processes

MDPI

Article

A Fault-Tolerant and a Reconfigurable Control Framework: Application to a Real Manufacturing System

Imane Tahiri [1,*], Alexandre Philippot [1], Véronique Carré-Ménétrier [1] and Abdelouahed Tajer [2]

[1] Research Center for Science and Information Technology and Communication, University of Reims Champagne Ardennes, 51100 Reims, France; alexandre.philippot@univ-reims.fr (A.P.); veronique.carre@univ-reims.fr (V.C.-M.)
[2] Systems and Applications Engineering Laboratory, University of Cadi Ayyad, Marrakech 40000, Morocco; a.tajer@uca.ac.ma
* Correspondence: imane.tahiri@univ-reims.fr

Abstract: In this paper, we propose a framework to implement a fault-tolerant and a reconfigurable distributed control approach in programmable logic controller (PLC) for manufacturing systems (MS). The reconfiguration methodology adopted in this paper is based on supervisory control theory (SCT), and it is triggered following sensor fault detection. The lost information about these sensors is replaced by timed information allowing the MS to continue its operations. The switch from a normal behavior to a degraded behavior when a sensor fault appears is ensured by reconfiguration rules. The main objective of our framework is to implement the obtained control into a PLC. To meet this objective, the distributed controllers of the two operating modes as well as the reconfiguration rules are interpreted into different Grafcet models. The implementation of these different models is verified by a checker-model technique before being tested on a digital twin and validated on a real MS.

Keywords: fault tolerant control; control reconfiguration; distributed control; manufacturing systems; digital twin; PLC implementation

Citation: Tahiri, I.; Philippot, A.; Carré-Ménétrier, V.; Tajer, A. A Fault-Tolerant and a Reconfigurable Control Framework: Application to a Real Manufacturing System. *Processes* **2022**, *10*, 1266. https://doi.org/10.3390/pr10071266

Academic Editor: Blaž Likozar

Received: 16 June 2022
Accepted: 24 June 2022
Published: 27 June 2022

1. Introduction

Over the past decades, the performance of manufacturing systems (MS) has been increasing significantly. The gain in performance has been accompanied by a complex increase of plants, resulting in a strong request for availability and security. However, the ability to perform the tasks for which the system has been designed can be hindered by the appearance of abnormal phenomena, such as faults. In this context, we propose a new approach: a distributed fault-tolerant and reconfigurable control for a programmable logical controller (PLC) implementation purpose.

A reconfiguration process can be triggered by events linked to products or to production resources. A change in production can be related to the nature of production or the product's quality or quantity. Usually, these changes may result in the addition and/or the removal of some material resources in the current production. In addition, the change of state of a production resource is characterized by two events: breakdowns and repairs. In case of a failure, the reconfiguration process should first seek to replace the failed resource with another, the aim in this context is to use hardware redundancies to replace the faulty element. The implementation of a reconfiguration process then depends on the trigger event and the time constraints that are applied in the MS following the occurrence of this event.

Two additional situations can be considered: (i) the launch of a new production when the system is in a shutdown situation and (ii) the occurrence of a failure in a system during operation [1]. The approach adopted in this paper deals within the second case.

Many solutions proposed in industry and research rely on hardware redundancy to recover the failure of a system component, a solution that can be expensive due to the

technology and the maintenance of MS components. To avoid this redundancy, we propose in this paper a distributed control approach which can be reconfigured, based on timed information of a Discrete Event System (DES) [2]. The idea is to design a reconfigurable control capable, in the case of a sensor fault detection, of adapting and exploiting the services still available and offered by the MS plant. The distributed control strategy used in this paper is inspired by the work proposed in [3].

The reconfiguration process consists of changing the current state of the system from the normal behavior controller to a target state belonging to the so-called degraded behavior controller (Figure 1) in order to maintain the operation of the system despite the faults (fault-tolerant control) which may hinder it. The information lost due to a faulty sensor is replaced with the information provided by timed estimators that describe the estimated operation of the sensor using training [4].

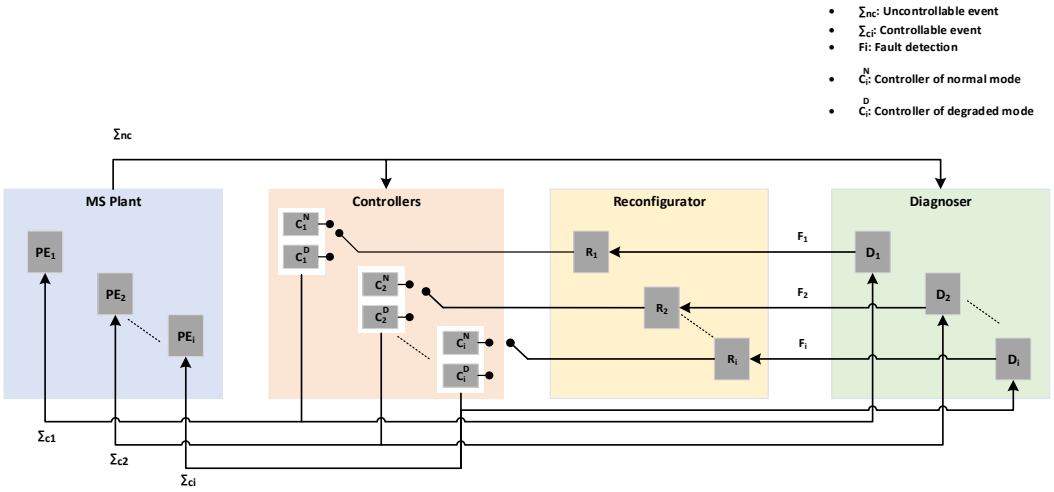


Figure 1. Distributed fault-tolerant and reconfigurable control principle.

In this work, we apply our method in the case of sensor fault detection. A sensor fault can be modeled as an unexpected passage of a sensor value from 0 to 1 (1 to 0) or a sensor stuck-off/on. Moreover, we provide a methodology starting from the design of a reconfigurable and fault-tolerant control to its implementation in a PLC for MS.

The overall view of the proposed approach is shown in Figure 2. The contribution is based on the following steps:

Step 1: The MS modeling is defined according to four steps: modeling of the normal behavior based on works presented [3], modeling of the degraded behavior based on timed information, modeling of liveness and security specifications, and modeling of reconfiguration specifications.

Step 2: The architecture design is based on supervisory control theory SCT founded on a distributed architecture to define two types of controllers (normal and degraded). The resulting controller models are interpreted into Grafcet according to translation rules, then the set of reconfiguration specifications is integrated to ensure the switch between the two modes. These reconfiguration specifications are also translated into Grafcet allowing to manage the two controllers Grafcets.

Step 3: Properties such as non-blocking and the reachability of the set of Grafcets resulting from the application of the reconfiguration approach are verified formally by a checker-model.

Step 4: If the verification phase is satisfying, all Grafcets are translated into a programming language and tested on a digital twin before being implemented in a real PLC. If the formal verification or simulation phase is not satisfying, the designer must make changes to the modeling of specifications and/or the plant.

Step 5: Part of our contribution is to implement the control in a virtual PLC and simulate it on the digital twin of the real system (software on the loop). A set of test

procedures (scenarios) is established to verify the good functioning of the digital twin. The simulated control is considered valid for implementation in the real PLC (hardware in the loop) or in the real system (real commissioning) only if all the established scenarios are executed correctly.

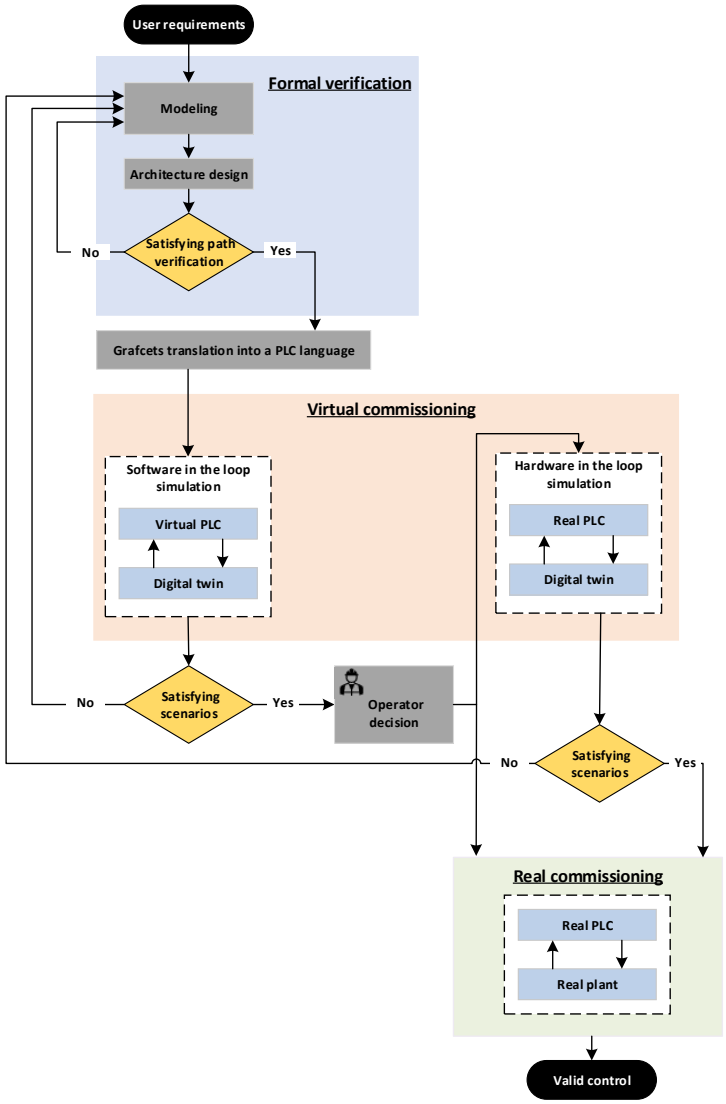


Figure 2. Architecture of the proposed approach.

This paper is organized as follows. In Section 2, the reconfigurable control elaboration principle as well as its formal verification is introduced. Section 3 presents the proposed methodology for virtual commissioning of tolerant and reconfigurable control. In Section 4, we apply our results around an MS. The contributions of this manuscript are discussed in Section 5. Finally, in Section 6, some indications of our future works are reported.

2. Principle of the Formal Verification

2.1. Reconfigurable Control Design

The proposed distributed control and the reconfiguration of MSs appears as an approach to manage the complexity of the control synthesis for large systems. In a previous work [5], we detailed the different steps of obtaining normal and degraded behavior controllers. However, in this article we present the basic steps of obtaining a reconfigurable and fault tolerant control as shown in Figure 3.

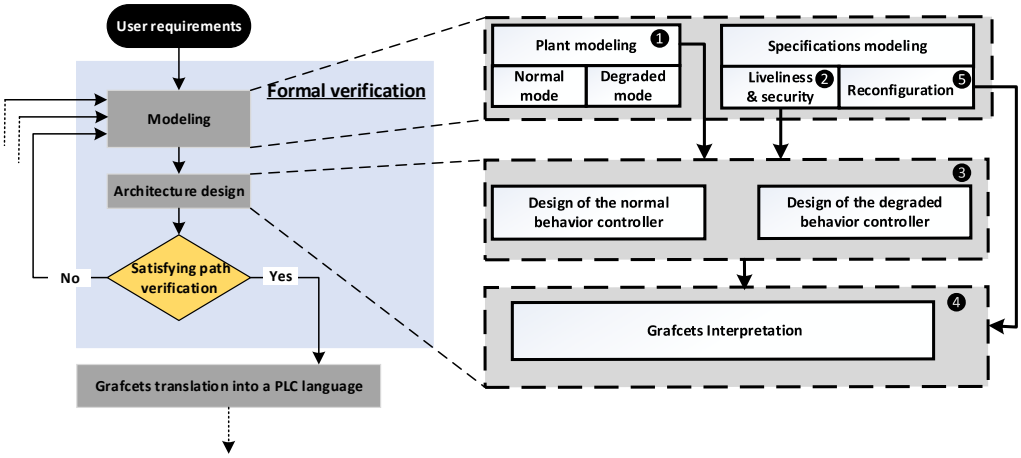


Figure 3. Steps of obtaining normal and degraded behavior controllers.

The modeling and the architecture design are based on the following five principal steps:

❶ Plant decomposition into several plant elements (i × PE) and then each PE is modeled according to the two behaving modes (normal and degraded). To create a degraded behavior mode controller during a reconfiguration request, the modeling of the plant normal behavior is extended to a timed modeling, where events are integrated to compensate the behavior of faulty elements;

❷ Modeling of two sets of system specifications: local and global safety (what the system must not do) and liveliness (what the system should do) specifications which are specific to each PE. The specifications modeling is based first on a Boolean logical equations representation, then translated into extended finite state automata, facilitating the control synthesis step and reducing the combinatorial explosion of the resulting models;

❸ Control synthesis: the framework of supervisory control theory (SCT) is adapted for synthesizing two types of controllers: one for normal behavior and the second one for degraded behavior in order to ensure a control redundancy to guarantee the continuity of operation when faults are detected;

❹ Distributed controllers models are interpreted into Grafcet (Standard IEC60848) [6] in order to be implemented in a PLC;

❺ Interpretation of a set of reconfiguration rules into the form of logical equations in order to ensure the switch between the two operating modes (normal mode—degraded mode) of each PE. These reconfiguration rules are also translated into Grafcet, making it possible to manage the two control Grafcets.

2.2. The Proposed Method for the Formal Verification of the Grafcets Models

The proposed distributed reconfiguration approach makes it possible to overcome the composition issues between models, which are the source of the combinatorial explosion of the state space. The proposed methodology implies for each PE: a normal behavior controller model, a degraded behavior controller model, and a reconfiguration Grafcet. However, there is a risk of Grafcets multiplication and therefore a risk of losing the overall vision of the specifications for the operator in the case of complex systems. Therefore, before any implementation, a verification of distributed controllers as well as the reconfiguration Grafcets established is necessary. This step consists of verifying firstly the deadlock property of the designed control and secondly the reachability property of the Grafcets of normal and degraded behaviors during a reconfiguration.

Since the Grafcets are obtained following an automatic procedure, their encoding is always assumed to be correct if the designer correctly follows the translation rules. Indeed, if the verification of a property is not satisfied, this implies that a fault was made during the modeling steps.

For this purpose, distributed controllers are verified using a verification model (checker-model) before implementation in a PLC. The set of distributed Grafcets of normal and degraded behaviors as well as the Grafcets ensuring reconfiguration are translated into structured text language (ST) [6] and verified using the UPPAAL software [7].

The implemented programs in PLCs are executed following a cycle consisting of four steps [8], as shown in Figure 4, apart from the initialization step during the first PLC cycle.

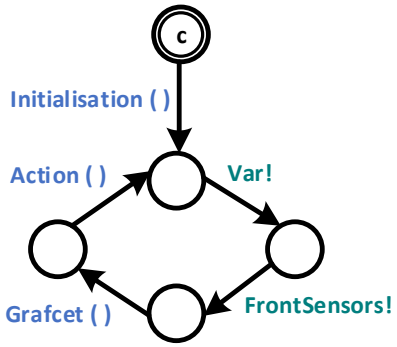


Figure 4. Modeling of a PLC cycle.

The four steps are presented as follows:

- Inputs reading (**var!**): Synchronization with the input models;
- Rising and falling edges of the sensors reading (**sensor edges!**): synchronization with the edge management models;
- Execution of the main program (call of the function **Grafcet ( )**);
- Update outputs (call of the function **action ( )**).

The PLC cycle model allows synchronization of all models in the system, while respecting the sequential execution order of the cyclic tasks of the PLC program. During the first PLC cycle, an initialization step is necessary to perform some operations, such as the initialization of programs, internal variables, and the state of system equipment. The synchronizations are broadcast type. The initialization is a priority task and is instantaneous. This translates into UPPAAL by the fact that the states concerned are labeled by C (for committed).

The model of Figure 5, synchronized with the PLC cycle model of Figure 4 through the message "var?", allows for randomly assigning true or false values to a sensor (represented by the argument "x" of the model) during each PLC cycle. This model is instantiated for each sensor observed by the system.

Figure 5. Generator model of sensors.

The modeling of the system elements is based on rising and falling edges. For this, a generator model of these edges is modeled, as shown in Figure 6. The change from x to 1 (x = 1) allows reading at 1 of the rising edge of x presented by (RE_x = 1), then the reading of the value of the rising edge changes to 0 (x is always at 1). Subsequently, x is deactivated, which is detected by the reading at 1 of the falling edge of x (FE_x = 1).

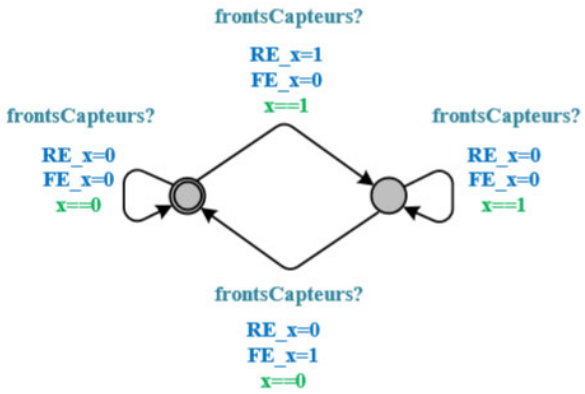


Figure 6. Generator model of sensors edges.

The ST modeling applied to the system's Grafcets is given by the following steps:

- Declaration of all the variables constituting the Grafcets (steps, sensors, action, durations, faults, etc.);
- Translation of Grafcets into self-holding equations [9]. To determine them, first of all, it is necessary to define the transition functions of all the obtained Grafcets, then the step activation equations, and finally, the different transition functions. The function of a transition "**ft**" is given by a conjunction of the state of the preceding steps and the logic condition associated with the transition. A step is activated if the function of one of the input transitions is true. It remains active as long as the output function is false. The actions associated with a step are active when the step is active [9];
- Assignment of actions to the corresponding states.

To check for possible Grafcets transcription errors in ST, a formal check can be applied in UPPAAL. Indeed, it is necessary to check that all the steps Xi of the designed Grafcets are reachable. However, this verification remains insufficient, only static test by proofreading the program can ensure that this transcription is not wrong.

The models obtained from the controllers as well as the reconfigurators cover the state space of the system, that is to say that all the desired evolutions of the system are already presented in the models translated into ST. Therefore, an exhaustive check of the deadlock property (is there an evolution leading to a deadlock?) is enough to demonstrate that the reconfigurable control is not blocking.

The overall control can only be translated into a PLC programming language (the choice of programming language is left to the user) if all requirements and properties are executed successfully. Otherwise, the obtained control presents errors that must be diagnosed and noted to correct them. Assuming the design of the control is correct, the error usually manifests in the plant and the specifications modeling.

3. Towards a Real Implementation of a Reconfigurable Control

Unlike real commissioning (RC) which requires the presence of the real machine and the real control system, the principle of virtual commissioning (VC) [10] is based on the connection between the control system and a plant simulator PS (virtual simulation model of the real machine). A PS is a model that is responsible for reproducing the behavior of the physical MS to be controlled, it is called a "digital Twin" (DT).

One of the first definitions of the digital twin is given by [11]: "A digital twin is a digital informational construction of a physical system seen as a single entity". In this context, the word "twin" implies that this digital information would be linked to the physical system throughout its operating cycle.

The concept of a DT of an MS refers to a 3D model of a complete MS with its resources, whether human or industrial. This digital representation can relate to a more or a less substantial set, such as a production line or an entire factory [12]. Applying this concept to manufacturing production allows manufacturers to create digital representations suitable

to their systems and production processes using collected data and information enabling analysis, decision making, and control to meet a defined goal (Figure 7).

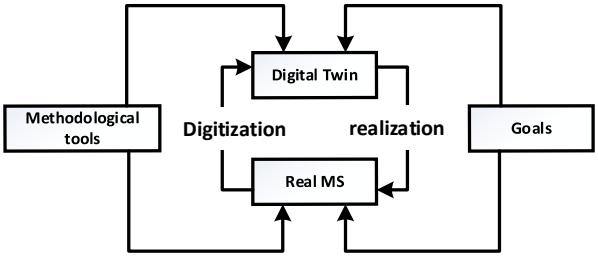


Figure 7. Digital twin concept.

The virtual commissioning previously presented in Figure 2 makes it possible to link a simulated MS (digital twin in our case) to a simulated PLC [13]. This configuration is used for checking the overall control, it is known as software-in-the-loop simulation (SIL Simulation), which means "simulation with software in the loop". It is a 100% virtual mode that allows the user to check only the software part of the control system.

A set of test procedures (scenarios) is established to verify the desired behavior of the digital twin. The obtained control following the framework evoked in Section 2.1 is simulated on the digital twin, and it is considered valid for implementation in the real system only if all established scenarios are executed correctly. The appearance of an unsatisfactory scenario implies that a fault was made during the modeling step. After the faults have been identified and corrected, all the previous steps are restored until the set of all scenarios of the digital twin is executed correctly. The control is then considered valid and implemented in a real PLC. Two cases are distinguished: (i) a hardware-in-the-loop simulation (HIL Simulation), based on the connection between the real PLC and the DT. The virtual commissioning here allows the verification of the obtained control through a simulation. Or, (ii) a real commissioning, the established scenarios are then tested again, but on the real MS; a satisfying execution of these scenarios means that the control is valid, and no changes are needed. Otherwise, the obtained control is wrong due to a modeling fault, and it must be corrected and rechecked. The DT realization is supposed to be correct, or a non-satisfying scenario can be linked to an error/gap during the realization of the DT, in this case a correction of the DT is necessary.

4. Application

4.1. Description of the Studied System

The previous sections have made it possible to describe step by step the different stages of the proposed approach to design an implementable, reconfigurable, and fault-tolerant control for MS. in this section, we propose to apply the proposed framework on a benchmark with a real system available within the CReSTIC laboratory at Reims Champagne Ardennes University (URCA), the advantage being that this benchmark has its digital twin that can be used for the verification step.

The URCA has five platforms which aim to optimize, promote, and provide the community with high-level technological expertise. These platforms are open to all URCA teams, academic teams outside of URCA, as well as the industrial community. Among these, "CellFlex4.0" is a training and research platform directly linked to the concepts of Industry 4.0. It includes (Figure 8) a multi-renewable energy platform supplying a flexible bottling and packaging workstation (CellFlex) with its digital twin (NXMCD), the whole is completed by software tools for simulation, emulation, and systems virtualization (Factory I/O and Home I/O, Emulate3D, CIROS).

Figure 8. CellFlex4.0 platform.

The flexible workstation (FW) of the CellFlex4.0 platform makes it possible to implement and to illustrate the different flows of information and products circulating within a modern industrial company. This production unit is oriented "packaging and conditioning," and it has hardware and software for control and industrial supervision. It comes from the ICOS axis of CPER 2007–2013, and it is currently part of the **FFCA** project (Factories of Future Champagne-Ardenne) of the CPER 2017–2022. In this paper, the workstation is used in part for the validation of proposals and its digital twin under NX-MCD from Siemens is used for the virtual commissioning.

The FW consists of filling bottles to package them in batches of six (called sixpack) to store them or to export them. It is distributed in six stations and one central conveyor, as illustrated in Figure 9. Functionally, the import/export station supplies six empty packs on the central conveyor, which connects each workstation. During this time, caps are routed to the bottling station, which has been supplied with a raw material mixture made by the mixing station. The bottling station makes it possible to have filled, corked, and traced bottles with RFID chips. Then, they are transported to the transfer box station which, when six bottles are available, places them in an empty sixpack.

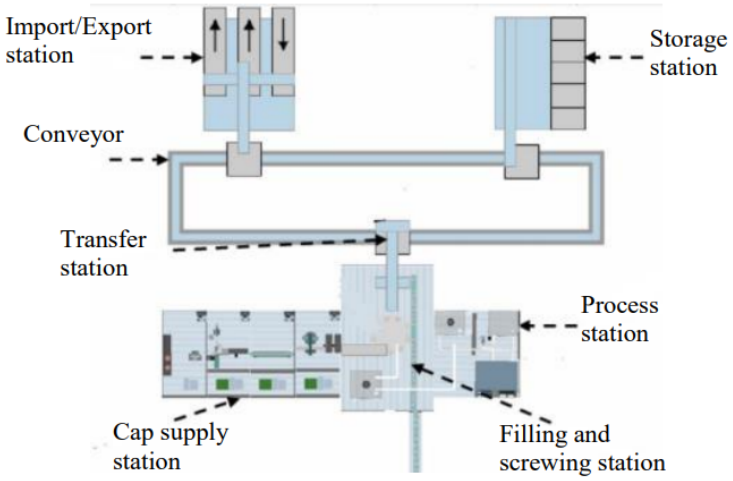


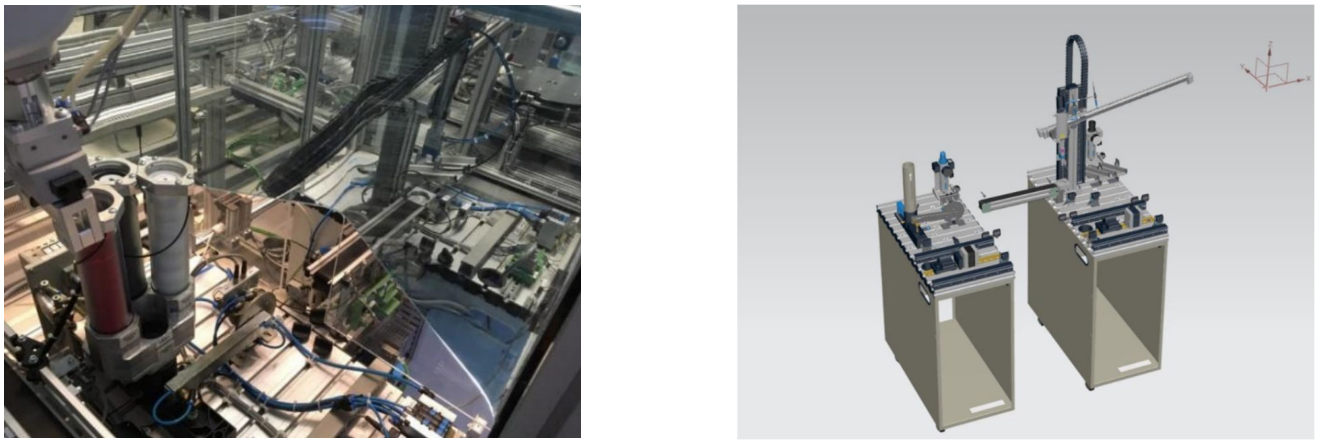
Figure 9. Workstation of the Cellflex 4.0.

This full sixpack is then either stored (storage station) or sent to the import/export station for recovery by the operator. The transfer station also has a vision system to identify packaging problems. The bottles are then transferred to a robotic cell in order to (i) remove the cap to put it back in the stock of the capping station, (ii) empty the bottle to reprocess the product at the mixing station, (iii) reinject the empty bottle in the bottling station.

The overall system is controlled via six industrial Siemens PLC/S7-1500 type (with or without remote I/O), and it has supervision interfaces allowing for global or local control of the FW.

In this paper, the reconfigurable control design approach is applied only to the capping station (CS) (Figure 10a). The workstation is a distributed system communicating between stations by Profinet. For example, the CS communicates with the bottling station via an exchange variable on the presence or not of bottles for removing a cap. The CS (Figure 9) is considered to be a distributed MS made up of 8 actuators controlled by different technologies and 15 constituent detectors:

- A store of caps in 3 colors (white, black, red);
- A device to evacuate them;
- A vacuum switch with aspiration and expiration for gripping the plugs;
- A buffer conveyor which allows the caps to be conveyed to the manipulator arm;
- A manipulator arm which, by means of a clamp, transfers the caps to the bottles at the bottling station.



(**a**) (**b**)

Figure 10. (**a**) Cellflex 4.0 capping station and its (**b**) digital twin.

The digital twin of this station can only represent the white cap store, so black and red cap processing will not be considered here. The inputs and the outputs of the PLC controlling the CS are shown in Tables 1 and 2, respectively.

Table 1. PLC outputs of the capping station.

Event	Actuator	Description
VBB	White plug cylinder	Retraction of the white plugs cylinder (monostable control)
EJ	Ejector	Exit of the ejection cylinder (monostable control)
VTAS	Suction cup	Air suction from the suction cup
VTEX		Air exhalation from the suction cup
VRM	Rotary cylinder	Moving the rotary cylinder to the magazine
VRC		Moving the rotary cylinder to the conveyor
CONV	Conveyor	Conveyor rotation
BMC	Manipulation arm	Moving the manipulation arm to the conveyor
BME		Moving the handling arm to the bottling
PINCES	Pliers	Opening of the grippers (monostable control)
DVL	Lifting cylinder	Lowering of the lifting cylinder (monostable control)

Table 2. PLC inputs of the capping station.

Sensor	Description
c_vbb	White plugs cylinder retracted
pm	Empty store
cer	Ejection cylinder retracted
ces	Ejection cylinder extended
c_vrc	Rotary cylinder on the conveyor
c_vrm	Rotary cylinder on the magazine
c_vt	Suction cup under pressure
dconv	Presence of a stopper at the start of the conveyor (in reverse logic)
fconv	Presence of a plug at the end of the conveyor (in reverse logic)
c_bmc	Conveyor side handling arm
c_bme	Handling arm on the bottling side
vlb	Lift cylinder in low position
vlh	Lift cylinder in high position
recept	Presence of a cap on the receptacle at the end of the conveyor
bp	Information de la station d'embouteillage sur la présence d'une bouteille

4.2. System Modeling

The plant of the capping station consists of eight actuators:

- A single-acting cylinder (white plug cylinder) having a limit sensor active when the cylinder is retracted. To drop a cap, the cylinder must be retracted completely, released, then deactivated when the cylinder sensor indicates that the cylinder is retracted;
- A single-acting cylinder (ejector) surrounded by three sensors: two limit sensors (retracted position and extended position) and an infrared sensor which detects the presence of a cap in front of this cylinder;
- A double-acting cylinder (rotating arm) with two arm presence detectors (magazine side and conveyor side);
- An actuator for the aspiration and expiration of air (suction cup) with a pressure sensor to detect the sucked plugs;

- A belt conveyor, operated by a motor with one direction of rotation associated with two limit presence sensors at the extremities of the conveyor and a limit presence sensor on the receptacle;
- A double-acting cylinder (handling arm) with two position sensors in the conveyor zone and in the CS zone;
- A single-acting cylinder (lifting arm) with two high and low position sensors;
- A clamp that is not associated with any sensor.

The models of the different plant elements are obtained by the design of their models, resulting from a synchronous composition of the models of the detectors and their corresponding actuator models.

In this application, we integrate a fault in the limit switch sensor in the extended position (ces). We associate the activation of this sensor (↑ces) with a ck_1 clock and its deactivation (↓ces) with a ck_2 clock. The model of this plant element then becomes a timed model.

By applying local and global synthesis, we obtain distributed controllers for both normal and degraded behavior. Then, interpretation rules of an automaton to a Grafcet are applied to the distributed controllers of the CS previously studied; we obtain the control Grafcets of the normal behaviors presented in Figure 11 and the degraded behavior of the ejector shown in Figure 12. For more details, the methodology for obtaining these different models was presented in a previous work [5].

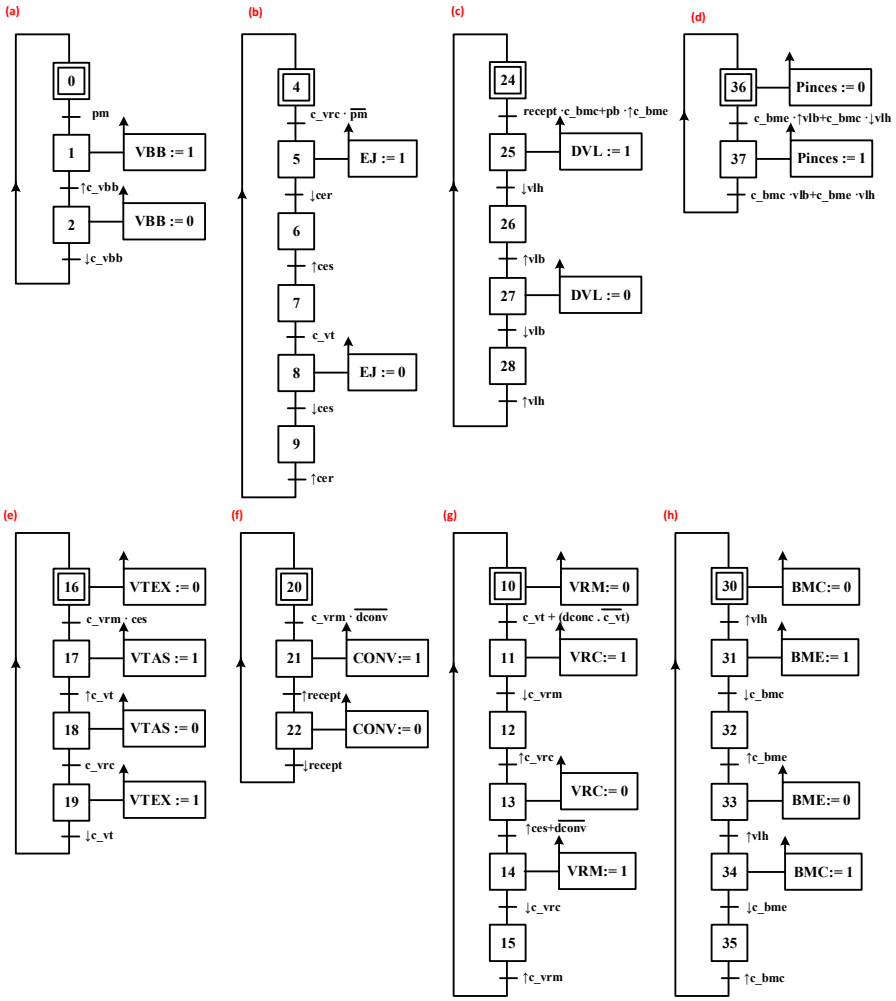


Figure 11. Grafcets for controlling the normal behavior of the different PEs: (**a**) white plug cylinder, (**b**) ejector, (**c**) lifting arm, (**d**) gripper, (**e**) suction cup, (**f**) conveyor, (**g**) rotary cylinder, (**h**) manipulation arm.

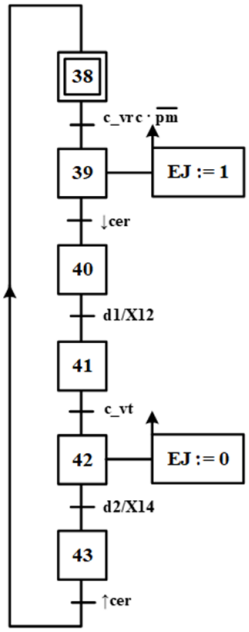


Figure 12. Grafcet for controlling the degraded behavior of the ejector.

4.3. Modeling of the Reconfigurator

The faulty element of this station is the "**ces**" sensor associated with the ejector. To ensure the reconfiguration of this actuator upon detection of the fault "**fs**", a reconfiguration rule must be determined to guarantee switching between the Grafcets of normal $G^{N(EJ)}$ and degraded $G^{F(EJ)}$ behaviors.

The change from $G^{N(EJ)}$ to $G^{F(EJ)}$ and vice versa is based on the following two equations:

$$\begin{aligned} &RC_1\text{: } \textbf{If } X_6 \textbf{ and } f_{s;} = 1 \textbf{ Then} \\ &(F\text{: } G^{F(EJ)}\{X_{40}\}) \textbf{ and } (F\text{: } G^{N(EJ)}\{\}) \\ &\textbf{Else If } X_{40} \textbf{ and } f_s = 0 \textbf{ Then} \\ &(F\text{: } G^{N(EJ)}\{X_6\}) \textbf{ and } (F\text{: } G^{F(EJ)}\{\}) \end{aligned} \quad (1)$$

$$\begin{aligned} &RC_2\text{: } \textbf{If } X_8 \textbf{ and } f_s = 1 \textbf{ Then} \\ &(F\text{: } G^{F(EJ)}\{X_{42}\}) \textbf{ and } (F\text{: } G^{N(EJ)}\{\}) \\ &\textbf{Else If } X_{42} \textbf{ and } f_s = 0 \textbf{ Then} \\ &(F\text{: } G^{N(EJ)}\{X_8\}) \textbf{ and } (F\text{: } G^{F(EJ)}\{\}) \end{aligned} \quad (2)$$

The translation of the two equations into Grafcet makes it possible to obtain the reconfiguration model of the ejector presented by the Grafcet in Figure 13.

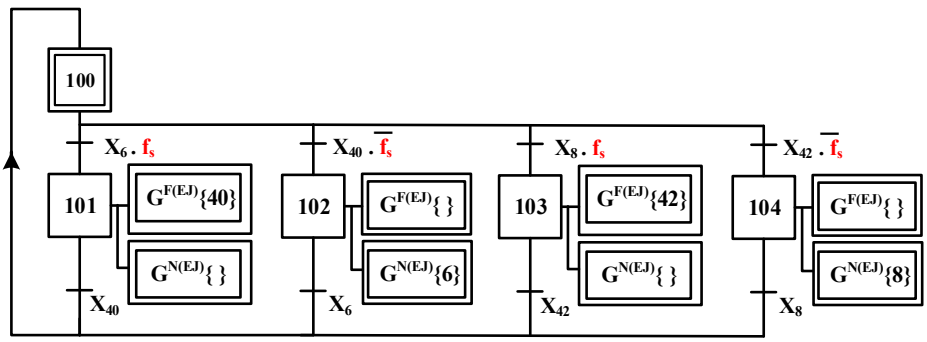


Figure 13. Grafcet interpretation of the reconfiguration constraints CR1 and CR2 (Reconfigurator model).

4.4. Verification Models and Programs

Before any PLC implementation of Grafcets, a set of properties expressed in the form of specifications and conditions must be formally verified.

The first specification to check is the **deadlock** property of the distributed Grafcets of the control as well as the Grafcet of the reconfiguration of the CS. This is ensured by checking the condition "Is there a path leading to a deadlock?" This condition is expressed in the UPPAAL checker by the following proposition:

- **Condition 1:** E < > deadlock.

If the result of this check is satisfied, it means that no deadlock is found and that the control Grafcets are not blocking. It is then possible to check other properties.

The second specification consists in verifying the **reachability** property of the requested Grafcet after reconfiguration. This is ensured by checking the condition "is there a path where the step associated with the reconfiguration request is active, and the Grafcet step requested after reconfiguration is deactivated?" This condition is checked for all the steps where a reconfiguration is triggered. It is expressed in UPPAAL by the following four conditions:

- **Condition 2:** [E < > X101 and not X40];
- **Condition 3:** [E < > X102 and not X6];
- **Condition 4:** [E < > X103 and not X42];
- **Condition 5:** [E < > X104 and not X8].

After a reconfiguration request, it is necessary to verify that only one Grafcet is functional for the ejector. The existence of a case where two steps are active simultaneously in two different Grafcets (third specification) must be verified. This condition is expressed by:

- **Condition 6:** [E < > (X4 or X5 or X6 or X7 or X8 or X9) and (X38 or X39 or X40 or X41 or X42 or X43)].

The result of the formal verification tests of all the Grafcets is presented in the following table (Table 3). The verification of each condition is carried out in an average time of 25s.

Table 3. Results of the formal verification of all the Grafcets of the corking station.

Checked Condition	Check Result
Condition 1	Satisfying
Condition 2	Satisfying
Condition 3	Satisfying
Condition 4	Satisfying
Condition 5	Satisfying
Condition 6	Satisfying

A satisfying verification of all these conditions allows a passage to a second simulation test on the digital twin before final implementation on the real system.

4.5. Towards a Control Implementation on the Capping Station

The digital twin of the CS (Figure 10b) works by collecting a combination of data. This tool allows us to simulate the station equipment and the production line in more detail by including its kinematics. It then becomes possible to validate the working conditions of the operators or even to operate the PLCs in real conditions and to detect interferences, for example. The objective behind the verification of the obtained control via the digital twin is the realization of the tests and the operating scenarios in order to modify the PLC programs if scenario feedback is false, while the real equipment is stopped.

After a formal verification of the previous properties, it is appropriate either to translate the Grafcet specification models into PLC programming language or to retrieve the ST

program established during the formal verification on UPPAAL. The program is then implemented in the Siemens PLC simulator PLCSIM-ADVANCED in order to control the digital twin under NXMCD.

By testing that all the simulated scenarios correspond to the desired operations (normal and degraded), the control models are then implemented in the real PLC (S7-1500) for controlling the CS.

An unsatisfactory execution of a scenario implies a return to the step of modeling the system and the specifications. For the real station, an unsatisfactory scenario may also be due to a functional problem of the digital twin. This time around, the digital twin needs to be fixed.

Among the scenarios tested we identified the following two scenarios (Figure 14), which allowed us to modify some specifications:

- **Scenario 1**: When the magazine is emptied, the retraction of the white plug cylinder is expected to supply the magazine with a plug.
- **Scenario 2**: The handling arm is positioned on the bottling side with a cap tightened by the clamp and the lifting cylinder is controlled to lower and to raise. The clamp must open once the lower position of the lifting cylinder is reached.

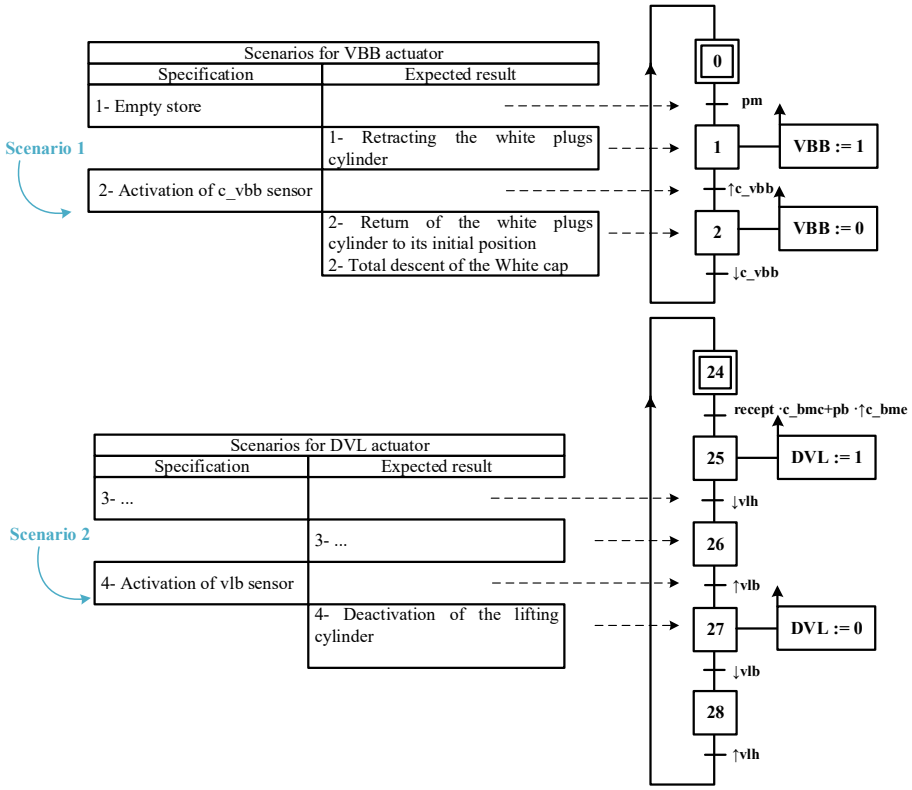


Figure 14. Extract scenarios to verify.

A first test on the digital twin (Table 4) allowed us to identify an error when defining the overall specifications of the lifting cylinder (scenario 2). Raising the cylinder after reaching its low position does not allow the pliers the time needed to loosen the plug. To this, we have added a timer as a specification on VDL inhibition. After the correction of this specification, a second test on the digital twin was launched, the result this time is correct for both scenarios.

Table 4. Results of scenarios before and after correction of specifications.

Scenario	Simulation Result	Simulation Result after Correction n°1	Result of the Application on the Station	Application Result on the Station after Correction n°2
Scenario 1	Correct	Correct	False	Correct
Scenario 2	False	Correct	Correct	Correct

The first attempt at the station turned out to be unsuccessful. We have identified an error in scenario one, which was correct during the simulation on the digital twin. The error that occurred at the cap cylinder is related to the reaction time difference between the real system and the digital twin. The time taken by the cylinder to make its round trip does not allow the total descent of the cap, for this we have added a specification on the deactivation of VBB. A time delay is necessary for the descent of the cap. After correcting the specification in question and a second test on the station, the result is correct.

These corrections allowed us to obtain two new control Grafcets for the two actuators VBB and VDL shown respectively in Figure 15a,b.

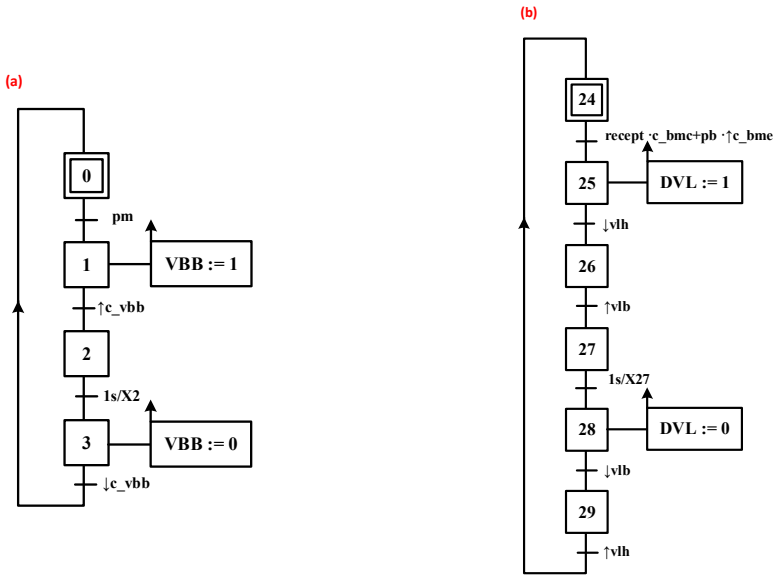


Figure 15. Control Grafcets of VBB (**a**) and VDL (**b**) actuators after specifications correction.

5. Discussion

Manufacturing systems (MS) represent an important class of industrial systems that have become complex and susceptible to malfunctions with important consequences for the productivity, the production quality, and the safety of goods and people. Nowadays, it is a big challenge to implement formal control design approaches in order to guarantee operational safety. Accordingly, the work presented in this paper focuses on the implementation of a reconfigurable and fault tolerant control for MS controlled by PLC.

Most of the proposed approaches in the literature for control reconfiguration have been developed for systems where the information used is centralized [14]. When the MS to be studied is large, the design of reconfigurable controllers, based on centralized synthesis methods derived from the supervisory control theory (SCT), turns out to be a complex task [15]. The solution to ensure the reconfiguration of MS is very often based on the use of hardware redundancies, a solution that can be expensive for companies needing to reconfigure several faulty elements [1].

In the context of SCT, taking into account faulty events as well as events associated with reconfiguration for centralized approaches leads not only to a state space explosion but also to a model interpretation problem [16]. Moreover, the literature shows that there are few

methodologies for the implementation of reconfigurable controllers and more specifically in a PLC [17,18]. To address these issues, a framework for flexible, reconfigurable, and implementable controllers is developed in this paper.

The main idea of the approach is to ensure the continuity of the services of MS in case of a sensor fault detection that can hinder its normal behavior. To achieve this objective, two operating modes are necessary (the normal mode and the degraded mode (taking into account faults). The first behavior is controlled using a distributed control synthesis. While the second behavior is controlled by an extended approach where timed information is considered. First, the local controllers are designed by a synthesis between the models of the plant elements and the specifications corresponding to each desired behavior. A specification is defined as a logical equation. Then, global specifications are defined to ensure the overall behavior where all distributed controllers can communicate with each other. Once the distributed controllers are obtained, they are interpreted into Grafcet specification for a PLC implementation purpose. Instead of defining a single Grafcet containing the two operating modes and switching between the two of them by a simple "OR" in the receptivity, we opted for a separation of the operating modes and the reconfigurator (switcher). To switch from Grafcet models of normal behavior to Grafcet models of degraded behavior, reconfiguration constraints are also defined and interpreted into a Grafcet. This framework allows a clear view to the operator of the active operating mode.

The obtained control is operationally safe by construction, and it ensures the control of the two behaviors of an MS but it can be blocking. To overcome this problem, we opted for formal checking tests of the Grafcets corresponding to the different distributed controllers before the implementation. Our approach implies a strong solicitation of the reconfiguration Grafcets in case of fault detection but also in case of normal restarting of the system. These Grafcets represent a "switch" between the operating modes. It is therefore necessary to ensure that the overall control is not leading to a deadlock, and that each step of the Grafcets is reachable when a reconfiguration is requested.

Before implementing all these Grafcets, a methodology for verifying these properties using UPPAAL software is proposed. First, the deadlock of all controllers and reconfigurators is verified. Then, a check of the reachability of the Grafcets after a reconfiguration is necessary. If the Grafcet is indeed reached in the corresponding step, an additional verification is tested. It consists of checking whether two Grafcets operating in normal and degraded modes of the same plant element are activated at the same time, which is not allowed, because reconfiguration allows switching from the normal Grafcet to the degraded, while deactivating the first and vice versa.

The proposed reconfigurable control design approach is applied to an MS existing in our research laboratory (CReSTIC). First, the studied system was introduced and described. We then focused on a single station ensuring the plugging; the different plant elements are modeled taking into account a sensor fault on a limit switch belonging to the ejector of the station. Then, the local controllers of the normal and the degraded behaviors of these plant elements are determined through a synthesis of the control based on the SCT. By integrating all the global specifications, we establish the models of the distributed controllers, which we then translate into Grafcet for an implementation purpose. Subsequently, the reconfiguration specifications are defined and interpreted also into Grafcet. Before the PLC implementation of the set of the obtained Grafcets, they are checked on a UPPAAL checker to ensure that the designed control is not blocking and the reconfiguration request is still achieved. Finally, the Grafcets are implemented and applied on the digital twin and then on the real system.

6. Conclusions and Future Works

The main advantage of our approach is the use of a reconfiguration mechanism based on flexibility to evaluate different control objectives in order to achieve efficiency and reactivity. To ensure this flexible control, our method is founded on the distributed control

architectures of MS and on the determination of a set of reconfiguration constraints that allow switching between the different controllers in high flexibility and without any system downtime. Based on a distributed control architecture, the combinatorial explosion of the state space recurrent in centralized control approaches as [19,20] is avoided while facilitating the modeling steps of the plant as well as the specifications to be respected. In addition, it allows the reconfiguration of the single faulty plant element without reconfiguring the entire control system [20]. In addition, replacing the events associated with faulty sensors with timed events avoids the use of redundant elements. In addition, the overall behavior is maintained even if a sensor fault appears. Indeed, the faulty sensor in a global constraint is replaced by its corresponding timed information, which ensures the continuity of communication between the different distributed controllers of the degraded mode. Additionally, a method of implementing our contribution is proposed for an MS, to show its relevance in terms of usability.

While the approach presents several advantages, it has some drawbacks related, for example, to:

- Plant modeling (complexity in establishing the models corresponding to each plant element);
- Specifications modeling (how to ensure that they are sufficient and non-blocking?);
- The optimization of the models generated by the approach adopted for the design of the reconfigurable control;
- The lack of a software tool for the approach automation and the PLC memory allocation.

These advantages and drawbacks of the proposed work make it possible to identify several perspectives that constitute interesting and promising avenues for future research. These tracks relate to:

- Modeling: Designing models of normal and degraded behaviors can be tedious. The development of a library of proven models, linked to the different plant elements and local controllers for different MS equipment, may be a response to modeling difficulty. It would then be up to the operator to define the global constraints to determine the distributed controllers. Therefore, the modeling phase would become less complex. By choosing the equipment, the library could offer the control designer the two models of controllers corresponding to its two normal and degraded operations;
- Optimization: The resulting control Grafcets are automatically obtained from the distributed controllers interpretation step. However, the optimization of this interpretation seems possible in the context where some generated steps can sometimes be deleted. Indeed, the obtained control contains passages of the sensors from a reading of “0” to “1,” which do not influence the operation of the system and which can be deleted to reduce the resulting control Grafcets sizes;
- PLC memory allocation: The implementation of the obtained Grafcets for two operating modes (normal and degraded), and the implementation of the reconfiguration require a very large PLC memory. This is no longer an issue since PLCs have evolved more than a few years ago. However, the implementation of all Grafcets shows that 2/3 of the steps are redundant. Therefore, the implemented program in the PLC is ultimately doubled as a line of code. The perspective at this level is to study the impact of all additional lines on PLC cycle time and what can affect the production time;
- Automation of the approach: The proposed contribution in this work is based on the designer who intervenes at the modeling of the plant elements as well as the local and global specifications. The checks carried out first by UPPAAL before implementation and then by the digital twin after implementation allow for testing and correcting specifications if necessary. One possible perspective is to propose a software tool to help the designer to develop all the specifications. Indeed, similar to the engineering approaches of safe operating systems based on models [21], high-level conceptual models must be able to allow automatic interpretation of specifications.

Author Contributions: Conceptualization, I.T. and writing—original draft preparation, I.T.; writing—review and editing, A.P., V.C.-M., A.T. All authors have read and agreed to the published version of the manuscript.

Funding: The proposed work in this paper is developed as part of the FFCA project, funded under the CPER 2014–2020 named PFEXCEL.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Deschamps, E. Diagnostic de Services pour la Reconfiguration Dynamique de Systèmes à Evénements Discrets Complexes. Ph.D. Thesis, Institut National Polytechnique de Grenoble—INPG, Grenoble, France, 2007. Available online: https://tel.archives-ouvertes.fr/tel-00196462/document (accessed on 12 March 2019).
2. Ramadge, P.J.G.; Wonham, W.M. The control of discrete event systems. *Proc. IEEE* **1989**, *77*, 81–98. [CrossRef]
3. Qamsane, Y.; Tajer, A.; Philippot, A. A synthesis approach to distributed supervisory control design for manufacturing systems with Grafcet implementation. *Int. J. Prod. Res.* **2016**, *55*, 4283–4303. [CrossRef]
4. Tahiri, I.; Philippot, A.; Carré-Ménétrier, V.; Tajer, A. Time-Based Estimator for Control Reconfiguration of Discrete Event Systems (DES). In Proceedings of the 6th International Conference on Control, Decision and Information Technologies CoDIT'19, Paris, France, 23–26 April 2019.
5. Tahiri, I.; Parant, A.; Gellot, F.; Philippot, A.; Carre-Menetrier, V. Design and application of a reconfigurable control to a cyber-physical system. In Proceedings of the 17th International Conference on Informatics in Control, Automation and Robotics (ICINCO 2020), Paris, France, 7–9 July 2020.
6. *IEC 61131-3:2013*; Programmable Controllers—Part 3: Programming Languages. iTeh Standards Store: Newark, DE, USA, 2013. Available online: https://standards.iteh.ai/catalog/standards/iec/d44024f3-d345-4126-a5f8-21442d41d3bc/iec-61131-3-2013 (accessed on 30 June 2021).
7. UPPAAL. Available online: http://www.uppaal.org/ (accessed on 7 February 2020).
8. Niang, M. Vérification Formelle et Simulation pour la Validation du Système de Contrôle Commande des EALE (Équipements d'Alimentation des Lignes Électrifiées). Ph.D. Thesis, The Digital and Engineering Sciences Doctoral School, Reims, France, 2018. Available online: http://www.theses.fr/2018REIMS021 (accessed on 23 July 2019).
9. Machado, J.J.B.; Denis, B.; Lesage, J.-J.; Faure, J.-M.; Fereira, J. Logic Controllers Dependability Verification Using a Plant Model. In Proceedings of the 3rd IFAC Workshop on Discrete-Event System Design, DESDes'06, Rydzyna, Poland, 26–28 September 2006; pp. 37–42. Available online: https://hal.archives-ouvertes.fr/hal-00361815 (accessed on 18 June 2020).
10. Drath, R.; Weber, P.; Mauser, N. An evolutionary approach for the industrial introduction of virtual commissioning. In Proceedings of the 2008 IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg, Germany, 15–18 September 2008. [CrossRef]
11. Grieves, M.W.; Vickers, J. Digital Twin: Mitigating Unpredictable, Undesirable Emergent Behavior in Complex Systems. In *Transdisciplinary Perspectives on Complex Systems*; Springer: Cham, Switzerland, 2017. [CrossRef]
12. Shao, G.; Helu, M. Framework for a digital twin in manufacturing: Scope and requirements. *Manuf. Lett.* **2020**, *24*, 105–107. [CrossRef]
13. Lee, C.G.; Park, S.C. Survey on the virtual commissioning of manufacturing systems. *J. Comput. Des. Eng.* **2014**, *1*, 213–222. [CrossRef]
14. Cho, K.-H.; Lim, J.-T. Mixed centralized/decentralized supervisory control of discrete event dynamic systems. *Automatica* **1999**, *35*, 121–128. [CrossRef]
15. Kim, D.Y.; Park, J.W.; Baek, S.; Park, K.B.; Kim, H.R.; Park, J.I.; Kim, H.S.; Kim, B.B.; Oh, H.Y.; Namgung, K.; et al. A modular factory testbed for the rapid reconfiguration of manufacturing systems. *J. Intell. Manuf.* **2020**, *31*, 661–680. [CrossRef]
16. Zaytoon, J.; Riera, B. Synthesis and implementation of logic controllers—A review. *Annu. Rev. Control* **2017**, *43*, 152–168. [CrossRef]
17. Fabian, M.; Hellgren, A. PLC-based implementation of supervisory control for discrete event systems. In Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171), Tampa, FL, USA, 18 December 1998; Volume 3, pp. 3305–3310. [CrossRef]
18. Macktoobian, M.; Wonham, W.M. Automatic reconfiguration of untimed discrete-event systems. In Proceedings of the 2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), Mexico City, Mexico, 20–22 October 2017; pp. 1–6. [CrossRef]
19. Faraut, G.; Piétrac, L.; Niel, E. Control law synthesis and reconfiguration using SCT. In Proceedings of the 2010 Conference on Control and Fault-Tolerant Systems (SysTol), Nice, France, 6–8 October 2010; pp. 576–581. [CrossRef]
20. Kumar, R.; Takai, S. A Framework for Control-Reconfiguration Following Fault-Detection in Discrete Event Systems. *IFAC Proc. Vol.* **2012**, *45*, 848–853. [CrossRef]
21. Bévan, R.; Berruet, P.; de Lamotte, F.; Adam, M.; Cardin, O.; Castagna, P. Generation of multiplatform control for transitic systems using a component-based approach. In Proceedings of the 2012 IEEE 17th International Conference on Emerging Technologies Factory Automation (ETFA 2012), Krakow, Poland, 17–21 September 2012; pp. 1–8. [CrossRef]