Toward Self-Reconfiguration of Switched Ethernet Architectures in the Next Generation of Space Launchers

Jérémy Robert[1,2], Jean-Philippe Georges[1,2], Thierry Divoux[1,2]
[1]Université de Lorraine, CRAN, UMR 7039, Campus Sciences - BP 70239,
F-54506 Vandœuvre-lès-Nancy, France
[2]CNRS, CRAN, UMR7039, France
Email: firstname.name@univ-lorraine.fr

***Abstract*—Nowadays, many embedded systems use specific data buses (like CAN in vehicles or MIL-STD-1553B in launchers) to ensure the exchange of data. To reduce the cost, the mass and to increase performance, a solution is to rely on a components off-the-shelf (COTS) technology. As switched Ethernet is a well-known solution and widely implemented, this technology is studied for the next generation of space launchers. In this paper, we focus on the reliability issue. In case of network components failure and without reconfiguration mechanism, the network will not carry on control data anymore. Performance of Ethernet reconfiguration solutions may be not enough efficient regarding real-time constraints. This paper aims hence at coping with real-time guarantees of self reconfiguration for Ethernet networks, especially in the space launchers. An experimental evaluation has been leaded on a real case study during a Centre National d'Etudes Spatiales (CNES) French Research & Technology (R&T) activity.**

Keywords- fault-tolerant network; industrial Ethernet; real-time.

I. INTRODUCTION

Nowadays, many embedded systems (aircrafts, satellites, launchers) use several specific data buses which ensure the exchange of either the control data or the service data (measurement or supervision). Particularly, the Ariane 5 launchers use a MIL-STD-1553B data bus [1] to transport the control data while the telemetry data flow is managed by a dedicated (and proprietary) bus. The MIL-STD-1553B bus relies on a centralized communication scheme inherent to its command/response mechanism. As illustrated in Fig. 1, the duplex architecture was seen as a way to ensure reliability (this choice was not derived inherently from system requirements). In this architecture, the bus and the end-nodes are therefore redundant. When a failure occurs on the bus (or a bus segment), the control of the launcher is based only on the redundant sensors data (not concerned by the failure). As a consequence, the communication system is tolerant to a single failure.

Currently, there are many new objectives for the next generation of space launchers [2]:

- reduction of the total cost of the launch system,
- avionics mass reduction and
- avionics performance improvement (keeping at least the same reliability)

To meet these objectives, there is an attempt to rely on an unique components off-the-shelf (COTS) technology. In this context, the different networks could be unified in a single communication system which will support more and more traffics or even large data (such as images, videos or extended telemetry data). As a consequence, the bus MIL-STD-1553B become unsuitable (due to its low bandwidth).

Among several COTS networks, switched Ethernet is the most cost and mature effective solution [3]. Indeed, this technology is a well-known standard and widely implemented in several domains [4] such as manufacturing systems, automotive [5], transport, aviation [6], motion control [7]. As in all industrial area, it reclaims real-time and reliability requirements, such that the network must fulfill with Quality of Service (QoS) criteria like end-to-end delays, jitter, losses and availability. Several works have dealt with the temporal requirements of switched Ethernet architectures [8], [9], [10]. From this temporal aspect, the feasibility of switched Ethernet architecture (depicted in Fig. 2) in the next generation of space launchers has been highlighted in [11]. The purpose of this paper is to increase the robustness of switched Ethernet architectures up to (at least) the same reliability level than with MIL-STD-1553B.

As in all systems, the reliability in a switched Ethernet architecture can be increased with redundancy. This one can be classified into two categories (see IEC SC65C WG15 and [12]): "redundancy in the network" or "redundancy in the nodes". The first one suggests to extend the network topology with redundant links (and even switches): end stations are attached to a single switch only, and hence interconnected to a single network. In the second one, the nodes have on the contrary (at least) two ethernet cards connected to two (parallel) redundant Ethernet networks.

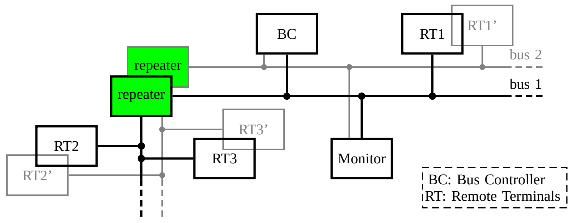

Figure 1. Overview of current MIL-STD-1553B architecture

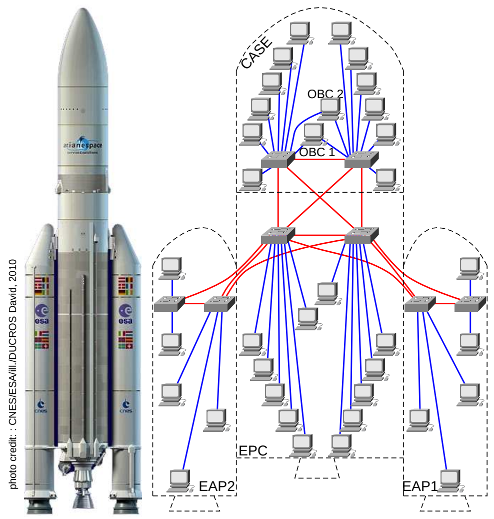

Figure 2. Physical meshed architecture

In reality, MIL-STD-1553B solution is based on a duplex architecture. However, it could lead to cost increase and all messages sent by all end nodes on the bus concerned by failure are still lost (the application was based only on the redundant sensors data). As introduced in [12], we suggest to refer to a "redundancy in the network" solution in which the network can reconfigure itself to ensure the continuity of service. It may be noticed that Automatic Repeat reQuest (ARQ) protocols can improve the reliability, but regarding the real-time requirement, none are considered here. To face with a failure of the local interconnection point of a device, redundancy of the end nodes will be used. Our objectives are therefore to have a reconfiguration mechanism transparent for the application point of view, and to minimize the time of the reconfiguration action (and so the losses). It corresponds to a real-time requirement for the reconfiguration strategy [13], and will be qualified in the following as:

THE RECOVERY TIME: *the time necessary to switch to a backup solution and ensure the continuity of service. An application can tolerate a limited time of interruption of service, called grace time [14].*

To minimize the recovery time, the literature reports many solutions [15] which rely on redundancy management. Redundancy is here supported by a physical meshed architecture. Relevant protocols in this category, Media Redundancy Protocol (MRP) [14] and Rapid Spanning-Tree Protocol (RSTP) [16] are based on two main steps: fault detection and topology reconfiguration. Fault detection is typically achieved by the asynchronous periodic transmission of particular frames between switches. Recovery time issues for real-time systems are coming from these exchanges and from the asynchronism due to the distribution of the algorithm. In order to reduce the recovery time, MRP protocol considers a centralized approach, but only for a ring topology. For all these reasons, this paper will suggest a novel centralized approach for the physical topology supervision and the logical topology reconfiguration. Another feature is that it can be applied in any physical meshed topology.

The remainder of the paper continues as follows: the section II presents an overview of our proposition and describes formally the proposal. Section III illustrates the proposed method by its application to a real case study obtained in the framework of a "CNES french R&T activity" and highlights the performance of our solution. Finally, in section IV some conclusions and future work are given.

II. Research Proposal: Autonomic Switched Ethernet Reconfiguration

The objective of the paper is to suggest a fast recovery solution enabling to ensure fast recovery time in harsh environment. In the framework of CNES activities, this paper deals primarily with space applications in which typical expected QoS criteria are : 1 ms end-to-end delays, any loss of frames, no current requirements in terms of bandwidth (note that in the future, it could be the case, e.g., for the telemetry traffic). This solution is particularly proposed for any meshed switched topologies.

A. Proposition overview

The core idea of our proposition is to reject the logical trees to control all paths in a deterministic way. It is suggested to apply a layer 2 routing to control the "route" taken by each flow. This consists of filling in the forwarding table a priori. Indeed, this table acts as a filter which enables the switch to match an incoming frame to an appropriate MAC-destination address entry and forward the frame to the next hop (with a specific output port) towards the final destination as specified in this table.

Let us remember that the objective of our proposition is that the network could reconfigure itself when a failure occurs. As a consequence, the network should be able to detect a failure and modify (or more exactly reconfigure) the forwarding table to restore an active (and known) path. To achieve this, we suggest to introduce a central agent capable of detecting any failure and reconfiguring an active logical topology. This agent has a global view of the network (and so of the network state). This view can be obtained by requesting all switches periodically (*via* Simple Network Management Protocol - SNMP). Indeed, any switches store the states of its ports in a specific database called Management Information Base (MIB). This information is updated thanks to a physical link integrity function (defined in the standard 802.3) on any switches.

Therefore, from the global view of the network, the central agent can detect a link or a switch failure. In case of link failure, the port state (concerned by the failure) is modified (from "up" to "down" state). When a switch failure occurs, the switch is not reachable anymore (no response to the periodic request).

Once a failure detected, the central agent sends a reconfiguration message in order to modify a (or many) output port(s) in the forwarding table. Although exchanges induced by this reconfiguration mechanism will lead to an over-consumption of the network capacity, it will remain highly limited compared to a capacity of 100 Mb/s and to the current amount of traffic (around 500 Kb/s) for Ariane 5 launcher. It leads to a set of new paths between two nodes. As in RSTP, paths will be computed along a spanning-tree. In this proposition, paths with unsuitable end-to-end delays will be rejected [11]. In fact, paths will be computed by a routing algorithm such as Dijkstra's algorithm or Bellman-Ford algorithm. To maintain a deterministic control, they will be computed off-line before the configuration of the network, i.e. in the design phase. The number of paths will be related to the number of flows and to the reliability (e.g. for a single recovery, at least two times the number of flows, but in reality less).

The next subsection presents in a formal way, the reconfiguration algorithm which will be implemented in the central agent. It is planed that hardware implementation issues will be dealt with as well as the distribution of this functionality for facing with this central point of failure.

B. Formal considerations

In a formal way, we consider a network graph $G = (X, A)$ where X is the matrix containing the state of all switch ports and A the matrix representing the physical connectivity between the switches. Let n the number of switches in the architecture and m the number of ports that a switch has (in this paper, it is assumed that all switches have the same number of ports). The element of X is written as $X_{i,j} \in \{0, 1\}$ which represents the state of switch $j \in \{1, n\}$ port i with $i \in \{1, m\}$. X is defined such as $X = [X_1 \dots X_j \dots X_n]$ where X_j represents the port states of switch j. The element of A is noted as $A_{(i,j),(i',j')} \in \{0, 1\}$ represents the physical capacity to communicate between the port i of the switch j and the port i' of switch j'. This network graph will have to be updated periodically as written in line 3 of Algorithm 1. Indeed, in monitoring the states of ports (the matrix X), the network graph is accordingly modified. It is important to note that the performance of our solution will depend on the value of this monitoring period. This criterion will be evaluated in the next section.

Let us remember that our goal is to fill in or update the forwarding table. For that, we have to look for the output ports associated to each destination on all switches. A list D of all stations (represented by their MAC addresses ($@MAC$)) is defined. It is necessary to know on which port of which switch are located the end-stations. A matrix D' is thereby defined to contain the network attachment of the different stations such as $D'_{i,j} \in D$ represents the connection of end stations (with MAC address $@MAC$) on the port i of switch j. As a matter of course, for each switch and for each

```
Data: $G = (X, A_{(i,j),(i',j')})$ : the network graph;
      $D$ : the list of all stations;
      $D'$ : the network attachment of the different stations;
Result: the vector $R_j =< @MAC, i >$ with $@MAC$ the destination and $i$
        the output port to reach $@MAC$ (next hop) from the switch $j$
1 begin
2     Periodically
3         $G \longleftarrow$ update($G$);
4         foreach switch $j$ do
5             foreach $@MAC \in D$ do
6                 (i,j') | $D'_{i,j'} = @MAC$ ;
7                 $R_j \longleftarrow$ Bellman-Ford($j \rightarrow j'$, $G$);
```

Algorithm 1: Reconfiguration algorithm

MAC address in the list D, we can identify on which switch j' is located an end-station (see lines 4-6 in Algorithm 1).

This information and the updated network graph are the inputs of the well-known Bellman-Ford routing algorithm (see line 7 in Algorithm 1). The result of this step is the next hop to reach the destination. Knowing the switch to cross and the physical connectivity (the matrix A), we can deduce the output port to fill in the forwarding table. Of course, if it is concerned by the failure, it is modified (see lines 8-9 in Algorithm 1) in the forwarding table.

III. Experimental Work

Our research laboratory collaborates closely with the CNES to lead R&T activities. In this framework, the objective is to reach a certain level maturity of switched Ethernet technology for the future Ariane launchers. This level can be assessed by the Technology Readiness Level (TRL) [17]. In this work, the TRL 3 is expected and consists to study separately each elements (and problem area) of the technology. It should constitute a “proof-of-concept” without being in an operational environment. As a consequence, a computer network is used to show the feasibility of our algorithm, the reachable interruption time of service and the impact of our algorithm on the number of lost frames. However, a real applicative traffic is considered to perform the laboratory tests and to measure parameters of interest.

The tested architecture is depicted on the Fig. 3. It corresponds to the upper stages (Cryogenic Upper Stage - CASE - and Cryogenic Propulsion stage - EPC - as shown in Fig. 2). In this experiments, the switches are Cisco 2950 series (IOS version: 12.1(22)EA1). In a formal way, this example can be initially described by the matrix X, A, D' and the vector D as can be seen in (1) and (2).

$$X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix} D = \begin{pmatrix} @MAC1 \\ @MAC2 \\ @MAC3 \\ @MAC4 \\ @MAC5 \\ @MAC6 \\ @MAC7 \\ @MAC8 \\ @MAC9 \\ @MAC10 \\ @MAC11 \\ @MAC12 \\ @MAC13 \end{pmatrix} D' = \begin{pmatrix} @MAC1 & 0 & @MAC8 & 0 \\ @MAC2 & 0 & @MAC9 & 0 \\ @MAC3 & 0 & @MAC10 & 0 \\ @MAC4 & 0 & @MAC11 & 0 \\ @MAC5 & 0 & @MAC12 & 0 \\ @MAC6 & 0 & @MAC13 & 0 \\ @MAC7 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad (1)$$

$dim(X)= 24 \times 4,\ dim(D)= 13 \times 1\ and\ dim(D')= 24 \times 4$

$$A = \begin{array}{c|ccccccccccccccccc}
 & (1,1) & \dots & (22,1) & (23,1) & (24,1) & \dots & (22,2) & (23,2) & (24,2) & \dots & (22,3) & (23,3) & (24,3) & \dots & (22,4) & (23,4) & (24,4) \\
\hline
(1,1) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
\vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
(22,1) & 0 & \dots & 0 & 0 & 0 & \dots & 1 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
(23,1) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 1 & 0 \\
(24,1) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 & 0 \\
\vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
(22,2) & 0 & \dots & 1 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
(23,2) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 & 0 \\
(24,2) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 1 \\
\vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
(22,3) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 & 0 & 0 \\
(23,3) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
(24,3) & 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
\vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
(22,4) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 1 & 0 & 0 & \dots & 0 & 0 & 0 \\
(23,4) & 0 & \dots & 0 & 1 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0 \\
(24,4) & 0 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 1 & \dots & 0 & 0 & 0 & \dots & 0 & 0 & 0
\end{array} \qquad (2)$$

In this real scenario, the centralized communication mode is considered: all traffic flows are sent to the destination device 1. The features of these flows are described in the Table I. Each Ethernet frames are stamped and numbered. These two information enable to determine the delay and the inter-arrival time. A failure is simulated by unplugging a network cable at a random time. The application packet loss involves an inter-arrival time superior to the inter-arrival time without loss. It corresponds therefore to a service interruption time (time of non-reception of application frames).

Before considering all traffic flows, a periodic traffic (with a period equals to 1 ms and a size to 72 $bytes$) sent by device 9 to device 1 has been studied. A failure has been simulated on the link connected to the port 24 on the switch $Sw3$. The interruption time has therefore been measured according to a Monte-Carlo scenario (10 first experiments, to be continued) for several monitoring periods: 100 μs, 1 ms, 10 ms and 100 ms. The obtained results are depicted in the Fig. 4.

The Fig. 4 shows that for any period (100 μs, 1 ms, 10 ms and 100 ms), the interruption time is around 200 ms. The variation of the interruption time can be explained by the detection time of "link_loss" which is between 50 ms and 150 ms (defined in the standard 802.3 [18]), plus the reconfiguration time and the update time in the MIB. Among the tested monitoring period, the 100 ms period is the closest to the detection time. That's why, it can be considered a period equals to 100 ms. Moreover, when the frame with the forwarding update is sent, we record this date. We can therefore deduce a (bounded) approximation of the reconfiguration time which is the difference between the end of interruption of service and this date. The reconfiguration time is around 20 ms.

A second test has been leaded on three of the flows which are the flow 1 on the device 13, the flow 2 on the device 9 and the flow 3 on the device 8. These flows are interesting because of their different periods (288 ms, 72 ms and 18 ms). The objective is to highlight the impact of our algorithm on the number of lost frames. The same failure as previously (on the link Sw1-Sw3) is considered. The forwarding table of the switch $Sw3$ is initially configured as shown in Table II.

Consider, for instance, the flows of the device 13 (to device 1) and the link Sw1-Sw3 failure. As the graph is updated periodically (see lines 2-3 of Algorithm 1), the failure will be observed in the matrix X (once the failure is electrically detected *via* the physical link integrity test in the switch and reported in his MIB) as can be seen in 3.

$$X = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \end{pmatrix} \qquad (3)$$

The algorithm calculates (or more precisely it has been offline precalculated as mentioned in II.B) the next hop (the new output port to reach the destination, i.e. device 1 in this example). If we apply the algorithm on this example, for the third switch and the device 1 (which is the destination of the flows of the device 13) with MAC address @MAC1 (lines 4-5 in Algorithm 1), we obtain:

line 6: (i,j') = (1,1)

line 7: the Bellman-Ford algorithm calculate the next hop to reach the switch j'=1 from the switch $j = 3$ according to the current graph G. The Bellman-Ford will have the choice between the ports 23, 24 as next hop because the matrix

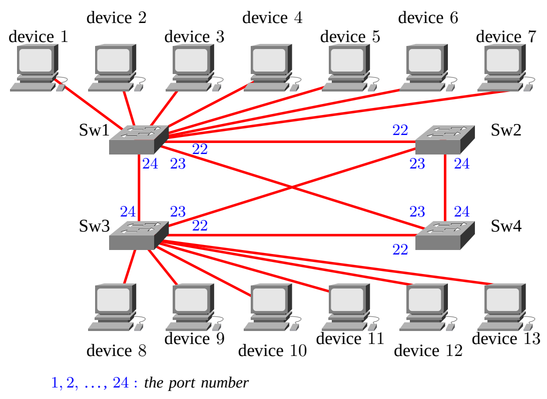

Figure 3. Architecture of case study

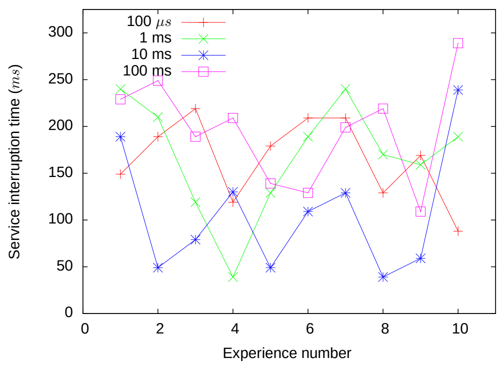

Figure 4. Service interruption time

TABLE I. EXPERIMENTAL TRAFFICS (BYTES, MS)

Device	Flow 1		Flow 2		Flow 3		Flow 4		Flow 5	
	length	period	length	period	length	period	length	period	length	period
2	72	12996	72	12996	-	-	-	-	-	-
3	72	288	72	72	72	144	72	144	72	1152
4	72	1152	72	1152	80	1152	72	1152	80	72
5	72	1152	94	72	72	12996	36	72	-	-
6	72	1620	72	1152	72	1152	72	1552	-	-
7	72	288	-	-	-	-	-	-	-	-
8	72	288	72	288	72	18	72	18	-	-
9	72	72	72	72	-	-	-	-	-	-
10	72	1152	72	1152	72	1152	72	1152	-	-
11	72	72	72	36	-	-	-	-	-	-
12	72	1152	72	72	72	72	72	36	72	1152
13	72	288	-	-	-	-	-	-	-	-

A gives the possible output ports according to the physical connectivity (i.e. ports {22,23,24}) and the matrix X gives the output ports states (i.e. port 21 = 0 so "up", port 23 = 1 so "up" and port 24 = 0 so "down"). Consider the following Bellman-Ford result: next hop = 23. As a result, the vector R_j equals to <@MAC1, 24> (before the failure) is updated to <@MAC1, 23>.

Consequently, the forwarding table of the switch $Sw3$ will be therefore modified as shown in Table III (the parameter $dot1dStaticAllowedToGoTo$ corresponding to the output port becomes 0x00 00 02 (i.e. the port 23). Notice that the forwarding table of the switch $Sw1$ will be also modified (by the algorithm) such a way that the messages of the devices $8, 9, \ldots, 13$ are sent on the output port 22. Moreover, in this example, any modifications will be done on the switch $Sw2$ because its forwarding table is initially configured to send the messages from the devices $8, 9, \ldots, 13$ to the devices $1, 2, \ldots, 7$ *via* the output port 22, and the messages from the devices $1, 2, \ldots, 7$ to the devices $8, 9, \ldots, 13$ *via* the output port 23 (which is the initial Bellman-Ford result).

Table IV shows the number of lost frames in the case of Sw1-Sw3 failure (the cable Sw1-Sw3 is unplugged) by using our solution (measures) or RSTP (the interruption time for RSTP has been considered as being of 1 s [15] for computing the losses). Any flows sent by the devices 2, 3, ..., 8 are lost. Indeed, the failure does not occur on their paths. For the other flows sent by the devices 9, 10, ... 13, the number of lost frames depends on the period of the flow and the interruption time. It is therefore possible to see that there is no loss for all flows with a period of 288 ms (vs. 4 by using RSTP). Only two frames have been lost for the flows with a period of 72 ms (vs. 14 by using RSTP). In the case of flows with a period of 18 ms, 14 frames are lost (vs. 56 by using RSTP). However, in the worst case, it could be observed 1 (respectively 3 and 15) loss(es) for the flows with a period of 288 ms (respectively 72 ms and 18 ms).

Fig. 5 and 6 show the evolution of the end to end delays. When a failure occurs, the reconfiguration involves a longest path (because the shortest path relatively to the number of hops had been set first). Consequently, the delay increases when the reconfiguration is effective as shown in the two graphs. In this example, the delay growth is equivalent to the switch latency (around only one transmission time because of switching mode: store & forward). However, this growth is proportionally limited (relatively to the service interruption time). Let us note that the zoom on each graph enables to estimate (and retrieve as presented in Table IV) the losses in regard to the interruption time due to the failure.

IV. CONCLUSION AND FUTURE WORK

This paper suggests a novel centralized approach for the physical (meshed) topology and the logical topology reconfiguration. Experimental measurements have shown that the the recovery time has been decreased and the number of losses is lower compared to RSTP (although it is still non-null especially for the flow with a small period).

Finally, the future work will integrate this solution on

TABLE II. INITIAL FORWARDING TABLE OF THE SWITCH SW3

dot1dStatic-Address	***dot1dStatic-ReceivePort***	***dot1dStatic-AllowedToGoTo***	***dot1dStatic-Status***
@MAC1	0	0x00 00 01	3
⋮	⋮	⋮	⋮
@MAC7	0	0x00 00 01	3
@MAC8	0	0x80	3
@MAC9	0	0x40	3
@MAC10	0	0x20	3
@MAC11	0	0x10	3
@MAC12	0	0x08	3
@MAC13	0	0x04	3

TABLE III. FORWARDING TABLE OF THE SWITCH SW3 AFTER RECONFIGURATION

dot1dStatic-Address	***dot1dStatic-ReceivePort***	***dot1dStatic-AllowedToGoTo***	***dot1dStatic-Status***
@MAC1	0	0x00 00 02	3
⋮	⋮	⋮	⋮
@MAC7	0	0x00 00 02	3
@MAC8	0	0x80	3
@MAC9	0	0x40	3
@MAC10	0	0x20	3
@MAC11	0	0x10	3
@MAC12	0	0x08	3
@MAC13	0	0x04	3

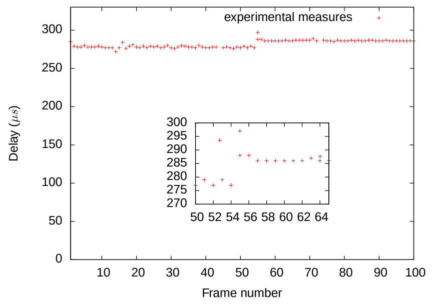

Figure 5. Delay of the flow 1 on device 13 (period: 288 ms)

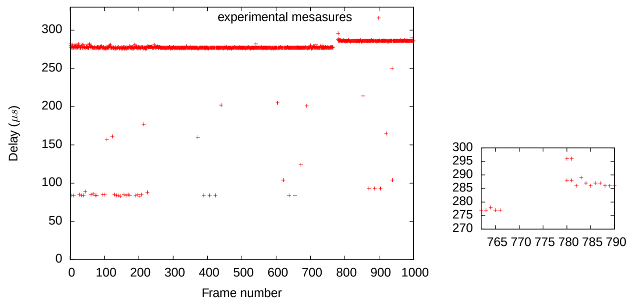

Figure 6. Delay of the flow 3 on device 8 (period: 18 ms)

a laboratory demonstrator (target TRL4) in order to study the impact of this active solution on others QoS parameters (e.g. the consumed bandwidth and its potential impact on the applicative end-to-end delays) and its robustness in embedded environment. Industrial switches will hence be used, which may very well decrease the detection time (and so the interruption time) if the "down state" is instantaneously notified (contrary to the values of "link_loss" in the link integrity test defined in the standard 802.3).

REFERENCES

[1] Department of Defense, "Military standard aircraft internal time division command/response multiplex data bus," MIL-STD-1553B, 1978.

[2] D. Monchaux, P. Gast, and J. Sangare, "Avionic-x : A demonstrator for the next generation launcher avionic," Embedded Real-Time Software and Systems (ERTS^2 12), Feb. 2012.

[3] A. Mifdaoui, F. Frances, and C. Fraboul, "Full duplex switched ethernet for next generation 1553B-based applications," Proc. IEEE Real Time and Embedded Technology and Applications Symposium (RTAS 07), Apr. 2007, pp. 45–56.

[4] J. Sommer, S. Gunreben, F. Feller, M. Kohn, A. Mifdaoui, D. Sass, and J. Scharf, "Ethernet–a survey on its fields of application," IEEE Communications Surveys & Tutorials, vol. 12, Apr. 2010, pp. 263–284.

[5] H. Zinner, J. Noebauer, T. Gallner, J. Seitz, and T. Waas, "Application and realization of gateways between conventional automotive and ip/ethernet-based networks," Proc. ACM Design Automation Conference (DAC 11), Jun. 2011, pp. 1–6.

TABLE IV. LOST FRAMES PER FLOW BY USING OUR SOLUTION (A) OR RSTP (B) IN THE CASE OF THE SW1-SW3 FAILURE

Device	Flow 1		Flow 2		Flow 3		Flow 4		Flow 5	
	A	B	A	B	A	B	A	B	A	B
2	0	0	0	0	-	-	-	-	-	-
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	-	-
6	0	0	0	0	0	0	0	0	-	-
7	0	0	-	-	-	-	-	-	-	-
8	0	4	0	4	14	56	14	56	-	-
9	2	14	2	14	-	-	-	-	-	-
10	0	0	0	0	0	0	0	0	-	-
11	2	14	8	28	-	-	-	-	-	-
12	0	0	2	14	2	14	8	28	0	0
13	0	4	-	-	-	-	-	-	-	-

[6] A. Mifdaoui, F. Frances, and C. Fraboul, "Centralized vs distributed communication scheme on Switched Ethernet for embedded military applications," Proc. IEEE Symposium on Industrial Embedded Systems (SIES 09), Jul. 2009, pp. 201–210.

[7] S. Vitturi, L. Peretti, L. Seno, M. Zigliotto, and C. Zunino, "Real-time ethernet networks for motion control," Computer Standards & Interfaces, vol. 33, Sep. 2011, pp. 465–476.

[8] J. Jasperneite, M. Schumacher, and K. Weber, "Limits of increasing the performance of industrial ethernet protocols," Proc. IEEE Emerging Technology and Factory Automation (ETFA 07), Sep. 2007, pp. 17 – 27.

[9] Grieu, "Analyse et évaluation de techniques de commutation ethernet pour l'interconnexion des systèmes avioniques," Ph.D. dissertation, Sep. 2004.

[10] J.-P. Georges, E. Rondeau, and T. Divoux, "Evaluation of switched ethernet in an industrial context by using the network calculus," Proc. IEEE Workshop on Factory Communication Systems (WFCS 02), Aug. 2002, pp. 19–26.

[11] J. Robert, J. Georges, T. Divoux, P. Miramont, B. Rmili *et al.*, "Ethernet networks for real-time systems: application to launchers," Proc. DAta Systems in Aerospace (DASIA 11), May 2011.

[12] T. Harima, "Introduction of iec activity on digital communication subsystem (overview of iec/sc65c)," Proc. IEEE ICCAS-SICE, Aug. 2009, pp. 1427–1431.

[13] G. Prytz, "Redundancy in industrial ethernet networks," Proc. IEEE Workshop on Factory Communication Systems (WFCS 06), Jun. 2006, pp. 380–385.

[14] M. Felser, "Media redundancy for profinet io," Proc. IEEE Workshop on Factory Communication Systems (WFCS 08), May 2008, pp. 325–330.

[15] M. Huynh, S. Goose, and P. Mohapatra, "Resilience technologies in ethernet," Computer Networks, vol. 54, Jan. 2010, pp. 57–78.

[16] IEEE Computer Society, "IEEE standards for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - common specifications - part 3: Media access control (mac) bridges," ANSI/IEEE Std 802.1D, Dec. 2004.

[17] J. C. Mankins, "Technology readiness levels," NASA white paper, vol. 6, Apr. 1995.

[18] IEEE Computer Society, "IEEE standard for information technology - telecommunications and information exchange between systems - local and metropolitan area networks - specific requirements - part 3: Carrier sense multiple access with collision detection (csma/cd) access method and physical layer specifications," IEEE Std 802.3, Dec. 2008.