU	64	-1
f_OBC_RIU_MEO_7	OBC	RIU	64	-1
f_OBC_RIU_MEO_8	OBC	RIU	64	-1
f_OBC_RIU_CMD_1	OBC	RIU	64	-1
f_OBC_RIU_CMD_2	OBC	RIU	64	-1
f_OBC_RIU_CMD_3	OBC	RIU	64	-1
f_OBC_RIU_CMD_4	OBC	RIU	64	-1
f_OBC_RIU_CMD_5	OBC	RIU	64	-1
f_OBC_RIU_CMD_6	OBC	RIU	64	-1
f_OBC_RIU_CMD_7	OBC	RIU	64	-1
f_OBC_RIU_CMD_8	OBC	RIU	64	-1
f_OBC_RIU_CMD_9	OBC	RIU	64	-1
f_OBC_RIU_CMD_10	OBC	RIU	64	-1
f_OBC_RIU_ACQUI_1	OBC	RIU	64	-1
f_OBC_RIU_ACQUI_2	OBC	RIU	64	-1

Table 3 Flow definition for the satellite C&C use case, device OBC-Part 1

Name	Source	Dest	Max. Data Size	r_f
f_STR_OBC_HK_1	STR	OBC	64	-1
f_STR_OBC_HK_2	STR	OBC	64	-1
f_STR_OBC_FDIR	STR	OBC	64	-1
f_STR_OBC_MEO_1	STR	OBC	64	-1
f_STR_OBC_MEO_2	STR	OBC	64	-1
f_STR_OBC_ACK_1	STR	OBC	64	-1
f_STR_OBC_ACK_2	STR	OBC	64	-1
f_STR_OBC_ACK_3	STR	OBC	64	-1
f_STR_OBC_ACK_4	STR	OBC	64	-1
f_STR_OBC_ACK_5	STR	OBC	64	-1
f_STR_OBC_DATA	STR	OBC	1088	-1

Table 4 Flow definition for the satellite C&C use case, device STR

Name	Source	Dest	Max. Data Size	r_f
f_OBC_STR_HK_1	OBC	STR	64	-1
f_OBC_STR_HK_2	OBC	STR	64	-1
f_OBC_STR_FDIR	OBC	STR	64	-1
f_OBC_STR_MEO_1	OBC	STR	64	-1
f_OBC_STR_MEO_2	OBC	STR	64	-1
f_OBC_STR_CMD_1	OBC	STR	64	-1
f_OBC_STR_CMD_2	OBC	STR	64	-1
f_OBC_STR_CMD_3	OBC	STR	64	-1
f_OBC_STR_CMD_4	OBC	STR	64	-1
f_OBC_STR_CMD_5	OBC	STR	64	-1
f_OBC_STR_ACQUI	OBC	STR	64	-1
f_OBC_INSTR1_HK	OBC	INSTR1	128	-4
f_OBC_INSTR2_HK	OBC	INSTR1	128	-4
f_OBC_NAVCAM_HK	OBC	NAVCAM	128	-4

Table 5 Flow definition for the satellite C&C use case, device OBC-Part 2

Name	Source	Dest	Max. Data Size	r_f
f_INSTR1_OBC_HK	OBC	INSTR1	128	-4
f_INSTR2_OBC_HK	OBC	INSTR1	128	-4
f_NAVCAM_OBC_HK	OBC	NAVCAM	128	-4

Table 6 Flow definition for the satellite C&C use case, device INSTR1-2 and NAVCAM

3.3.2 VBN application use case : adding flows

Name	Source	Dest	Max. Data Size	r_f
f_NAVCAM_OBC_SCI	NAVCAM	OBC	1048576	-1

Table 7 Flow definition for the satellite VBN use case, device INSTR1-2

3.3.3 Payload application use case : adding flows

Name	Source	Dest	Max. Data Size	r_f
f_INSTR1_SSMM_SCI	INSTR1	SSMM	4096	-3
f_INSTR2_SSMM_SCI	INSTR2	SSMM	4096	2

Table 8 Flow definition for the satellite payload use case, device INSTR1-2

3.4 Adding Flow Constraints

Now that the flows of our system have been identified, let us assign them some flow constraints. Again, for easier readability, the flow constraints are gathered in Tables 9 to 17.

Name (Suffix)	$\Upsilon(f)$
f_* (all flows)	ReceptionAfterEmission()
f_* (all flows)	Ordered Emission()
f_* (all flows)	Ordered Delivery()
f_* (all flows)	Ethernet Frame Size()

Table 9 Flow constraints definition for the satellite use case

Please note that the parameters of the Reaction and MinimumSpace Constraints have not be fully computed at this stage.

3.4.1 C & C application use case : adding flows constraints

Name (Suffix)	$\Upsilon(f)$
f_STR_OBC_HK_*	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_1, f\_OBC\_STR\_HK\_*) \end{cases}$
f_STR_OBC_FDIR_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_2, f\_OBC\_STR\_FDIR\_*) \end{cases}$
f_STR_OBC_MEO_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_3, f\_OBC\_STR\_MEO\_*) \end{cases}$
f_STR_OBC_ACK_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_4, f\_OBC\_STR\_CMD\_*) \end{cases}$
f_STR_OBC_DATA_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_5, f\_OBC\_STR\_ACQUI\_*) \end{cases}$

Table 10 Flow constraints definition for the satellite C&C use case, device STR

f_RIU_OBC_HK_*	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_6, f\_OBC\_RIU\_HK\_*) \end{cases}$
f_RIU_OBC_FDIR_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_7, f\_OBC\_RIU\_FDIR\_*) \end{cases}$
f_RIU_OBC_MEO_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_8, f\_OBC\_RIU\_MEO\_*) \end{cases}$
f_RIU_OBC_ACK_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_9, f\_OBC\_RIU\_CMD\_*) \end{cases}$
f_RIU_OBC_DATA_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, P_{MIF}) \\ TimeZone(0, P_{MIF}) \\ Reaction(\omega_{10}, f\_OBC\_RIU\_ACQUI\_*) \end{cases}$

Table 11 Flow constraints definition for the satellite C&C use case, device RIU

f_OBC_STR_HK_*	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$
f_OBC_STR_FIDR_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$
f_OBC_STR_MEO_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ MinimumSpace(\gamma_1) \end{cases}$
f_OBC_STR_CMD_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone((\Delta(Input) + \Delta(Processing)) * P_{MIF}, P_{MIF}) \\ TimeZone((\Delta(Input) + \Delta(Processing)) * P_{MIF}, P_{MIF}) \\ Jitter(1\mu s) \end{cases}$
f_OBC_STR_ACQUI_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \Delta(Input) * P_{MIF}) \\ TimeZone(0, \Delta(Input) * P_{MIF}) \\ Jitter(500\mu s) \end{cases}$
f_OBC_RIU_HK_*	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$
f_OBC_RIU_FDIR_*	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$

Table 12 Flow constraints definition for the satellite C&C use case, device OBC part. 1

f_OBC_RIU_MEO	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ MinimumSpace(\gamma_2) \end{cases}$
f_OBC_RIU_CMD	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone((\Delta(Input) + \Delta(Processing)) * P_{MIF}, P_{MIF}) \\ TimeZone((\Delta(Input) + \Delta(Processing)) * P_{MIF}, P_{MIF}) \\ Jitter(1\mu s) \end{cases}$
f_OBC_RIU_ACQUI	$\begin{cases} PeriodicMessageProduction() \\ ImplicitDeadlines() \\ InjectionZone(0, \Delta(Input) * P_{MIF}) \\ TimeZone(0, \Delta(Input) * P_{MIF}) \\ Jitter(500\mu s) \end{cases}$
f_OBC_INSTR1_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$
f_OBC_INSTR2_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$
f_OBC_NAVCAM_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \end{cases}$

Table 13 Flow constraints definition for the satellite C&C use case, device OBC part. 2

f_INSTR1_OBC_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_{11}, f\_OBC\_INSTR1\_HK\_*) \end{cases}$
f_INSTR2_OBC_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_{12}, f\_OBC\_INSTR2\_HK\_*) \end{cases}$

Table 14 Flow constraints definition for the satellite C&C use case, device INSTR1, INSTR2

f_NAVCAM_OBC_HK	$\begin{cases} InjectionZone(0, \lvert r_f \rvert * P_{MIF}) \\ TimeZone(0, \lvert r_f \rvert * P_{MIF}) \\ Reaction(\omega_{13}, f\_OBC\_NAVCAM\_HK\_*) \end{cases}$

Table 15 Flow constraints definition for the satellite C&C use case, device NAVCAM

3.4.2 VBN application use case : adding flows constraints

f_NAVCAM_OBC_SCI	$\begin{cases} ControlledInputTrafficShape(R, B) \\ ControlledOutputTrafficShape(d, N_{Frames}) \end{cases}$

Table 16 Flow constraints definition for the satellite VBN use case, device NAVCAM

▶ Remark. The parameters of the ControlledInputTrafficShape and ControlledOutputTrafficShape constraints for NAVCAM have not been computed yet.

3.4.3 Payload application use case : adding flows constraints

f_INSTR1_SSMM_SCI	$\begin{cases} ControlledInputTrafficShape(R,B) \\ ControlledOutputTrafficShape(d,N_{Frames}) \end{cases}$
f_INSTR2_SSMM_SCI	$\begin{cases} ControlledInputTrafficShape(R,B) \\ ControlledOutputTrafficShape(d,N_{Frames}) \end{cases}$

Table 17 Flow constraints definition for the satellite Payload use case, device INSTR1-2

▶ Remark. The parameters of the ControlledInputTrafficShape and ControlledOutputTrafficShape constraints for INSTR1 and INSTR2 have not been computed yet.