

## Article

# Integrating Modelling of Maintenance Policies within a Stochastic Hybrid Automaton Framework of Dynamic Reliability

Simone Arena <sup>1,\*</sup>, Irene Roda <sup>2</sup> and Ferdinando Chiacchio <sup>3</sup>

<sup>1</sup> Department of Mechanical, Chemical and Materials Engineering, University of Cagliari, Via Marengo 2, 09123 Cagliari, Italy

<sup>2</sup> Department of Management, Economics and Industrial Engineering, Politecnico di Milano, Via Lambruschini 4/b, 20156 Milano, Italy; irene.roda@polimi.it

<sup>3</sup> Department of Electrical, Electronic and Computer Engineering, University of Catania, Viale Andrea Doria 6, 95025 Catania, Italy; chiacchio@dmi.unict.it

\* Correspondence: simonearena@unica.it

**Abstract:** The dependability assessment is a crucial activity for determining the availability, safety and maintainability of a system and establishing the best mitigation measures to prevent serious flaws and process interruptions. One of the most promising methodologies for the analysis of complex systems is Dynamic Reliability (also known as DPRA) with models that define explicitly the interactions between components and variables. Among the mathematical techniques of DPRA, Stochastic Hybrid Automaton (SHA) has been used to model systems characterized by continuous and discrete variables. Recently, a DPRA-oriented SHA modelling formalism, known as Stochastic Hybrid Fault Tree Automaton (SHyFTA), has been formalized together with a software library (SHyFTOO) that simplifies the resolution of complex models. At the state of the art, SHyFTOO allows analyzing the dependability of multistate repairable systems characterized by a reactive maintenance policy. Exploiting the flexibility of SHyFTA, this paper aims to extend the tools' functionalities to other well-known maintenance policies. To achieve this goal, the main features of the preventive, risk-based and condition-based maintenance policies will be analyzed and used to design a software model to integrate into the SHyFTOO. Finally, a case study to test and compare the results of the different maintenance policies will be illustrated.

**Keywords:** Monte Carlo simulation; dynamic fault trees; multistate systems; SHyFTOO; matlab



**Citation:** Arena, S.; Roda, I.; Chiacchio, F. Integrating Modelling of Maintenance Policies within a Stochastic Hybrid Automaton Framework of Dynamic Reliability. *Appl. Sci.* **2021**, *11*, 2300. <https://doi.org/10.3390/app11052300>

Academic Editor: Suk Joo Bae

Received: 19 February 2021

Accepted: 28 February 2021

Published: 5 March 2021

**Publisher's Note:** MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



**Copyright:** © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

## 1. Introduction

In recent years, the concept of Industry 4.0 has gained interest worldwide, leading many manufacturers and organizations towards a digital transition. This transition is based on the idea of intelligent production, which refers to smart and agile manufacturing and smart factories by using technological levers such as Artificial Intelligence (AI), the Internet of Things (IoT) and Cyberphysical System (CPS) [1]. This innovative approach leads companies to cope with the challenges of a much more dynamic environment where market demands require both more flexible solutions to meet high levels of product customization and production profitability through intelligent process control and management [2]. The availability of these new technologies has driven the manufacturing environment towards an increase in connectivity and interaction among systems, humans, and machines that allows for the integration of different automated or semi-automated systems. This integration represents the core idea of the evolution of real physical systems into high-level cyber technologies. However, the adoption of Industry 4.0 and IoT paradigms has led to an increasing complexity and dynamics of processes and products. In particular, the complexity of systems consisting of a large number of interconnected technological elements, such as subsystems and components working in dynamic operational environments, is one of the main challenges and requires a good

engineering approach, optimized operations and proper maintenance to keep the system in an optimal state [3]. To this end, maintenance planning plays a key role to respond to these continued advances in the dynamicity and complexity of the manufacturing environment, leading to an increasing interest in maintenance decision-making methods and algorithms [4,5]. In this context, the concepts of the reliability and maintainability of critical engineering assets are essential aspects of modern industry. Indeed, the need to ensure proper operation and to avoid the failure of equipment [6], is closely related to supporting companies in the maintenance decision-making process, aiming at both improving productivity and reducing maintenance costs. To this end, the dependability and the risk assessment are developed. The former is used for determining the availability, safety and maintainability of a system and establishing the best mitigation measures to prevent serious flaws and process interruptions. The latter is able to systematically identify, analyze, evaluate, and mitigate failure risks in assets [7,8]. During the years, several methods have been developed, e.g., RAMS (Reliability, Availability, Maintenance and Safety) analysis, HAZOP (HAZard and OPERability analysis), FMECA (Failure Mode, Effect, and Critically Analysis) and Bayesian networks (BN) as reported in numerous reviews in the literature. Li [9] presented state-of-the-art reviews focusing on different simulation models of assessing the risks of failure applied to power utility systems. Fraser et al. [10] analyzed different methods for assessing equipment failure risks for guidance in maintenance decisions. Aven [11] reported trends in perspectives and approaches to risk assessment and management. Smith [12] and Berg [13] analyzed methodologies applicable for quantifying risks of operable assets by considering different case studies. Kabir [14] presented an overview of both standard and different extensions of fault tree analysis and its application in model-based dependability analysis. Insua et al. [15] presented an overview of Bayesian methods applied to the decision-making process in reliability. Adedipe et al. [16] reported a systematic review and evaluation of existing research on the use of BN models in the wind energy sector. In the same sector, Cevasco et al. [17] presented a review and discussion of the component identification of offshore wind turbines. Hoffmann Souza et al. [18] presented a survey on reliability that can be used to support different types of strategic decisions in the context of Industry 4.0. Chemweno et al. [8] reported a detailed review on risk assessment as support of maintenance decision making, with a particular focus on dependability modelling methods.

By analyzing both the literature and industrial practice, the different dependability assessment methodologies available, spanning from qualitative to quantitative models, have the purpose of investigating the functioning of systems and related processes. Concerning the quantitative models, Reliability Block Diagrams (RBD) and Fault Tree Analysis (FTA), have played a central role in the industrial field since they are based on a high-level formalism that favors the construction and the understanding of a model. Moreover, the main hypotheses about the failure/repair independence of the systems' components from each other and the system working conditions allow a quick model resolution.

In recent years, driven by the need to increase the accuracy of the dependability assessment and overcome the limitation of the previous hypotheses, researchers have started focusing on hybrid models that capture both the stochastic and the deterministic nature of a system so as to consider the physical evolution of a process. The branch of the dependability engineering that has met this challenge is nowadays known as Dynamic Probabilistic Risk Assessment (DPRA) or Dynamic Reliability [19].

The literature offers several methodologies for the modelling of a DPRA problem, such as Stochastic Partial Differential Equations (SPDEs), Piecewise Deterministic Markov Process (PDMP), Markov Regime Switching Models (MRSM) and Stochastic Hybrid Automaton (SHA). In particular, SHA consists of a combination of discrete and continuous states that express in each state a specific way of working of the system, through a set of characterizing mathematical equations. Transitions among these states are governed by probability distributions [20]. One advantage of this method over the previous types of formalism is that the variety of behaviors that can be captured is greater, and this facilitates

the characterization of multistate systems; moreover, as the model complexity increases and an analytical closed solution is not available, Monte Carlo simulation can be used.

Repairable systems are a special subset of multistate systems. In fact, restorations can trigger complex relationships among the system components, like changes of spare priority policies, load sharing, degradation, redundancy mechanisms and so forth. Moreover, in the modelling of a repairable system, one of the main concepts to consider is the type of maintenance policy used to restore or retain a system within an acceptable boundary of operating conditions. Generally, maintenance can be based on unplanned or planned actions with the aim to provide optimum system reliability, availability and safety performance at the lowest possible maintenance costs [21]. In the first case, a component is restored (or replaced) as soon it presents evidence of a malfunction which could affect the system operations. In the latter case, maintenance is performed periodically to avoid the failure of a component. Therefore, the dependability analysis of such systems is a complex task as it requires tools able to consider all these modelling features [22].

In a previous work, a DPRA-oriented software library (SHyFTOO) for the analysis of Stochastic Hybrid Fault Tree Automata (SHyFTA) was presented [23]. The SHyFTOO library offers several features to aid with the dependability analysis of a complex repairable system, including a recent upgrade that allows the coupling with the Matlab-Simulink toolbox [24]. This simplifies the design of a dynamic system and, on the other hand, favors the implementation of complementary software libraries that support the analysis of other system characteristics.

In this manuscript, the authors focus on the modelling of maintenance in order to extend the SHyFTOO library. To this end, a designed ad hoc Simulink model is developed and then integrated within the native library aimed at covering other maintenance strategies in DPRA modelling. Finally, to assess the performance of the proposed model, the case study reported in [25] is used as reference case.

The remainder of this paper is presented as follows: in Section 2, a short survey about the most well-known maintenance policies is carried out. Section 3 illustrates the Dynamic Reliability research framework summarizing the architecture of the SHyFTOO library. Moreover, the Maintenance Toolbox, which is the main novel object of this paper, is introduced. A case study is finally discussed in Section 4 and used to compare the results considering the different maintenance policies. The conclusions and future research are depicted in Section 5.

## 2. State of the Art

### 2.1. Maintenance Policies

Maintenance planning consists of all the scheduled set of procedures, activities, or tasks to follow aimed at both maintaining proper reliability equipment and preventing undesirable downtime and breakdowns. These activities involve the definition of a proper maintenance policy and job planning and scheduling [26]. In accordance with the time of maintenance executed, maintenance-types classification is usually divided into two main classes: (i) corrective maintenance (CM) consisting of all the rules and actions carried out after the system breaks down and (ii) preventive maintenance (PM) consisting of preventing the failure by means of all the procedures and activities carried out during system operation. According to the definition reported by [27], the activities that characterized corrective maintenance strategy are grouped into minimal repair (1C) and corrective replacement (2C). The former consists of the procedures involved in restoring the system to the failure rate it had when it failed without system time modification, while the latter involves the replacement of the system. Thus, the system time restarts to zero while the reliability curve belongs to the new implemented system. Concerning preventive maintenance, it is characterized by two different activities: simple preventive maintenance (1P) and preventive replacement (2P). The former involves the modification of the system reliability to some newer time aimed at maintaining the system in the normal operating conditions,

while the latter consists of the implementation of corrective replacement aimed at restoring a new reliability curve in the system state.

PM activities can be classified into periodic maintenance or time-based maintenance (TBM), and predictive maintenance or condition-based maintenance (CBM). Time-based maintenance is a conventional maintenance practice based on a calendar schedule, in which maintenance tasks (e.g., preventive repair times/intervals) are performed on a system at fixed or predetermined time intervals. Thus, to estimate the system aging, failure time data or used-based data [28] are adopted. According to the hazards or failure rate trends, known as bathtub curves [29], the TBM strategy assumes that the failure behavior of the considered system is predictable.

Finally, condition-based maintenance has attracted much attention over the last decade, as reported in [30,31], since the widespread deployment of sensors, actuators and controllers. According to [32], CBM involves maintenance procedures and activities carried out according to the information collected through the condition monitoring process. This process allows detecting indicative prognostic parameters aimed at providing proper information on potential or incipient faults. A detailed overview of CBM is provided by Noman et al. [33] using bibliometric methods, and by Alaswad et al. [34] focusing on mathematical modelling and optimization approaches.

## 2.2. Dynamic Reliability Modelling

To account for the complexity of processing systems, more elaborate methods are required to evaluate the performance and reliability of these systems aiming at detecting their dynamic behavior properly. One of the main factors in reliability modelling of complex dynamic systems is to consider the dynamic influence of the operations, environment, or interactions between components, processes and system performance over the failure rates or degradation process.

To this end, dynamic reliability has been proposed by combining the traditional reliability methods with process modelling, automation and dynamic systems theory. Erguido et al. [35] proposed a dynamic opportunistic maintenance model for wind power generation systems to optimize the total life cost cycle (LCC) and to improve the WF energy-based availability. Zhou et al. [36] presented a dynamic reliability-centered maintenance method for natural gas compressor stations. The decision-making process determined that equipment may not benefit from frequent maintenance in terms of both availability and cost. In [37] the authors developed a numerical method based on piecewise deterministic Markov process to determine the optimal maintenance time for a heated hold-up tank. In [38] the authors studied the reliability centered maintenance strategies for process plants. Liu and Zio [39] investigated the dynamic reliability assessment and failure prognostics of a system with dependent multistate/continuous degrading components. Tsai et al. [40] presented the optimization of the periodic PM of a system with deteriorated components by considering both 1P and 2P activities at different PM stages for a mechanical system.

The authors in [41] proposed a new dynamic fault tree simulation based on an event-driven approach. An optimization of CBM strategies for a slowly degrading system depending on soft failure and condition monitoring at equidistant, discrete time epochs is provided by [42]. A model to integrate production and preventive maintenance planning model is proposed by [43] for a multistate system with independent and common cause failures.

Finally, a comprehensive review is reported in [4] on decision making in PM for smart manufacturing applications.

## 3. Dynamic Reliability Framework

The reliability function  $R(t)$ , also referred to as the survival function, is defined as the probability that the device is still functioning at time  $t$ . and it is expressed as follows:

$$R(t) = P(S(t) > T_M) = 1 - \int_0^{T_M} f_s(t) dt \quad (1)$$

where  $S$  represents the stochastic variable describing the time  $t$  of surviving of a system while  $f_s(t)$  is the probability density function of the failure component. In reliability modelling, Dynamic Reliability method has been conceived as an extension of the traditional theory in order to relax the hypotheses of the classical approach. The main goal is to formulate mathematical models able to reflect numerous characteristics of complex systems and, in particular, the effects of the physics of a process into the system. The implementation of the Dynamic Reliability model allows assessing the aging process of a system overcoming the traditional reliability theory restriction. Indeed, in the traditional approach, the aging effect is simplified assuming a linear dependency with the mission time (e.g., at time  $t = T_m$  the system has operated uninterruptedly and its working condition is changed in terms of functionality, availability, reliability, and safety as it operated for  $T_m$  unit time). This assumption results in a reduced quality of the model. To this end, Manno et al. [44] adopt a Piecewise Markov Process to consider the actual aging  $L(t)$  of a component; thus, its effects are accounted for only when such a component is operating. Accordingly, the nonlinear aging  $L(t)$  can be introduced in place of linear variable  $t$  within the Equation (1) as reported:

$$R(t) = P(S(L) > T_M) \quad (2)$$

Dynamic reliability models have to consider the influence of the working and operational conditions on the system operations. The generic formulation of such influence can be written as follows:

$$W = (t, X(t), e) \quad (3)$$

where  $\Gamma$  is a nonlinear discrete function that depends on three different variables: the time  $t$ , the system dynamic  $X(t)$  and the vector  $e$ . The latter consists of stochastic variables that allow mapping the function  $W$  into a set of discrete states aimed at identifying the possible operational conditions of the system. Therefore, Equation (2) should be rewritten taking into consideration all these effects, as follows:

$$R(t) = P(S(L) > T_M, W, X(t)) \quad (4)$$

The dynamic reliability formulation as reported in Equation (4) reveals the dependencies with respect to the aging  $L$ , the working conditions  $W$  and the system evolution  $X$ . As it can be seen, this equation is a conditional and joint probability that belongs to the class of Stochastic Partial Differential Equations (SPDEs). The complexity of such problems requires a wide variety of mathematical techniques and high computational efforts. In this context, the simulation-based approach represents a key aspect of solving SPDEs; thus, the main concern of researchers shifts from the resolution algorithms to the modelling tools that must be powerful, intuitive, and user-friendly.

### 3.1. Stochastic Hybrid Fault Tree Automaton

In a recent paper, a new mathematical framework named Stochastic Hybrid Fault Tree Automaton (SHyFTA) [45], specifically designed for dynamic reliability problems, was conceived. This framework builds on the separation of concern paradigm, which aims at undertaking the modelling of a dynamic reliability problem by breaking down the hybrid process of a system into two separate subprocesses, the stochastic and the deterministic, which are first modelled independently and then joined by means of state variables.

Therefore, in such a type of modelling the first step consists of the analysis and synthesis of the two abovementioned mathematical models, separately. In particular, the stochastic process is designed using the Dynamic Fault Tree modelling formalism, whereas the deterministic one can be modelled with any mathematical or logic formalism.

A Stochastic Hybrid Fault Tree Automaton can be implemented by means of its characteristic set of variables constituted by 13 elements:

$$\{S, e, X, Y, \delta, H, G, F, P, GA, BE, T, C\} \quad (5)$$

defined as follows:

- $S$  corresponds with the discrete set of stochastic and deterministic states  $\{S_S, S_D\}$ ;
- $e$  corresponds with the discrete set and stochastic events  $\{e_D, e_S\}$ ;
- $X$  corresponds with the discrete set of time variables in  $\mathbb{R}^+$ ;
- $Y$  corresponds with the discrete set of arcs  $g: g^g \leftarrow g^o$  where  $g^g$  and  $g^o$  are, respectively, the goal and the origin states;
- $\delta: S \times X \rightarrow (\mathbb{R}^{n^+} \rightarrow \mathbb{R})$  is an activity function that model the evolution of  $X$  in  $S$ ;
- $H$  corresponds with the discrete set of time-points in  $\mathbb{R}$  triggering the occurrence of a stochastic or a deterministic event;
- $G$  corresponds with the discrete set of functions of type “guard-condition” for the generic variable  $X_i$  on the generic state  $s_j$ ;
- $F: H \times S \times X \rightarrow (\mathbb{R}^{n^+} \rightarrow [0, 1])$  is an application that, varying with continuity in time (time-points), associates to a stochastic event  $e_S$  of a variable  $X_i$  in the discrete state  $s_j$  the corresponding probability distribution;
- $P$  is the probability of the system to be in  $s_i \in S_S$ ;
- $\Pi$  corresponds with the discrete set of the fault tree gates;
- $BE$  corresponds with the discrete set of basic events, including the class of Hybrid Basic Events [44];
- $T$  corresponds with the Top-Event of the Fault Tree;
- $C$  corresponds with the link between basic event and gates.

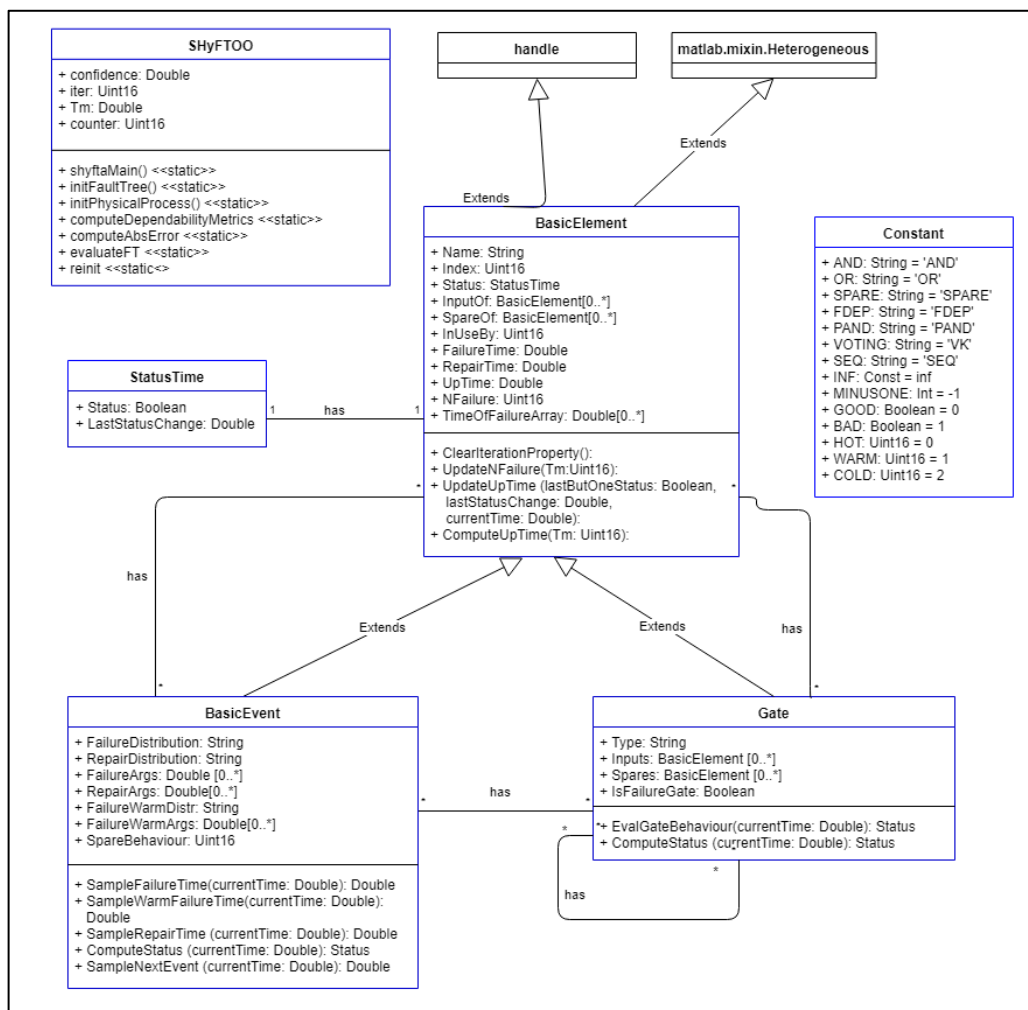
For instance,  $S_S$  is the set of stochastic states (e.g., failed, working, under maintenance) characterizing a system; likewise,  $e_S$  can represent the events that bring the system to move from one stochastic to another stochastic event (e.g., the working to failure event, or vice versa). The same applies to the deterministic counterpart of a SHyFTA model ( $S_D, e_S$ ). The evolution of the system among the states  $S_S$  or  $S_D$  is ruled by means of activity functions  $\delta$  and can be inhibited or activated by the guard conditions  $G$ . The activities can be timed or instantaneous and are described by generic function  $F$  that can vary depending on the conditions of a system (e.g., in a dynamic reliability scenario, the failure rate of a component is not fixed but depends on the working conditions). From the system point of view, the high-level behavior of a SHyFTOO, can be described by means of a Dynamic Fault Tree. Dynamic Fault Tree is a high-level formalism that allows modelling the stochastic and timed interdependencies of a system, simplifying the designing of multistate and multicomponent systems' behaviors. In fact, the functional relationships among the system components can be modelled with the PAND (Priority And Gate), SEQ (Sequence Enforcing), SPARE and FDEP (Failure Dependency) gates. Moreover, the SHyFTA formalism includes a novel gate named SHA that can process both deterministic and stochastic variables [23].

Due to its inherent complexity, the modelling and simulation of a SHyFTOO can be simplified by using the SHyFTOO library, resumed in the next section.

### 3.2. The SHyFTOO Library

The modelling and resolution of a Stochastic Hybrid Fault Tree models is not an easy task. For this reason, in a recent paper [23], an open source library, named SHyFTOO, has been coded in Matlab<sup>®</sup> and released under the Academic Free License v3.0. The SHyFTOO exploits the Object-Oriented paradigm offered also within the Matlab<sup>®</sup> environment. This facilitates the reusability and extension and reusability of its core software components. Moreover, it supports the modelling of the Extended Repairable Dynamic Fault Trees, therefore it does not suffer from the limitation of the implementation of a cascade of dynamic gates.

Figure 1 shows the UML class-diagram of the library including the dependencies among the classes. The black-contour classes correspond with the built-in classes of Matlab<sup>®</sup>, namely those classes which are native of the Matlab<sup>®</sup> framework, whereas the blue-contour classes are the ones that characterize the SHyFTOO library.



**Figure 1.** SHyFTOO library: UML Class diagram (developed in Matlab®).

For instance, the `matlab.mixin.heterogeneous` has been used as superclass (of the `BasicElement`) in order to support the implementation of heterogeneous arrays, whereas the class `handle` has been used to support the reference of a variable (of the library) by address. The object-oriented paradigm allows to extension of parents' class, the implementation of the `BasicElement` class as superclass of `BasicEvent` and `Gate` classes. In fact, these latter are characterized by a set of identical properties (e.g., `Index`, `Name`, `Status`, `FailureTime`, etc.) which are inherited from the `BasicElement` class and by other properties which are defined in their specialized classes.

Thanks to this formalism, the Fault Tree model can be defined as a collection of objects of the type `BasicElement` (no matter if they are `BasicEvent` or `Gates`):

1. A one-to-one relationship between the class `StatusTime` and the class `BasicElement`. The shared variable which links these two classes is the attribute `Status` (datatype `StatusTime`) of the `BasicElement` instance; in the simulation of the Fault Tree, this property provides the current status of the component and the last firing time (the point-time H);
2. A n-to-n relationship between the classes `BasicElement`, `Gate` and `BasicEvent`: any generic instance of type `BasicElement` can be the input of one or more other `BasicElement` instances. For instance, a repeated `BasicEvent` for two or more `Gates`, or when a `Gate` or a `BasicEvent` can be the input of multiple `Spare` gates.

Clearly, the SHyFTOO library consists of several other SW components. The SHyFTOO package is downloaded with some components that, according to the model to simulate, need to be edited. Among the relevant files to modify, it is worth mentioning the following:

- SHyFTAmain.m: this is the main file of a SHyFTA model; it contains the main parameters characterizing the simulation (e.g., Time of Simulation, Iterations, and so on);
- initFaultTree.m: in this file, it is possible to specify the structure of the fault tree model;
- shyftaMetrics.m: in this file it is possible to code ad hoc variables used to define metrics which are out of the scope of the native SHyFTA analysis (e.g., reliability, availability of each component are built-in metrics). For instance, it can be used to implement Importance Measure Analysis, sensitivity, or other metrics.

For further information, interested readers can refer to [23], which explains the main properties of each class and how to construct a SHyFTA model. The structure of the generic initFaultTree.m is shown in Figure 2. The main information to include in this script are the mission time  $T_m$  and the events and gates constituting the fault tree.

```

%% Define the Fault Tree Structure %%
Tm = 8760; %[h]

%% Define BEs %%
HBE1 = BasicEvent('HBE1','hybrid','exp',[10000,1.2],[1/48]);
BE2 = BasicEvent('BE2','exp','', [5E-4], []);

%% Define Gates %%
AND1 = Gate('AND1','AND',true,[HBE1, BE2]);
TOP = AND1;
%% Recall Matlab Script %%
%verify if the FT Structure is valid (it will modify the value of the variable UNVALID_FT)
createFTStructure

```

Figure 2. Example of fault tree with the SHyFTOO library.

In [24], the SHyFTOO library has been extended to support the integration with Simulink. This represents a powerful feature as it alleviates the efforts of engineers and risk practitioners as far it concerns the modelling of a dynamic system. The coupling of the SHyFTOO library with Simulink is achieved by an additional software component named SHyFTA.slx that has to describe the deterministic process characterizing the physical dynamics of the model. In order to work, the Simulink file SHyFTA.slx has to contain some built-in SHyFTOO blocks, which are compulsory. To simplify the coding of the SHyFTA.slx, the SHyFTOO library package provides a template model (file SHyFTA\_TEMPLATE.slx) containing, as said, the basic components. Figure 3 shows the SHyFTA\_TEMPLATE.slx model.

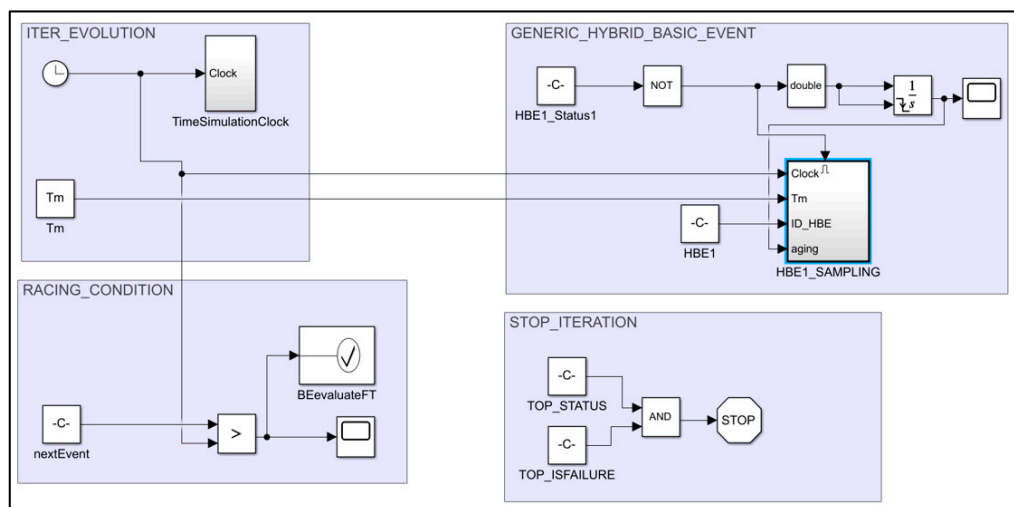



Figure 3. SHyFTA\_TEMPLATE.slx file of the SHyFTOO library package.




The important Simulink components shown in Figure 3 are needed to implement the shared variables that couple the stochastic components defined in the Fault Tree model (initFaultTree.m file) with the physical model. To better understand the functioning of these components, the following details are assumed:

- The “iter evolution” (see Table 1) has to be used without any change. In fact, it is the block that controls, for each iteration, the simulation clock of Matlab® and verifies if the time of mission  $T_m$  has been reached to start automatically a new iteration;
- The “race condition” (see Table 2) must be used without any change. This block is needed to verify, among the traditional Basic Event (handled in the Fault Tree) and the Hybrid Basic Events (which play a role also in the physical process), which component will cause the next firing event. To achieve this task, this block is in a race condition against all the “generic hybrid basic event” blocks (one for each Hybrid Basic Event) as reported in Table 4. If the nextEvent time-point is higher than the Simulink clock it means that the Basic Event of the Fault Tree model has triggered sooner than a Hybrid Basic Event. In this case, the iteration is paused, and the status of the Fault Tree is evaluated. Otherwise, as soon a “generic hybrid basic event” triggers (before the race condition) it means that a Hybrid Basic Event is causing the next firing event.
- The “stop iteration” (see Table 3) is the block that ends the iteration when the Top Event occurs, and the fault tree model is set for evaluating the system reliability. In fact, in that case, a failure of the Top Event cannot be reverted if a component is repaired.

**Table 1.** Main elements of the “iter evolution” block.

Block/Variable	Description
Mission Time $T_m$	Mission time of the process. This variable is defined in the SHyFTAmain.m
Clock 	Clock of Simulink. This is a built-in component of Matlab/Simulink.
TimeSimulationClock	This block evaluates if the mission time has been reached and restarts a new iteration.



**Table 2.** Main elements of the “race condition” block.

Block/Variable	Description
nextEvent	This variable keeps track of the basic events in charge to trigger when their status change.
Relation Operator 	This block is a built-in component of Matlab/Simulink. It allows the comparison of two inputs. The result is a Boolean value.
BEevaluateFT	It is a Simulink Assertion that allows putting in race condition the Basic Events with all the existing Hybrid Basic Events, modelled by the generic hybrid basic event blocks.

**Table 3.** Main elements of the “stop iteration” block.

Block/Variable	Description
STOP	This is a Simulink built-in block assertion. When a condition is verified the STOP simulation is triggered and a new iteration is restarted.
TOP STATUS	This is the shared variable that contains the information of the Top Event status (failed or working)
TOP_ISFAILURE	This shared variable is set with the IsFailureGate property of the TOP Gate. If the IsFailureGate is True and the top event gets failed (TOP STATUS = True), then the STOP Assertion is raised.

**Table 4.** Main elements of the “generic hybrid basic event” block.

Block/Variable	Description
HBE_Status	This block contains the shared variable of the status of the corresponding Hybrid Basic Event modelled by the corresponding “generic hybrid basic event” block. In order to create an ad hoc model, the HBE_Status of each “generic hybrid basic event” block must be modified.
Relation Operator 	This block is a built-in Matlab/Simulink component that performs the neglection of a Boolean input. It must not be modified. Accordingly, the output is also a Boolean (false if the input is true and true in the other case).
HBE_Name	This block has to be set with the name of the Hybrid Basic Event modelled by the corresponding “generic hybrid basic event” block In order to create an ad hoc model, the HBE_Name of each “generic hybrid basic event” block must be modified.
Integrator 	This block is used to integrate and sole the Piecewise Deterministic Markov Process modelling the dynamic aging of the Hybrid Basic Event component.

As said, for each Hybrid Basic Event, the SHyFTA (.slx) Simulink model has to contain a block of type “generic hybrid basic event”. The task of this block is to verify the status of the generic hybrid basic event that is in race condition against all the other hybrid basic events and the regular basic events. Conversely from the previous blocks, for each “generic hybrid basic event”, the setting of the HBE\_Status and HBE blocks must be modified according to the dynamic and physical features of the process of the basic event.

### 3.3. The Maintenance Box

The default maintenance policy of repairable components supported natively in the SHyFTOO library is the 2C (correcting and restoring as new). In this manuscript, we are going to show how to integrate other maintenance strategies by designing an ad hoc Simulink model, the Maintenance Box, that can be configured accordingly. To do that, the generic Simulink block of a Hybrid Basic Event has to be modified and coupled to the Maintenance Box as shown in Figure 4. The implementation of the Maintenance Box allows supporting both the corrective and the time-based preventive maintenance policies discussed in Section 2. The main principle is to act into the reset trigger of the aging variable of the Hybrid Basic Event only if the component is supposed to be restored as good as new. In order to do that, a gain variable defined in the SHyFTAmain.m script must be set. This variable is used in the gain factor K of the Maintenance Box to update the value of the aging of the Hybrid Basic Event.

The preventive maintenance policy can be configured by setting the Base Time variable with the value of the period of the time interval of inspection.

Table 5 shows the main blocks of the Maintenance Box whose setting variables are defined in the SHyFTAmain.m.

**Table 5.** Main elements of the Maintenance Box block.

Block/Variable	Description
Corrective/Preventive	This block contains the setting variable that defines whether the restoration model has to follow a Corrective or a Preventive Maintenance. In the SHyFTAmain.m the variable to set is named “Corrective” and can take the value 0 (Corrective) or 1 (Preventive).
Base Time	This block contains the setting variable that defines, for each component, what is the time to wait before performing preventive maintenance. In the SHyFTAmain.m the variable to set is named “MaintenanceTime”.
K (Gain)	This block contains the setting variable that defines if the component is restored as good as new, as bad as old or to a newer aging. The parameter has to be set to Inf (as good as new), 0 (as bad as old) or to any value $K > 1$ . In this latter case, the newer aging will be given by the ration Aging/K.

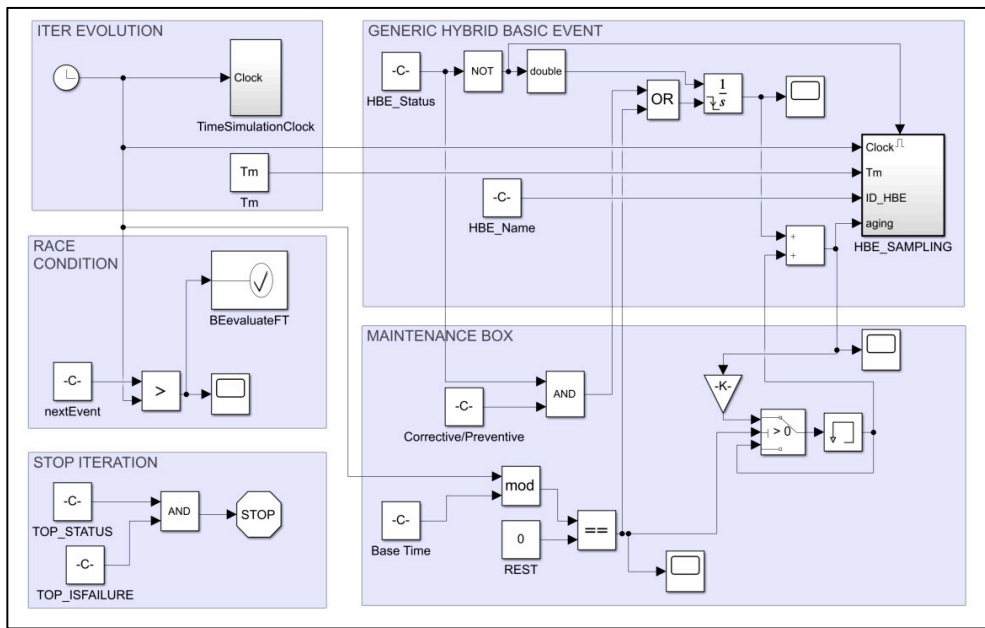


Figure 4. Maintenance Box coupled with the “generic hybrid basic event” block.






### 3.4. Validation of the Maintenance Box

To demonstrate the functioning of the Maintenance Box, a validation use case has been solved and compared with an equivalent Stochastic Activity Network (SAN) model, solved with the Mobius® Software [46].

The Stochastic Activity Network is a high-level formalism that generalizes Stochastic Petri Nets (SPN). Whereas the latter are constituted by places and transitions (timed or instantaneous), SAN offers other elements and mechanisms for the construction of complex stochastic models. The basic components of Stochastic Activity Networks are places, activities (or transitions), Input Gates (IG), and Output Gates (OG). Places and activities (or transitions) can be used with the same logic of a Stochastic Petri Nets; but, in SAN the place marking can be either discrete (like in SPN) and continuous by the means of Extended Place (EP).

Timed activities can support different types of distribution functions that define the firing time of an activity. Among them, we can recall that both deterministic timed and stochastic activities are allowed. The parameters of the distributions can be marking dependent. Finally, Input and Output Gates allow enrichment of the logic that controls the activation and the reactivation of a transition and can be also used to modify the marking of the network. Table 6 illustrates the main elements of the SAN formalism as implemented in the Mobius® tool. It is important to point out that the Mobius® tool allows users to code the logic of the Input and Output Gates with the programming language C++; moreover, it supports the definition of new types of entity (structs and vector of structs) with properties that can be referred inside the Input and Output Gates.

**Table 6.** Main elements of the Stochastic Activity Network (SAN) formalism.

Symbol	Description
	Traditional Place counts the number of tokens contained in the place. This element uses the same logic of places of Stochastic Petri Nets (SPN).
	Extended Place can be used to count any generic type of marking. This applies to real number or discrete entities (that can be also structs or vectors). In particular, for real numbers, the Mobius® tool support float or double type.
	Input Gate can be used to define the logic for the activation of a transition and further actions that the SAN has to perform before the transition has started. The logic can be coded using the programming language C++, supported in the Mobius® tool.
	Output Gate defines the actions that follow the completion of an activity. Generally, this component embeds the logic that defines the marking of the SAN at the end of a transition; using the programming language C++, supported in the Mobius® tool, the codification of the SAN behavior is much simpler than using the SPN formalism.
	Timed and Instantaneous Transition: define the occurrence of an event that depends on the marking of the SAN. In the SAN the transitions are enabled with a marking and their enabling condition are embedded in the Input Gate linked to the transition. A timed transition starts as soon an event of the SAN is verified and triggers after a deterministic or stochastic sampled time. Conversely, an instantaneous transition occurs as soon the event is verified.

The use case implemented for the validation of the Maintenance Box is constituted by a simple AND Gate that takes as input a Hybrid Basic Event (HBE1) and a traditional Basic Event, BE2 has been tested.

Table 7 illustrates the failure/repair rates parameters adopted for the system components, considering a mission time  $T_m = 8760$  h.

**Table 7.** Parameters of the system components.

ID	Failure Distribution	Failure Parameters	Repair Distribution	Repair Rate
HBE1(A)	Weibull	$\beta = 1.2; \gamma = 10,000$	Exponential	$1/48$ ( $h^{-1}$ )
BE2 (B)	Exponential	$5 \times 10^{-4}$ ( $h^{-1}$ )	-	-

The proposed benchmark of test is characterized by a hybrid Basic Event, HBE1, that is affected by aging. Two scenarios have been tested: the corrective (2C) maintenance and a preventive maintenance (2P). In fact, as a result of the maintenance activities, this component can be replaced as good as new (aging is reset to 0). Moreover, for the maintenance policy 2P, it is assumed that it occurs in the system every 2000 h.

As said, in order to account for the aging process of HBE1, a Weibull probability density function with shape parameter  $\beta = 1.2$  ( $\beta > 1$  models the increasing of failure rate with time) can be used:

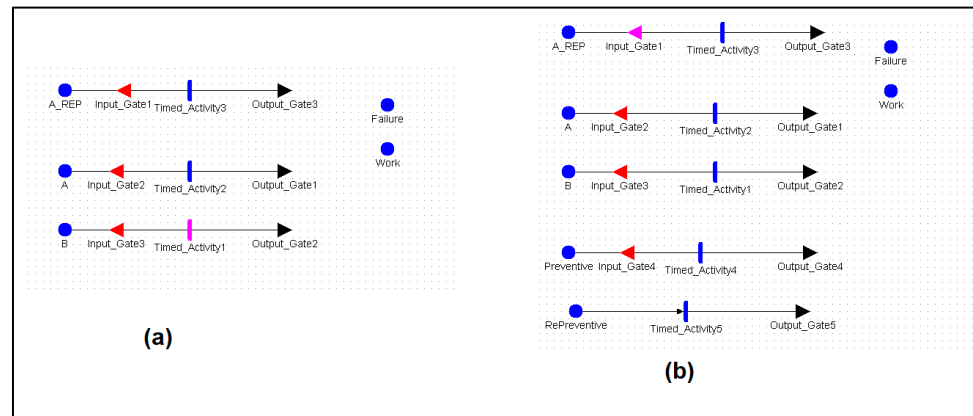
$$f(t) = t/\gamma \cdot (t/\gamma)^{\beta-1} e^{-\left(\frac{t}{\gamma}\right)^\beta} \tag{6}$$

If in the previous equation  $\beta = 1$ , the probability density function takes the form of the exponential distribution, typical of random failures, with failure rate  $\lambda = 1/\gamma$ .

From Equation (6) it is possible to derive the instantaneous Weibull failure rate function:

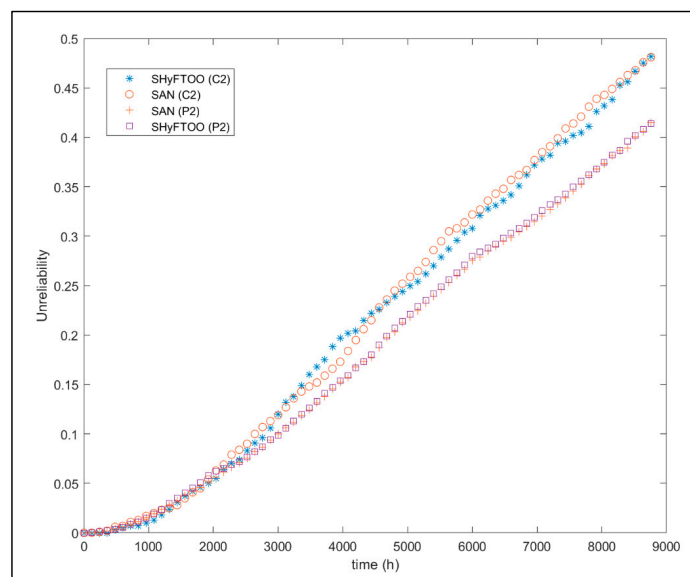
$$f(t) = \frac{f(t)}{1 - F(t)} = \beta/\gamma \cdot (t/\gamma)^{\beta-1} \tag{7}$$

Figure 5 shows the Stochastic Activity Network models of the proposed benchmark AND gate designed to test the effectiveness of the SHyFTOO Maintenance Box. In particular, the Stochastic Activity Network of Figure 5a models the AND gate with the corrective maintenance (2C) policy, whereas Figure 5b can account for the preventive policy 2P. Figure 6 illustrates the comparison of the SHyFTOO Maintenance Box and the SAN model for the 2C and 2P maintenance policies. The results obtained confirm that the Maintenance Box model retrieve the correct results.



**Figure 5.** Stochastic Activity Network model of the AND Gate: (a) with corrective maintenance (2C) and (b) with preventive maintenance (2P).

These results confirm that policy (2P), as expected, is the most effective maintenance strategy. It can be seen that the unreliability of the system is similar but, as soon the first preventive maintenance occurs (at 2000 h) the two groups of trends start to diverge. This is because every 2000 h the preventive maintenance replaces the component as new, restarting the aging to 0.



**Figure 6.** System unreliability for the AND benchmark comparing SAN (Stochastic Activity Networks) and SHyFTOO models.

#### 4. Case Study

The case study presented in this paper models the reliability of a steam turbine. It extends the analysis presented by Salehpour-Oskouei and Pourgol-Mohammad [25] who illustrated the benefits of condition monitoring. Specifically, Salehpour-Oskouei and Pourgol-Mohammad proposed a model that can enable a conditioning monitoring process by adding some sensors for checking the status of the steam turbine subsystems and avoids undesired failures. This modelling can be implemented by the use of a 2-input PAND gate that takes as first input the condition monitoring sensor and a second input the system to monitor. In this way, the PAND triggers only if the condition monitoring sensors fail before the system is monitored.

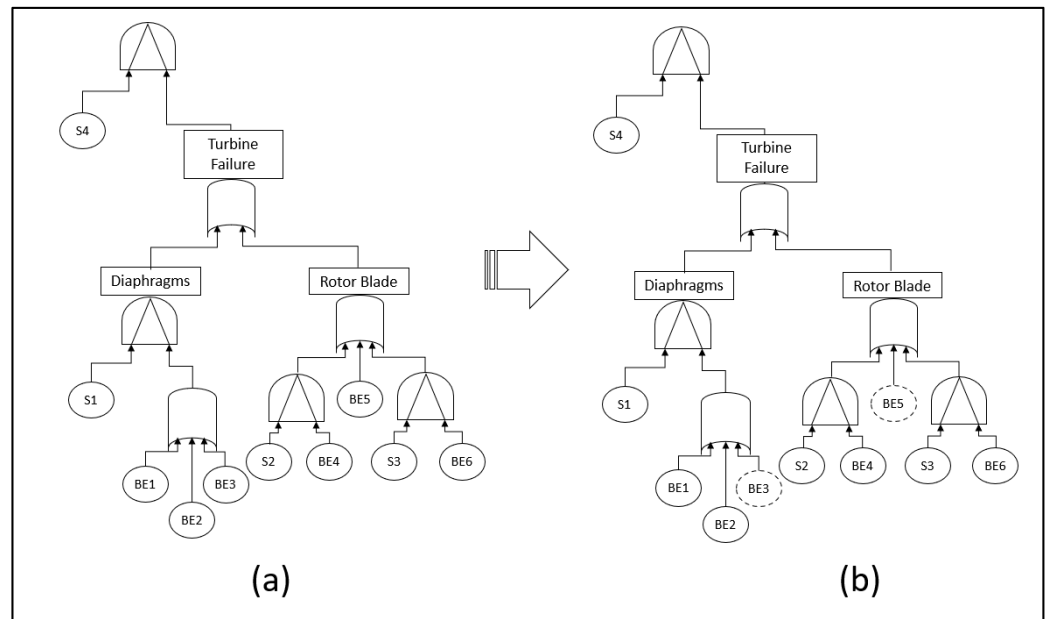
Figure 7a shows the Dynamic Fault Tree of the steam turbine, while Table 8 shows the main parameters of the system components [25]. As can be seen, the simplified model of the steam turbine, based on a FMEA, is characterized by the relevant failure modes that affect the functioning of the Diaphragms and the Rotor Blade. The condition monitoring depends on the sensors S1, S2, S3 and S4 which, respectively, monitor the status condition of the Diaphragms, of the Vibration faults, of the Steam Temperature and, lastly, of the Turbine.

**Table 8.** Parameters of the Dynamic Fault Tree Basic Events.

ID	Description	System	Dist.	Params	Disyt	Params
BE1	Steam Temperature	Diaphragms	Exp.	$\lambda = 2.857 \times 10^{-5}$	-	-
BE2	Steam Humidity		Exp.	$\lambda = 1.176 \times 10^{-4}$	-	-
BE3	Debris Penetration *		Weibull	$\gamma = 8503$ $\beta = 1.2$	Exp.	1/168
BE4	Vibration	Rotor Blade	Exp.	$\lambda = 1.176 \times 10^{-4}$	-	-
BE5	Crack Formation *		Exp.	$\gamma = 7003$ $\beta = 1.2$	Exp.	1/168
BE6	Steam Temperature		Weibull	$\lambda = 2.857 \times 10^{-5}$	-	-
S1	Wireless Accelerometer	Diaphragms	Exp.	$\lambda = 8 \times 10^{-5}$	-	-
S2	Accelerometer	Rotor Blade	Exp.	$\lambda = 1.74 \times 10^{-4}$	-	-
S3	Thermometer	Rotor Blade	Exp.	$\lambda = 1 \times 10^{-6}$	-	-
S4	Tachometer	Turbine	Exp.	$\lambda = 80 \times 10^{-6}$	-	-

What has been missed in the original formulation [25] so far described, is that condition monitoring can actually enable preventive maintenance with the benefit of extending the remaining useful life of the system. In order to do that, it is necessary to code a model that must account for the restoration of those basic events which can be characterized by a repair rate.

As shown in Table 8, in the model thereby proposed, the basic events which can benefit from a restoration are the Crack Formation and Debris Penetration (as highlight with asterisk). Moreover, for these two events, it is assumed that they follow a Weibull distribution in order to model the deterioration effect. This hypothesis suits better than the random fault behavior proposed in [46], that makes use of the exponential distribution. Therefore, the formula of Equation (5) can be used again with  $\beta = 1.2$  and  $\gamma = 1/\lambda_i$ . As for the time of restoration, the repair rate for the Crack Formation or the Debris Penetration has been set to 1/168 h, which corresponds to a restoration mean time of a week.



**Figure 7.** Fault Tree models of the Steam Turbine with condition monitoring sensors: (a) Dynamic Fault Tree model and (b) SHyFTA model.

The other basic events, designated to model the occurrence of an undesired process condition (temperature, humidity and vibration) not related to a fault of a steam turbine component, cannot be characterized by a repair rate, therefore they are considered as nonrepairable events. As for the sensors, it is assumed that the failure event refers to a measuring error due to a wrong calibration or loss of precision, rather than a breakage. Therefore, also for these two basic events, a repair rate is not applicable.

The SHyFTA model of Figure 7b has been coded using the SHyFTOO library. The Monte Carlo simulation process has been set for running 100,000 iterations.

This simulation campaign aims to compare the results of the DFT of Figure 7a versus the SHyFTA model of Figure 7b. For the former, due to the limitations of the DFT modelling, it was assumed to adopt only a policy of type 2C that restores the system as good as new from the Crack Formation and the Debris Penetration, assuming the repair rates of Table 7.

The SHyFTA model has been obtained from the Dynamic Fault Tree of Figure 7a by replacing the basic events BE3 and BE5 of the Debris Penetration and the Crack Formation with the corresponding hybrid basic events HBE3 and HBE5.

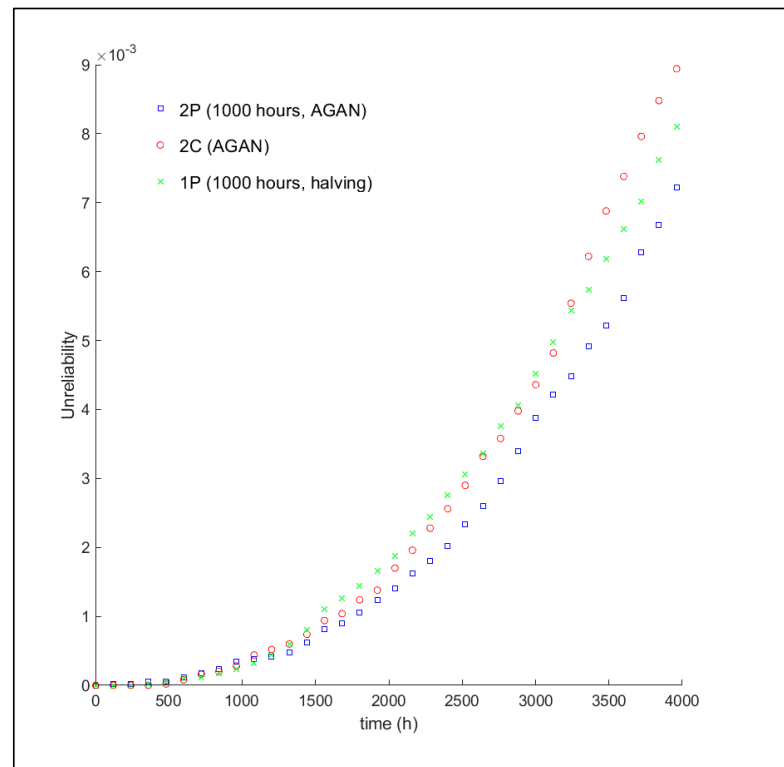
In the SHyFTA model, these latter can be coded so as to hook the time variable to the actual aging of the basic event, and better model the characteristic nonlinear time deterioration of such events. In this way, the aging is modelled by the means of a Piecewise Deterministic Markov Process expressed with the following formula:

$$\frac{dL(t)}{dt} = u(t) \tag{8}$$

where  $u(t)$  is a discrete stochastic variable that measures the instantaneous degradation rate of the component. As for the SHyFTA model, the simulation results will account for two preventive maintenance policies: 1P halves the effect of the Debris Penetration and the Crack Formation in the system ( $L(t) = L/2$ ), whereas the preventive maintenances 2P restores the condition as good as new (e.g.,  $L(t) = 0$ ). The two preventives maintenances take place every 1000 h. Likewise, the DFT simulation, the corrective approach 2C restores the Crack Formation and Debris Penetration aging as good as new.

Figure 8 shows the results of the SHyFTA model of Figure 7b. In this case it is also possible to see that the preventive policy improves the system reliability. In particular,

the policy 2P reduces the system unreliability from  $8.74 \times 10^{-3}$  to the value of  $7.08 \times 10^{-3}$ . The policy 1P performs in between and presents a system unreliability of  $8.05 \times 10^{-3}$ .



**Figure 8.** Steam Turbine unreliability of the SHyFTA model of Figure 7b.

## 5. Conclusions

In this paper, a Stochastic Hybrid Automaton framework has been used to evaluate the effects of different maintenance policies on the reliability of complex and dynamic systems. To this end, the SHyFTOO library [24] has been modified and extended with a new Simulink component, the Maintenance Box, that can be easily configured to accommodate the modelling of corrective and time-based preventive maintenance policies and to evaluate the dynamic reliability of a system.

The toolbox has been validated by comparing the results obtained with it and using the equivalent model of a SAN, confirming that the Maintenance Box model retrieves correct results. Moreover, the applicability of the solutions was demonstrated in a case study modelling the reliability of a steam turbine. As expected, the results of the simulation show that a preventive maintenance that restores components as good as new (e.g., the replacement of the worn-out component) increases the reliability and the availability of a system with respect to a corrective maintenance policy.

The advantages of the use of this type of modelling rely on the fact that the SHyFTA can be coded so as to integrate complex operational aspects such as environmental conditions affecting the working/failure behavior of a system and evaluate the economic impacts of the management decisions.

From an industrial application perspective, the developed modelling solution represents an easy to implement tool to drive practitioners in the area of reliability and maintenance engineering for supporting the decision-making process related to complex and dynamic systems.

The integration of the Maintenance Box to the SHyFTOO library can open the way to further developments by extending it, including the possibility to model a reliability-centered preventive maintenance policy as well. In this way, it will be possible to evaluate and compare more complex scenarios by performing economic analyses including the costs



of replacement or costs linked with the lack of services due to the downtime necessitated by the corrective maintenance.

The tool SHyFTOO can be freely downloaded at the following link <https://github.com/chiacchiof/SHyFTOO-Matlab> (accessed 19 February 2021). The Maintenance Box can be granted for free for academic use and can be acquired by emailing the authors of this paper.

**Author Contributions:** Conceptualization, S.A., I.R. and F.C.; methodology, S.A., I.R. and F.C.; software, F.C.; validation, S.A., I.R. and F.C.; formal analysis, F.C.; investigation, S.A., I.R. and F.C.; resources, S.A.; data curation, F.C.; writing—original draft preparation, S.A. and F.C.; writing—review and editing, I.R. and F.C.; visualization, S.A.; supervision, I.R.; project administration, I.R.; funding acquisition, S.A. and F.C. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Not applicable.

**Acknowledgments:** This research was realized with the contribution of PON R&I 2014–2020—AIM (Attraction and International Mobility), project AIM 1815402-1.

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

## References

1. Monsone, C.R.; Mercier-Laurent, E.; János, J. The overview of digital twins in industry 4.0: Managing the whole ecosystem. In Proceedings of the IC3K 2019, 11th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management, Vienna, Austria, 17–19 September 2019; Volume 3, pp. 271–276.
2. Mourtzis, D.; Vlachou, E. A cloud-based cyber-physical system for adaptive shop-floor scheduling and condition-based maintenance. *J. Manuf. Syst.* **2018**, *47*, 179–198. [\[CrossRef\]](#)
3. Montero Jimenez, J.J.; Schwartz, S.; Vingerhoeds, R.; Grabot, B.; Salaün, M. Towards multi-model approaches to predictive maintenance: A systematic literature survey on diagnostics and prognostics. *J. Manuf. Syst.* **2020**, *56*, 539–557. [\[CrossRef\]](#)
4. Bousdekis, A.; Lepenioti, K.; Apostolou, D.; Mentzas, G. Decision Making in Predictive Maintenance: Literature Review and Research Agenda for Industry 4.0. *IFAC Pap.* **2019**, *52*, 607–612. [\[CrossRef\]](#)
5. Roda, I.; Arena, S.; Macchi, M.; Orrù, P.F. Total Cost of Ownership Driven Methodology for Predictive Maintenance Implementation in Industrial Plants. *IFIP Adv. Inf. Commun. Technol.* **2019**, *566*, 315–322.
6. Kari, K. A cost model of industrial maintenance for profitability analysis and benchmarking. *Int. J. Prod. Econ.* **2002**, *79*, 15–31.
7. International Organisation of Standardisation ISO 31010. *Risk Management-Risk Assessment Techniques*; ISO: Geneva, Switzerland, 2009.
8. Chemweno, P.; Pintelon, L.; Muchiri, P.N.; Van Horenbeek, A. Risk assessment methodologies in maintenance decision making: A review of dependability modelling approaches. *Reliab. Eng. Syst. Saf.* **2018**, *173*, 64–77. [\[CrossRef\]](#)
9. Li, W. *Risk Assessment of Power Systems: Models, Methods, and Applications*; John Wiley & Sons: Hoboken, NJ, USA, 2014.
10. Fraser, K.; Hvolby, H.-H.; Tseng, T.L.B. Reliability paper Maintenance management models: A study of the published literature to identify empirical evidence a greater practical focus is needed. *Int. J. Qual. Reliab. Manag.* **2015**, *6*, 635–664. [\[CrossRef\]](#)
11. Aven, T. Risk assessment and risk management: Review of recent advances on their foundation. *Eur. J. Oper. Res.* **2016**, *253*, 1–13. [\[CrossRef\]](#)
12. Smith, D.J. *Reliability, Maintainability and Risk: Practical Methods for Engineers*; Butterworth-Heinemann: Oxford, UK, 2017.
13. Berg, H.P. Risk management: Procedures, methods and experiences. *Reliab Theory Appl.* **2010**, *1*, 79–95.
14. Kabir, S. An overview of fault tree analysis and its application in model based dependability analysis. *Expert Syst. Appl.* **2017**, *77*, 114–135. [\[CrossRef\]](#)
15. Insua, D.R.; Ruggeri, F.; Soyer, R.; Wilson, S. Advances in Bayesian decision making in reliability. *Eur. J. Oper. Res.* **2020**, *282*, 1–18. [\[CrossRef\]](#)
16. Adedipe, T.; Shafiee, M.; Zio, E. Bayesian Network Modelling for the Wind Energy Industry: An Overview. *Reliab. Eng. Syst. Saf.* **2020**, *202*, 107053. [\[CrossRef\]](#)
17. Cevasco, D.; Koukoura, S.; Kolios, A.J. Reliability, availability, maintainability data review for the identification of trends in offshore wind energy applications. *Renew. Sustain. Energy Rev.* **2021**, *136*, 110414. [\[CrossRef\]](#)
18. Hoffmann Souza, M.L.; da Costa, C.A.; de Oliveira Ramos, G.; da Rosa Righi, R. A survey on decision-making based on system reliability in the context of Industry 4.0. *J. Manuf. Syst.* **2020**, *56*, 133–156. [\[CrossRef\]](#)

19. Labeau, P.E.; Smidts, C.; Swaminathan, S. Dynamic reliability: Towards an integrated platform for probabilistic risk assessment. *Reliab. Eng. Syst. Saf.* **2000**, *68*, 219–254. [[CrossRef](#)]
20. Fan, M.; Zeng, Z.; Zio, E.; Kang, R.; Chen, Y. A stochastic hybrid systems model of common-cause failures of degrading components. *Reliab. Eng. Syst. Saf.* **2018**, *172*, 159–170. [[CrossRef](#)]
21. Pham, H.; Wang, H. Imperfect maintenance. *Eur. J. Oper. Res.* **1996**, *94*, 425–438. [[CrossRef](#)]
22. Aizpurua, J.I.; Catterson, V.M.; Papadopoulos, Y.; Chiacchio, F.; Manno, G. Improved Dynamic Dependability Assessment through Integration with Prognostics. *IEEE Trans. Reliab.* **2017**, *66*, 893–913. [[CrossRef](#)]
23. Chiacchio, F.; Aizpurua, J.I.; Compagno, L.; D’Urso, D. SHyFTOO, an object-oriented Monte Carlo simulation library for the modeling of Stochastic Hybrid Fault Tree Automaton. *Expert Syst. Appl.* **2020**, *145*, 113139. [[CrossRef](#)]
24. Chiacchio, F.; Aizpurua, J.I.; Compagno, L.; Khodayee, S.M.; D’Urso, D. Modelling and resolution of dynamic reliability problems by the coupling of simulink and the stochastic hybrid fault tree object oriented (SHyFTOO) library. *Information* **2019**, *10*, 283. [[CrossRef](#)]
25. Salehpour-Oskouei, F.; Pourgol-Mohammad, M. Fault diagnosis improvement using dynamic fault model in optimal sensor placement: A case study of steam turbine. *Qual. Reliab. Eng. Int.* **2017**, *33*, 531–541. [[CrossRef](#)]
26. Dhillon, B.S. *Engineering Maintenance: A Modern Approach*; CRC Press: Boca Raton, FL, USA, 2002.
27. Lie, C.H.; Chun, Y.H. An Algorithm for Preventive Maintenance Policy. *IEEE Trans. Reliab.* **1986**, *35*, 71–75. [[CrossRef](#)]
28. Lee, J.; Ni, J.; Djurdjanovic, D.; Qiu, H.; Liao, H. Intelligent prognostics tools and e-maintenance. *Comput. Ind.* **2006**, *57*, 476–489. [[CrossRef](#)]
29. Ahmad, R.; Kamaruddin, S. An overview of time-based and condition-based maintenance in industrial application. *Comput. Ind. Eng.* **2012**, *63*, 135–149. [[CrossRef](#)]
30. Grall, A.; Dieulle, L.; Bérenguer, C.; Roussignol, M. Continuous-time predictive-maintenance scheduling for a deteriorating system. *IEEE Trans. Reliab.* **2002**, *51*, 141–150. [[CrossRef](#)]
31. Han, Y.; Song, Y.H. Condition monitoring techniques for electrical equipment—A literature survey. *IEEE Trans. Power Deliv.* **2003**, *18*, 4–13. [[CrossRef](#)]
32. Jardine, A.K.S.; Lin, D.; Banjevic, D. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mech. Syst. Signal Process.* **2006**, *20*, 1483–1510. [[CrossRef](#)]
33. Noman, M.A.; Nasr, E.S.A.; Al-Shayea, A.; Kaid, H. Overview of predictive condition based maintenance research using bibliometric indicators. *J. King Saud Univ. Eng. Sci.* **2018**, *31*, 355–367. [[CrossRef](#)]
34. Alaswad, S.; Xiang, Y. A review on condition-based maintenance optimization models for stochastically deteriorating system. *Reliab. Eng. Syst. Saf.* **2017**, *157*, 54–63. [[CrossRef](#)]
35. Erguido, A.; Crespo Márquez, A.; Castellano, E.; Gómez Fernández, J.F. A dynamic opportunistic maintenance model to maximize energy-based availability while reducing the life cycle cost of wind farms. *Renew. Energy* **2017**, *114*, 843–856. [[CrossRef](#)]
36. Zhou, D.; Zhang, H.; Li, Y.G.; Weng, S. A Dynamic Reliability-Centered Maintenance Analysis Method for Natural Gas Compressor Station Based on Diagnostic and Prognostic Technology. *J. Eng. Gas Turbines Power* **2016**, *138*. [[CrossRef](#)]
37. De Saporta, B.; Zhang, H. Predictive maintenance for the heated hold-up tank. *Reliab. Eng. Syst. Saf.* **2013**, *115*, 82–90. [[CrossRef](#)]
38. Vishnu, C.R.; Regikumar, V. Reliability Based Maintenance Strategy Selection in Process Plants: A Case Study. *Procedia Technol.* **2016**, *25*, 1080–1087. [[CrossRef](#)]
39. Liu, J.; Zio, E. System dynamic reliability assessment and failure prognostics. *Reliab. Eng. Syst. Saf.* **2017**, *160*, 21–36. [[CrossRef](#)]
40. Tsai, Y.T.; Wang, K.S.; Teng, H.Y. Optimizing preventive maintenance for mechanical components using genetic algorithms. *Reliab. Eng. Syst. Saf.* **2001**, *74*, 89–97. [[CrossRef](#)]
41. Gascard, E.; Simeu-Abazi, Z. Quantitative Analysis of Dynamic Fault Trees by means of Monte Carlo Simulations: Event-Driven Simulation Approach. *Reliab. Eng. Syst. Saf.* **2018**, *180*, 487–504. [[CrossRef](#)]
42. Tang, D.; Sheng, W.; Yu, J. Dynamic condition-based maintenance policy for degrading systems described by a random-coefficient autoregressive model: A comparative study. *Eksploat. Niezawodn. Maint. Reliab.* **2018**, *20*, 590–601. [[CrossRef](#)]
43. Alimian, M.; Saidi-Mehrabad, M.; Jabbarzadeh, A. A robust integrated production and preventive maintenance planning model for multi-state systems with uncertain demand and common cause failures. *J. Manuf. Syst.* **2019**, *50*, 263–277. [[CrossRef](#)]
44. Manno, G.; Zymaris, A.; Kakalis, N.P.; Chiacchio, F.; Cipollone, G.; Compagno, L.; D’Urso, D.; Trapani, N. *Dynamic Reliability of Three Nonlinear Aging Components with Different Failure Modes Characteristics*; Taylor & Francis Group, Ed.; CRC Press: Boca Raton, FL, USA, 2013.
45. Chiacchio, F.; D’Urso, D.; Famoso, F.; Brusca, S.; Aizpurua, J.; Catterson, V.M. On the use of dynamic reliability for an accurate modelling of renewable power plants. *Energy* **2018**, *151*, 605–621. [[CrossRef](#)]
46. Courtney, T.; Gaonkar, S.; Keefe, K.; Rozier, E.W.D.; Sanders, W.H. Möbius 2.3: An Extensible Tool for Dependability, Security, and Performance Evaluation of Large and Complex System Models. In Proceedings of the 39th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2009), Estoril, Lisbon, Portugal, 29 June–2 July 2009; pp. 353–358.