Operational dependability model generation

Changyi Xu

N°d'ordre NNT : 2020LYSEI129

THÈSE de DOCTORAT DE L'UNIVERSITÉ DE LYON
opérée au sein de
I'Institut National des Sciences Appliquées de Lyon

Ecole Doctorale 160
Électronique, Électrotechnique, Automatique

Spécialité/ discipline de doctorat :
Automatique

Soutenue publiquement le 16/12/2020, par:

Changyi XU

Operational Dependability Model Generation

Devant le jury composé de :

KOBI Abdelssamad	Professeur (Université d'Angers)	Rapporteur
BERRUET Pascal	Professeur (Université Bretagne Sud)	Rapporteur
LANUSSE Agnes	Ingénieur Chercheur (CEA)	Examinatrice
BRINZEI Nicolae	Maître de conférences (Université de Lorraine)	Examinateur
NIEL Eric	Professeur (INSA Lyon)	Directeur de thèse
DUMITRESCU Emil	Maître de Conférences (INSA Lyon)	Co-encadrant

Acknowledges

This research work was carried out between 2015 and 2020 at the Laboratory Ampere. I would like to express my heartfelt gratitude to all those whole helped me during the completion of this thesis. Without their support and encouragement, this thesis could not be finished.

First and foremost, I owe a special debt of gratitude to my supervisor, Professor Eric NIEL, for his invaluable help. He has walked me through all the stages of this PhD research and thesis writing. I really appreciate his rigorous scientific attitude and his profound accumulation of academic knowledge. His patient guidance and valuable suggestion is the wealth of my life. I also gratefully acknowledge the help of my co-supervisor, Professor Emil DUMITRESCU, who has offered me expert guidance in my academic study. He brought me a positive way of working, especially an optimistic attitude towards life. Without his great devotion of time or his insightful criticism, the completion of this thesis would not have been possible.

Second, I would give my hearty thanks to Mahya RAHIMI, Xiaoshan LU, Tahereh VAEZI, Xiaokang ZHANG, Teng ZHANG, Fei LIU and all the other faculty members in Laboratory Ampere. Their precious suggestions and valuable help in various aspects made me feel like home. Special thanks for Chao ZHANG, Lianxin HU, Lianxin's wife Ruijuan LIU, Xiao LI, Weigang FAN, Nan ZHANG and all my friends in France. We lived together, we helped each other, and we witnessed every smile and every drop of tear. Each moment is my valuable memory.

Third, my sincere gratitude would go to my beloved family and relatives, for their selfless help and solicitude in all sorts of ways. Much indebted to my parents, they have given me my life, they educated me with their life savings, they have supported me in each stage, especially, at any cost， they are always being prepared to raise me up at every moment I am failing down. Also, here expresses the mourning for my grandparents HE&CHANG, who built a big family where members look for each other for every silver lining and every happiness, who brought me up and made my childhood joyous and carefree. As the distance returning home was too long, I could not attend my grandfather's funeral in time, which makes me regretful in the rest of my life. May the past rest in peace, the living predicament.

Last but not the least, I cherish for my motherland-China the liveliest feeling of affection and gratitude. I am grateful for China Scholarship Council for supporting my research and living in France.

Abstract

Assessing complex industrial systems to be on dependable service is what the engineers and researchers have long been aiming for. Recent advanced researches in the Model-based safety assessment, especially the Structre Analysis and Component Modeling, provide the practicable methodologies to assess the dependability, yet a lack of the framework which is able to assess both the structure and the various behaviors of the components in one uniformed model retains them to achieve the excellent assessment. Moreover, as the system's operations are not considerable in the models, the service in the aspect of operational dependability is not able to be assessed both in quality and in quantity. Although several existing assessment tools have already show their potential to model the various behaviors in the form of n-state models or consider the operations as repair priority to be event sequence in the model, fusing 'structure', 'various behaviors' and 'operations' is still a challenge, highlighting a need for one viable framework that bridge the gap among them both by quality or quantity. In this research, a formal model generation approach is studied to bridge this gap, which is able to assess the system operatinal dependability by considering the system structure, various behaviors, and operations. Here, the composition of the component models is introduced in order to generate a global model of the system, the total breakdown states are identified according to the resulted failure expression for the purpose to fully consider the system's structure, and the operational dependability is further realized by quality by applying the trajectory specifications, while by quantity by developing a cost evaluating technology termed Capacity Calculation Fault Tree. In the end, a demonstration of a miniplant system illustrates the wide potential of this research for guaranteeing the dependable service of complex industrial systems.

KEYWORDS: Model-based safety assessment, Operational dependability, Dependability model generation, Capacity calculation fault tree.

Résumé

L'objectif à long terme des ingénieurs et des chercheurs est d'évaluer la fiabilité des systèmes industriels complexes. Les évaluations de la sécurité fondées sur des modèles effectuées ces dernières années, en particulier les études d'analyse structurelle et de modélisation des composants, fournissent des méthodes pratiques d'évaluation de la fiabilité,Toutefois, l'absence d'un cadre permettant d'évaluer simultanément la structure et les comportements des différents éléments d'un modèle unifié n'a pas permis d'obtenir d'excellentes évaluations. En outre, les opérations du système n'étant pas pris en compte dans le modèle, il n'est pas possible d'évaluer la qualité et la quantité du service en termes de fiabilité du opérations. Cette invention concerne un procédé de génération de modèle formalisé qui permet d 'évaluer la fiabilité du fonctionnement du système en tenant compte de sa structure, de ses divers comportements et de ses opérations. La composition du modèle de composant est introduite pour générer un modèle global du système. Afin de tenir pleinement compte de la structure du système, l 'état total de défaillance du système est déterminé sur la base de l' expression de défaillance obtenue. Sur le plan qualitatif, la fiabilité opérationnelle est encore renforcée par l'application des spécifications de trajectoire.Et, Sur le plan quantitif, il est renforcée par la mise au point d'une technique d'évaluation des coûts appelée arbre de calcul de capacité. Enfin, l'exemple d'un système industriel illustre l'énorme potentiel qu'offre l'étude pour garantir la fiabilité des services fournis par les systèmes industriels complexes.

MOTS CLÉS: Évaluation de la sécurité fondée sur des modèles, Fiabilité opérationnelle, Génération de modèle de fiabilité, Arbre de calcul de capacité

Contents

List of Figures

List of Tables

Research Motivation

Industrial systems are expected to fulfill their function while providing qualitative operation guarantees and reasonable confidence about their availability. They involve a large number of interacting components. Behaviors are by nature distributed, and the outputs of some components are fed into the inputs of the other components, according to a predefined structure. The structure of an industrial system covers two aspects. A functional aspect, where components interact in order to provide a global functionality, tackled by design engineers, whose concern is mainly providing the correct service in the expected time. Yet, the inherent complexity of these systems also comes from the number of constituting components and their individual reliability. Component failures may lead to unpredictable situations, synonym of malfunctioning or, for critical systems, to the serious hazard. Thus, a dysfunctional aspect concentrates a large part of the design efforts, and brings additional complexity to the system: additional components are supplied as backup and their operation needs accurate specification that is independent of the functional requirements of the global system. This design effort is compulsory in order to provide continuous operation guarantees and avoid the **total breakdown.** Moreover, the probability and impact of each failure, on the one hand, and of possible sequences of failures on the other hand, need to be accurately evaluated, so that appropriate maintenance actions can be scheduled. Hence, it appears to be interesting to associate operational requirements to conventional dependability requirements in order to assess typical service or recovery policies. This is the object of the **operational dependability assessment**.

Mathematical models have shown interesting features for analyzing failure mechanisms, and system dependability evaluation, which is the original intention of Model Based Safety Assessment, abbreviated MBSA[1, 2]. The word 'safety' in this context is a synonym to the dependability. In MBSA, the system is expressed by its faulty model. Based on this model, time-based indicators may be computed to assess the system's operational dependability, such as the meantime to failure (MTTF), the meantime to repair (MTTR) and the mean time between failures (MTBF). Alternatively, availability, reliability and maintainability probabilities can also be computed.

Failures and repairs can be naturally sensed as events, occurring within a complex scenario. Hence, the discrete-event system modeling approach appears to be well suitable for expressing dysfunctional system models. Mature approaches rely on Stochastic Petri Nets[3] or Markov Processes[4], for indicators calculation, model checking or even simulation.

The BDMP (Boolean Logic Driven Markov Process) approach provides fault-tree based modeling in order to generate the underlying dysfunctional behaviors as a Markov process[5]. One of the BDMP's

advantages is that it makes failure sequences explicit. For instance, the cold redundancy architectures are accurately modeled, because the redundant component failure may only occur after the failure of the primary component, and this is expressed by an appropriate failure sequence.

AltaRica[6] provides a powerful modeling solution under the form of guarded transition systems. AltaRica provides modeling flexibility through an opening towards other modeling languages, such as AADL, SysML and Event-B, for the purpose of enlarging the modeling domain and enhancing the safety assessment.

Beyond the approaches enumerated above, safety-critical systems need to be assessed according to four aspects: reliability, availability, maintainability and safety (RAMS)[1]. The reliability is the ability that a system continuously delivers its expected mission within a given time duration, without any failure occurrence. The availability is the ability that a system continuously delivers its expected mission within a given time duration, considering recoverable breakdowns. The maintainability denotes the ability that the system recovers from the totally broken down to be able to contribute to its attempted mission within a given time duration. The safety denotes the ability of the system to be harmless to human beings within a given time duration.

MBSA provides accurate reliability assessment work environment, provided that the underlying model complies with three requirements:

1. It is able to describe the whole dysfunctional behaviors. For a complex industrial system, involving several interacting components, each component should be modeled according to its dysfunctional behavior. The global behavior is obtained from the combination of local behaviors, according to a systematic structure which is designed for delivering the system service;

2. It is able to target specific configurations, and provide mechanisms to associate dysfunctional trajectories to these configurations, in order to compute dependability indicators. These configurations are usually modeled by states. For instance, a component model may feature several nominal operating states, such as idle, working, besides the breakdown state;

3. It is able to integrate additional requirements, such as qualitative or quantitative aspects. Qualitative requirements may be featured by precedence constraints or priorities. For example, in case several components are waiting for repair, the most important components should have the priority to be repaired first. Quantitative aspects are mostly related to costs of repair trajectories. Additional estimations of the resulting “health” of the repaired system may also be considered.

In order to guarantee the accuracy of the assessment, these aspects need to be expressed and handled formally. Yet, no research results exist up to now, offering dependability assessment features and handling in parallel the operational consideration of dependability. For example, the model generated from GRIF [7] or from AltaRica is short of an extension for these operational considerations.

In turn, BDMP is able to consider failure sequences starting with a trigger event, and also considers complex behaviors of the components (for example, success rate to turn on, and the behavior of 'reconfiguration'), but does not provide the ability to integrate repair trajectory requirements.

This is why, the motivation of this research is to provide a unified approach for dependability assessment of structured systems taking into account trajectory requirements. Several formal modeling and transformation steps are advocated, handling successively the system's structure and behavior.

The first step consists of a macroscopic assessment relying on the structure of the system at hand. The aim of this step is to identify how local component faults may impact on a possible global breakdown. Fault tree Analysis (FtA) is applied for the structural analysis and produces a failure statement for the global system, under the form of a logic expression. This expression illustrates the global breakdown configurations of the system at hand.

The failure of several certain components is necessary enough to cause the whole system broken down, which issues the minimal cut set. The combination of all the minimal cut sets constitutes the failure logic expression of this system.

The second step aims at relating exhaustively the global breakdown configurations to the component behaviors and local breakdowns which may collectively lead to a global breakdown. Each component behavior is modeled formally as a state based discrete event system. The global system behavior is obtained formally by combining the local behaviors of all components, by applying a state of the art composition operation. The global model obtained contains all the local component behaviors, and is able to exhibit any global breakdown configuration.

At this point, specific services such as repair requirements can be integrated. These are defined as behavioral models, possibly describing desired repair sequences or priorities, or other operational behaviors. The third step produces a global model which features these resulting behaviors in addition to the preceding one.

The fourth step aims at targeting global breakdown configurations in the behavioral model: the global breakdown statement is formally related to the system behavior, by an appropriate labeling procedure. The resulting labeled behavioral model features all the possible scenarios, containing local breakdowns and leading to a global breakdown.

Face to operational dependability, four objectives can be tackled:

1. Operational Reliability, by assuming no disturbance at the initial state and the accessibility to a total breakdown state, while including specific requirements for service or recovery;

2. Operational Maintainability, by considering that the system is initially totally broken down and the accessibility to a system healthily working state, including specific requirements for service or

recovery;

3. Operational Availability, by considering that the system is initially working, possibly experiencing failures and recoveries and including specific requirements for service or recovery;

4. Operational Safety, by providing a system design which implies no harmful situations for its human environment, while including specific requirements for service or recovery;

This work only focuses on Operational Reliability. Thus, each component can be reasonably assumed to "repairable". Thus, breakdowns may be followed by repairs, with one exception for reliability: there is no possibility to recover from a total breakdown. In order to express this particularity, no repairs should be allowed in the global breakdown configurations. This requires an additional transformation of the labeled behavioral model, by making definitive all the global breakdown configurations.

The resulting system features event-triggered trajectories leading to a definitive global breakdown. These events are in general stochastically described by rates; hence a continuous-time Markov Chain is derived from the labeled behavioral model and offers the usual computations for reliability assessment.

Alternatively, beyond computing failure probabilities, it appears to be interesting to evaluate the effectiveness of the repair management policy. The framework presented in this work also proposes this feature, through quantitative cost simulation, relying on the event and/or global configuration costs.

In conclusion, this work provides a formal modeling framework, based on modular behavioral generation, able to assess the operational dependability of the system, by taking into account specific service or repair specifications and by evaluating their effectiveness. A quantitative analysis methodology is developed, for the calculation of the system capacity, taking into account the components failures.

Chapter 1
State of the art

This chapter contains two parts: First, the context of this research; Second, the statement of the relative works.

Contents

1.1 Introduction

Industrial systems are expected to deliver a global service, and are designed from several interacting and possibly distributed components. The interactions between these components amount to either information, or material interdependencies. Each individual component is subject to failures and some components may be repaired. The impact of one component failure on the global service delivery may vary according to the interdependencies relative to this component. Thus, a component failure is not always a synonym of total breakdown with a definitive loss of the global service.

The guarantee of service continuity is generally critical in any industrial context. In order to provide such a guarantee, the failures of individual components and their possible impacts on the service delivery need to be assessed, and appropriate operations anticipated. Thus, the engineering of the global service needs to be completed by the modeling and evaluation of its reliability, availability and maintainability, issued from the knowledge provided by the failure assessment.

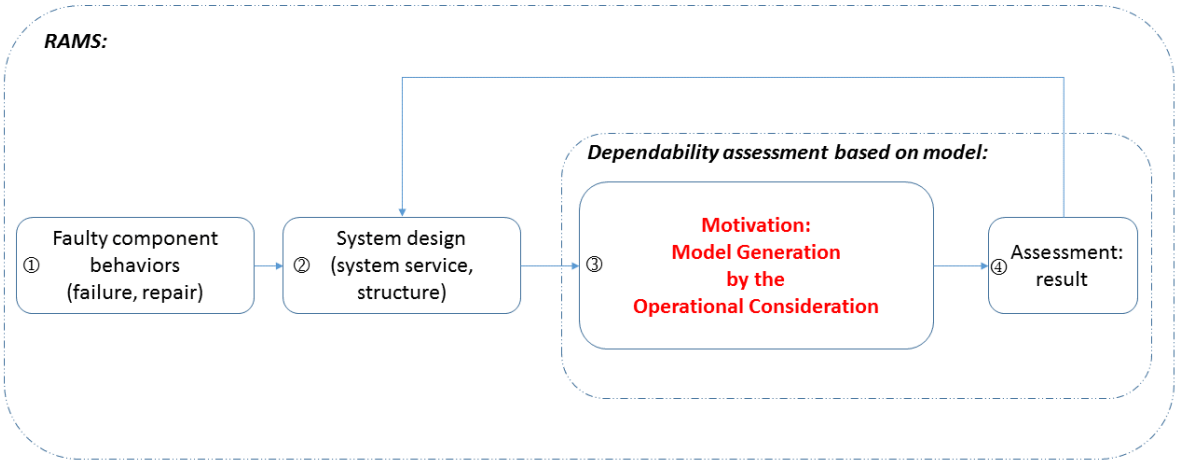


Figure 1 Dependable systems design flow

Figure 1 shows a usual system design flow encompassing both the functional and dysfunctional points of view [8, 9]. The accurate assessment of failure impacts requires several modeling stages:

1. the faulty component modeling step provides qualitative and quantitative knowledge about the faulty behaviors and their probability;
2. the system design step defines the structure and behavior of underlying components, as well as their interdependencies. The resulting system model may also highlight the behavioral impact of component failure on the global service expected;
3. the model-based dependability assessment step relies on both the system and component models. The assessment can be either qualitative or quantitative, and relies on dedicated formal tools for evaluating the overall continuity of the global service. This step provides both quantitative (mean time to failure or repair, failure probabilities, etc.) and qualitative results (corner-case failure scenarii

and their impact on the service expected).

In this chapter, Section 1.2 introduces the context of this research (① ② ③ in the figure). Section 1.3 introduces the state of the art of this research motivation (④ in the figure), in which the related theory extensions are studied and the typical model generation approaches are stated.

1.2 System Dependability

This section recalls the terms and concepts related to model based safety assessment. The Reliability, Availability and Maintainability (RAM) concepts, as well as their mathematical background (followed by the calculation method of time indicators) are presented in 1.2.1. The perimeter of the dependability notion used throughout this work is stated in 1.2.2.

1.2.1 Faulty component behaviors

In the context of possible fault occurrences, specific fault-related behaviors need to be considered. The Reliability is the ability of a system to continuously provide its expected service without breaking. The Availability denotes the system readiness for providing its expected service. The Maintainability denotes the ability of a system to be either maintained or repaired. Behind these intuitive and qualitative expressions, a mathematical framework exists providing their accurate definition and calculation. These are recalled in the sequel.

Assuming design and operational hypothesis, any system can be seen as one single component which can break and be repaired. Failures occur with an average rate denoted by 'λ', and repairs occur with an average rate denoted by 'μ'. The system can be seen as featuring two states: a working state, denoted by S0, and the failed state, denoted by S1. Figure 2 represents the Continuous Time Markov Chain (CTMC) models of the three dependability notions stated above for the system at hand.

In a Reliability model, the system is assumed to be initially functional. The initial state is the working state S0. It may fail with a rate of 'λ', and follow the transition to the failed state S1. This model offers the basis for the calculation of Mean Time to Failure indicators.

In the Maintainability model, the system is assumed to be initially broken, hence the initial state is S1. The system can be repaired with an average rate of 'μ', returning to the working state S0. This model offers the basis for the calculation of Mean Time to Repair indicators.

In the Availability model, the system is assumed to be initially functional, and may fail and be repaired recurrently. Hence, the initial state is S0. Failures occur with a rate of 'λ'. On a failure occurrence, the system model switches to state S1. Repairs occur with a rate of 'μ', and the system recovers its functional state S0. This model offers the basis for the calculation of Mean Time between

Failures indicators. Hence, these three models are representative for calculating the most conventional failure/repair mean time indicators[10-12].

Such representations focus essentially on the dysfunctional aspect of a system. Still, the model can represent both the system behavior in its extensive complexity, through various operating or idle states triggered by various nominal operation events, and its failure state, together with the specific failure/repair events. The initial state assumption defines the assessment framework: reliability, maintainability or availability. This is illustrated in Figure 2. Events {'α', 'β'} related to a nominal behavior of the system coexist with the failure/repair events {' λ',' μ'}. States S0 and S1 model the idle and working system configurations, whereas state S2 models the breakdown.

Reliability, Maintainability and Availability can be assessed on these models by adequately setting the initial state: S0 for Reliability and Availability (Figures 2d and 2f), and S2 for Maintainability (Figure 2e).

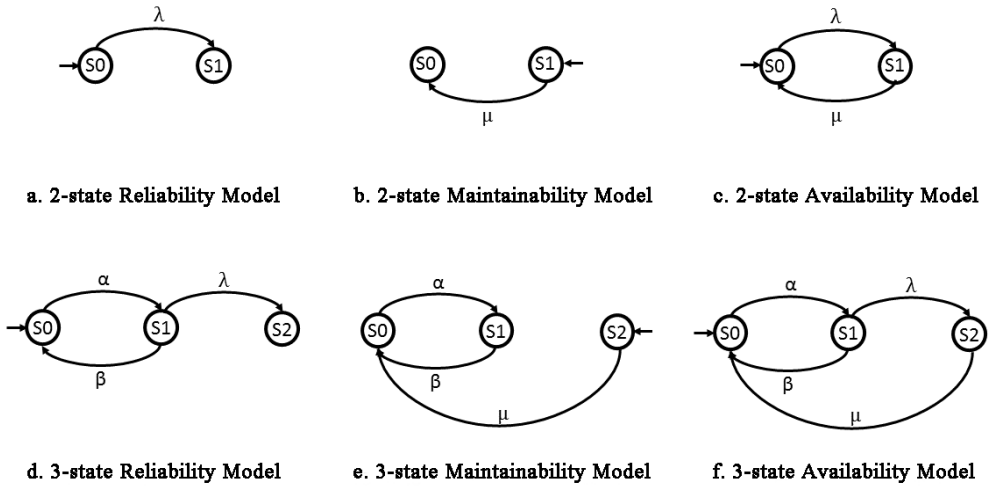


Figure 2 RAM Markov chain model

1.2.2 System design: service and structure

The complexity management of an industrial system meant to provide service calls for advanced engineering skills and resources, at several levels:

1. system design: definition of the functional, logical and physical architecture through requirement analysis, functional analysis and decomposition, logical and physical component mapping;
2. system development: for each component, either design its behavior, or source it externally, verify its correction, alone, and in progressive inclusion with the rest of the system;
3. system deployment: progressively set-up the system, according to a deployment plan, in consistent interaction with the ongoing activity[13];
4. system operation and life-cycle management: this encompasses predictive and curative maintenance.

These milestones appear to be sequential, but are strongly interdependent: the system operation calls for an acceptable and most of all *qualified* level of reliability, availability and maintainability (RAM). These aspects are orthogonal to the system service, they do not concern its nature but its continuity, which is why they may be easily overlooked.

A dysfunctional analysis phase [14, 15] can be lead in parallel with the system development phase, and often induces costly iterations in order to redesign components, due to the late identification of critical RAM aspects. The early integration and prediction of reliability, availability and maintainability features into the design flow [16, 17] is more recently advocated in order to consider dysfunctional aspects as soon as possible and hence, minimize the overhead of such redesign iterations.

Hence, a service provided by an industrial system universally relies on the interaction of services of its underlying components[18, 19]. These interactions materialize flows having the morphology of cascades, derivations, cycles, or a combination of these patterns. These patterns are identified as ***structures***, which can be either series, or parallel, or mixed. In a RAM analysis approach, a set of interacting services must be mapped to such a structure. Most often, this preliminary step abstracts away the service nature and only focuses on the interdependence[20]. For instance, in a series structure, regardless of its nature, a global service requires that all local services be operating. In a parallel structure, the global service requires that at least one local service be operating. It is assumed, without loss of generality, that component failures are never simultaneous.

These aspects are summarized in Figure 3. Each system has a structure, materialized recursively by components and a specific interconnection. The specification of the interconnection between components is mandatory, as it determines qualitatively the RAM properties of the whole system. Hence, as shown in Figure 3, the system service amounts to processing an input flow, identified as the "source", and produce an intended output, identified as the "target". The source can consist of either physical materials, or primary resources, or machinery, or energy, or finance, etc. And the target should be an expected result, either material or immaterial.

Besides, the system structure also determines the dysfunctional behaviors that need to be anticipated. The global system behavioral model features a collection of configurations which are identifiable as *able to work*, or simply *working* configurations on one hand, and breakdown configurations, named *total breakdown states*, together with all the possible transitions between these configurations. It is interesting to notice that this model is exhaustive: it represents all the possible sequences compliant with its components' behaviors. This makes it possible to reason about specific sequences expressing desired (realizable) behaviors, such as maintenance policies, prioritization, etc. Such requirements can be expressed by pruning inadequate sequences from system model, which is extremely error-prone. A separation between system and requirement specifications is of great interest.

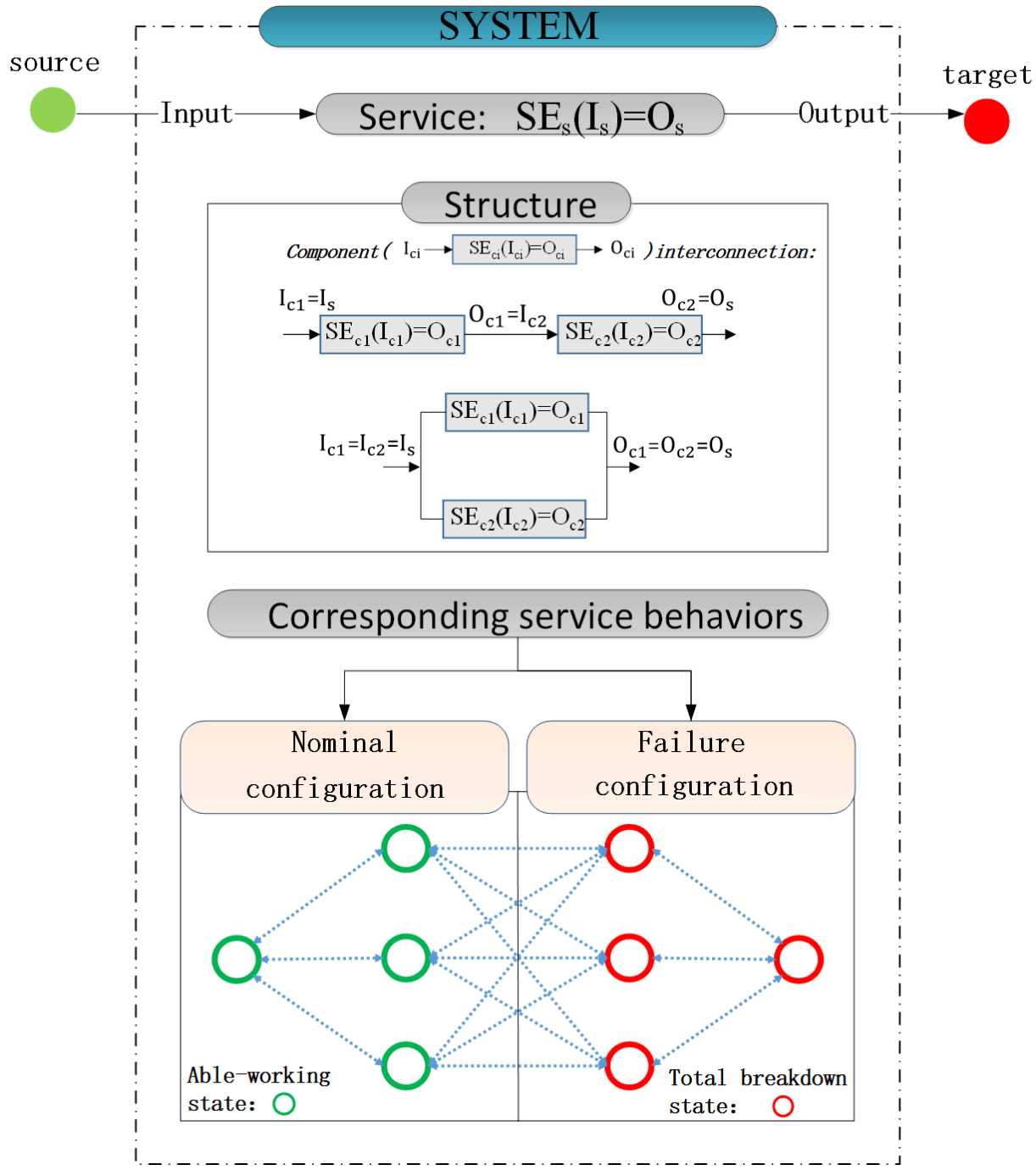


Figure 3 system: structure and service

Hence, in order to perform a service, underlying components provide different services and form a structure which can be complex[21-23]. A component supposed to deliver its service, by transforming input I_{ci} into output O_{ci} is denoted as Ci. Its service function is SE_{C_i}:

$$SE_{ci}(I_{ci})=O_{ci},$$

The behavioral models highlighted in Figure 2 provide a high abstraction level, which makes them representative for virtually any service: states such as “working”, “idle” and “breakdown” can always

be identified both in a system or in its underlying components. Thus, a service SE_{ci} operates in its "working" state of the underlying state-transition model of Ci. In this state, SE_{Ci} is nominally provided.

A system S is constructed by interconnecting its underlying services. For a system S providing a service SE_s relying on components $\{C_1, C_2, \dots, C_N\}$, this is achieved by adequately mapping service outputs O_{cj} of component C_j to the corresponding input I_{ci} of component C_i according to the desired interconnection.

Probing the availability of a system at a given moment is not always obvious. Yet, it is assumed in this document that when a system is broken down, it ceases to deliver its service. Its output is undefined, denoted ⊥. Hence, all components are considered fail-silent. Thus, a service produces an expected output, if it is operational, and ⊥ if it is broken down:

$$SE_{C_i}(I_{C_I}) = \begin{cases} O_{C_i}\ if\ C_i\ is\ operational \\ \perp\ if\ C_i\ is\ broken\ down \end{cases}$$

In the sequel, for consistency reasons, the following conditions should be assumed:

1. For any component C, $SE_C(\perp) = \perp$
2. For any component C, if C is broken down, then $SE_C(I_C) = \perp$. The reciprocal is not true: the output of a component can be undefined because of a component breakdown in its transitive fan-in.
3. For a collection of k flows $F = \{O_1, O_2, \dots, O_k\}$ it is useful to denote that:

$$F = \perp\ iff\ \forall i = 1 \dots k: O_i = \perp$$

Based on the two basic system structures, series and parallel, various complex structures can be built, yielding either well-known architectures, such as the cold-redundancy structure, the warm redundancy structure, the vote structure, the mixed structure, or more typical structures.

Thanks to the system representation as a structure of components, the global RAM features can be formalized and assessed in terms of the RAM of its components.

Hence, the notion of total breakdown can be defined in terms of local components' breakdowns and the system structure. The same holds for the system reliability, availability, and maintainability, which can be assessed globally, based on its structure and the behavioral model of each component.

The following paragraphs recall several conventional system structures [10, 24-27].

A. The series structure

This structure features a system S made of a set of N components $C_1, C_2, \dots, C_N$, fed by the source flow according to the following interconnection:

- $I_1 = source$;

- $\forall\, i \in 2 \ldots n :\ I_i = O_{1-1}$;
- $target = O_N$.

Figure 4 series structure system. The system' service results from the cooperating service of all components, where not a single component may fail.

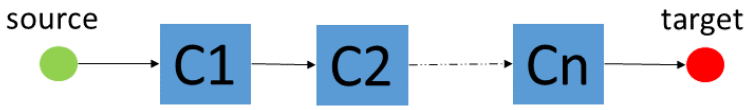


Figure 4 series structure system

The output of S is either the output of C_N or $\perp$. For the series structure, we have

$$SE_S \neq \perp \ \ iff \ \bigwedge_{i=1}^{N} SE_{C_i} \neq \perp$$

The service of S cannot be provided if at least one of the local components is not able to contribute its own service.

B. The parallel structure

To enhance the availability of the system's service or to enhance its production, several similar components are distributed to work side by side sharing the same input and producing the same output. On the view of the dependability, this designing method empowers the system by the ability of fault tolerant.

This structure features a system S made of a set of N components $C_1, C_2, \ldots, C_N$, fed by the source flow according to the following interconnection:

- $\forall i = 1 \ldots N :\ I_i = source$;
- $\forall\, i \in 1 \ldots n :\ O_i = SE(\, I_1)$;
- $target \in 2_1^{\{O_1, O_2, \ldots O_N\}}$.

Figure 5 illustrates the service of this system. To realize its service, its structure is constructed by n components denoted by C1, C2 …Cn. These components share the same input and this input is also the system input (marked by the 'source' in the figure): and they contribute to the same output (marked by the 'target' in the figure) as to be the output of the system [21-23]

The components working in the parallel structure system, often provide the same service, treating the same input and producing the same or similar outputs. However, the capacity of the components'

service may differ (shared load). So, the parallel structure can be considered as a redundancy-based design, where the components may have different service capacities. If some components fail, the system partially loses its capacity, but is still able to deliver its service function. Thus, in case that at least one of these components is able to work, the whole system is able to provide its function. In case that all the components lose their functions, this system is totally broken down.

Thus, for the parallel structure, it can be stated that:

$$SE_S = \perp \ iff \ \bigwedge_{i=1}^{N} SE_{C_i} = \perp \ OR \ (source = \perp)$$

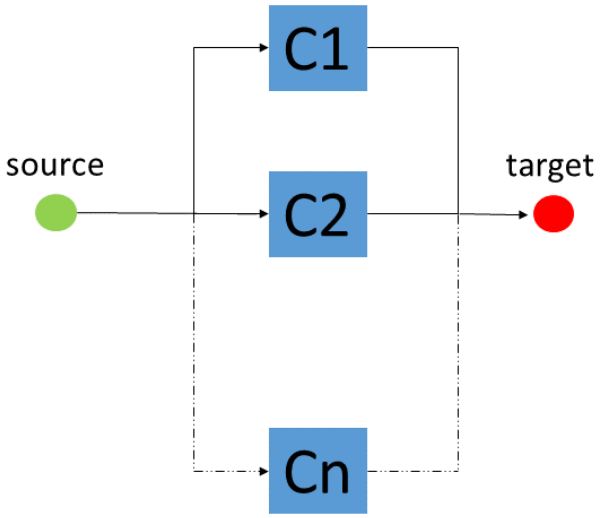


Figure 5 Parallel structure system

The service of this system cannot be provided if its input is undefined or if all of its components are unable to provide their service, because they are broken down.

Based on the equation above, the more components collaborating in the parallel structure system, the stronger ability to deliver the global service is. According to this, one method to enhance the system reliability is to set a redundant component (to build a parallel structure), assisting the major working components[28], which will be further introduced in the following parts.

C. The cold-redundancy structure

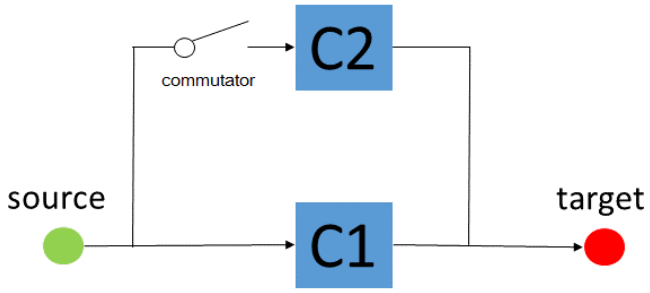


Figure 6 Cold redundancy structure

The structures recalled above can be combined and adapted, in order to obtain additional, possibly enhanced reliability features. Thus, in order to guarantee the availability of the system's service and empower its the fault tolerance, besides a **main** operating component there is an equivalent **reserve** component, ready for operation, and which can be triggered in case the main component breaks down.

Figure 6 illustrates this structure. It is generally assumed that the switch is fault-free. Yet, the non-commuting probability is not zero, related to a possible lack of switching command. The services of C_1 and C_2 are considered equivalent. They share the same input and contribute to the same output. Component C_1 is the main component: $I_s=I_{c1}$ and $O_s=O_{c1}$. If C_1breaks down, component C_2 is switched on and starts to work instead of C_1 : $I_s=I_{c2}$ and $O_s=O_{c2}$. This structure and use is known as cold redundancy. The service provided SE_S is expressed in the same way as for the parallel structure.

D. The warm redundancy structure

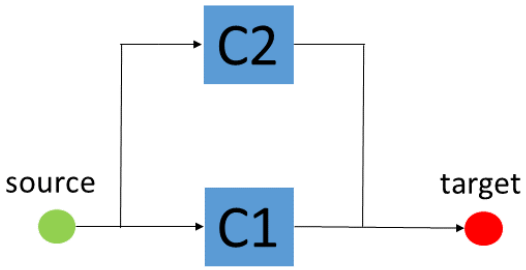


Figure 7 Warm redundancy structure

Simarly but not the same with the parallel structure, a redundancy structure always features an accompanying component working beside with a main component, and these two components have the same service function with the same input and output. However, the production capacity of the accompanying component is often weaker than the main working component. With respect to the parallel structure, the warm redundancy structure is rather operational: enhance the service availability till the main component is repaired or replaced.

Figure 7 illustrates a warm redundancy structure featuring a main component C_1 and a redundancy component C_2. C_1 and C_2 are working together, while the service treats the same input (from the source)

and they produce the same output (for the target). The service provided SE_S is expressed in the same way as for the parallel structure.

E. The voting structure

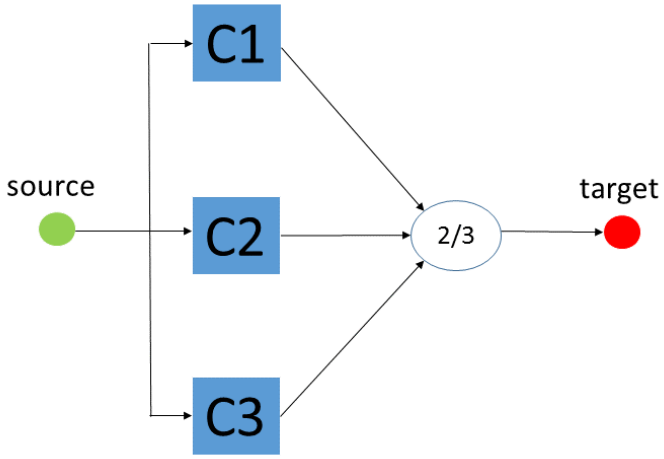


Figure 8 Vote structure system

The voting structure system is made of three or more components which are expected to provide identical service but which can differ in terms of either design, manufacturing, or sourcing. With the assumption that the voting engine is considered to be always perfect, the output of this system amounts to a decision made by voting engine: each component asserts its own output; the output asserted and shared majority between the components shall be the output of the system. In practice, voting structures contain three decision making components, as shown in Figure 8, which is the minimum in order to guarantee the majority.

Thus, components C1, C2 and C3 receive the same input. Their output is a decision and is submitted to voter which achieves the actual voting. If at least two components are healthily working, the system is able to deliver its service. It is important to note that in a voting architecture, components C_1, C_2, C_3 are not supposed to be fault-silent. A fault is rather synonym of a "wrong" decision, rather than an undefined output. The structure is fault tolerant, as long as the three components produce outputs. Hence, the service breakdown of this system is defined as:

$$SE_S = \perp \ iff \ \bigvee_{i=1}^{N} SE_{C_i} = \perp \ OR \ (source = \perp)$$

A breakdown occurs if at least one component is broken down.

F. The series-parallel structure

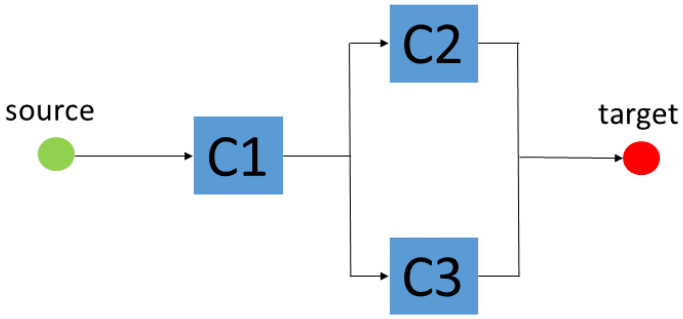


Figure 9 Series-Parallel Structure

The basic and regular structures recalled above can be combined in order to achieve more complex architectures, as shown in Figure 9. Such structures can use single components, such as C_1, which is used alone because it is considered very reliable, or simply very expensive, and also other components operating in redundancy, in order to provide higher performance, or reliability, or both. These structures are recursively analyzed, by identifying their subcomponents with simple structures (series, parallel, …).

G. Mixed Structure

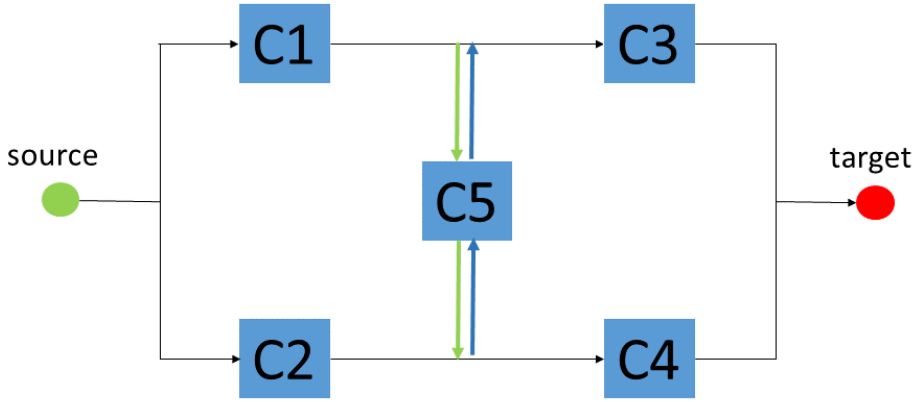


Figure 10 Mixed Structure

More complex flows can be defined and compose a mixed structure. The 'mixed' feature emphasizes that there may exist different interactions among the components and that these interactions amount to various superposed structures.

The example shown in Figure 10 features components C1 and C2 which share the input (source in the figure) of the system : $I_s=I_{C1}$, $I_s=I_{C2}$; components C3 and C4 both contribute to the output of the system(target in the figure) : $O_s=O_{C3}$, $O_s=O_{C4}$; component C3 treats the output of C1 : $O_{C1}=I_{C3}$; component C4 treats the output of C2 : $O_{C2}=I_{C4}$. However, component features a cyclic dependency: it processes the output of C1 and outputs its result to C4: $I_{C5}=O_{C1}$, $O_{C5}=I_{C4}$, and in the meantime it also

processes the output of C2 and outputs its result to C3: $I_{C5}=O_{C2}$, $O_{C5}=I_{C3}$. Again, these dependencies have a decisive impact on the dysfunctional behavior of this structure.

1.3 Model based dependability assessment: state of the art

Discrete Event Systems (DES) offer a helpful abstraction level for modeling and assessing system dependability. They feature *transitions* which can be driven by either probabilities, rates or events, are which can be used to model failure behaviors. Discrete states are used to represent the possible configurations of the system. Failure scenarii are represented by sequences of transitions between identified initial state and a pre-designed failure configuration. They provide qualitative information about failure occurrences. If quantitative figures are available, modeling consistently event-related probabilities or rates, quantitative dependability indicators can be further calculated. The discrete event system modeling approaches, such as Stochastic Petri nets and Markov chains are well suitable for this modeling. DES models are usually modeled by hand. Yet, in order to encompass the potential complexity due to the growing size of a system, the ability of systematic generation of global, complex reliability models from basic ones appears to be vital.

In order to discuss the Discrete Event System model generation approaches, the ordinary dependability assessment methodologies are briefly recalled in the following. Hence, it is worth being recalled that DES models provide a behavioral point of view of the system at hand, which help in both identifying breakdown states and numerically assessing the reliability. These are possibly used in synergy with models and methods representing structure and hierarchy such as the Fault tree Analysis (FtA) and Reliability Block Diagram (RBD) analysis. FtA is a top-down tree logic modeling method. The leaves which stand for the failure of the components are connected by the logic gates (‘and’ gate for parallel structure, ‘or’ gate for series structure, etc), and the result is evaluated on the top of the fault tree [29]. In RBD, blocks are used to present the components, and the blocks are connected with the series network and parallel network. The detailed methodology of FtA and RBD is studied in a subsequent section dedicated to GRIF.

In the sequel, contributions of the Discrete Event Systems domain to the Model Based Safety Assessment are stated, in section 1.3.1, and focused on MBSA generation approaches, in section 1.3.2.

1.3.1 Discrete Event System Theory Contribution to MBSA

Two main modeling approaches are recalled: modeling token flows using Stochastic Petri Nets (SPN) and modeling sequences using Continuous Time Markov Chain (CTMC). In order to compute the ordinary time indicators, four subsequent operations need to be recalled:

- model composition, in order to handle component-based systems and their inherent complexity;

- structural analysis, in order to determine the failure logic;
- state combination, in order to reduce the model size by regrouping states according to a quantitative rate equivalence notion;
- CTMC to stochastic Petri net transformation. Stochastic Petri nets own the advantage of modeling traceability.

The objective for inserting the DES theory into MBSA are based on its main abilities for both model generation and proof. Generation amounts to automatically delivering of a model thanks to composition. This ability is intensively used when handling complexity. The proof amounts to validating sequence and structure through properties statements such as observability and accessibility to specific states such as TBS's.These elements are recalled in the next two sections.

A. Continuous Time Markov Chains for system assessment

This part recalls the reliability Markov chain modeling methodology. An example is used to explain how to calculate the time indicator MTTF (Mean time to Failure). Behavioral and structural modeling are illustrated, together with the behavioral model composition.

CTMC modeling principle based on the system structure

Named after the Russian mathematician Andrey Markov (1856-1922), Markov chains model stochastic processes by expressing quantitatively transition relations between the states of the system. The Markov stochastic processes are memoryless: the next state only depends on the current state regardless of any past evolution [30, 31]. The Continuous Time Markov Chains (CTMC) express state transitions by relying on event occurrence rates [32, 33]. For instance, the failure rate represents the average number of failure occurrences by time unit.

For a given system satisfying the Markov memoryless property, a CTMC model is achievable and thus, an equivalent transition matrix can be derived, which allow the calculation of time indicators [10, 34-37].

As depicted in section 1.2.2, beyond the individual RAM indicators of each component, the system structure also has an impact on the global MBSA results. The expected results for MBSA would be the possibility to discuss on a better construction of service and/or better specification for the failed system face to the repair management.

Behind each system structure there is a static and a dynamic perception. The structure in itself is a static statement, unable to express the possible scenarii leading to partial and/or total breakdowns. Such scenarii are expected to express the actual run of a sequence of configurations, leading to a particular situation, such as a breakdown. This is a modeling gap which is conveniently filled by the CTMC modeling, in order to express the global dysfunctional behavior from individual ones. The behavioral model abstracts away most functional states and focuses on the dysfunctional aspects: component models usually feature two, maybe three kinds of states expressing inactivity, working and breakdown.

Inactivity and working states are sometimes grouped into a single state expressing the absence of breakdown.

Example

Figure represents two possible structures achievable from two individual components, C1 and C2, modeled by subfigures 12(a) and 12(b). Each component presents two states: a working state, denoted respectively by W1 and W2, and a brokendown state denoted respectively by B1 and B2. For i=1..2, component Ci can break down with a failure rate λi. Each component can be repaired by a service engineer, and the repair average rates are denoted μi.

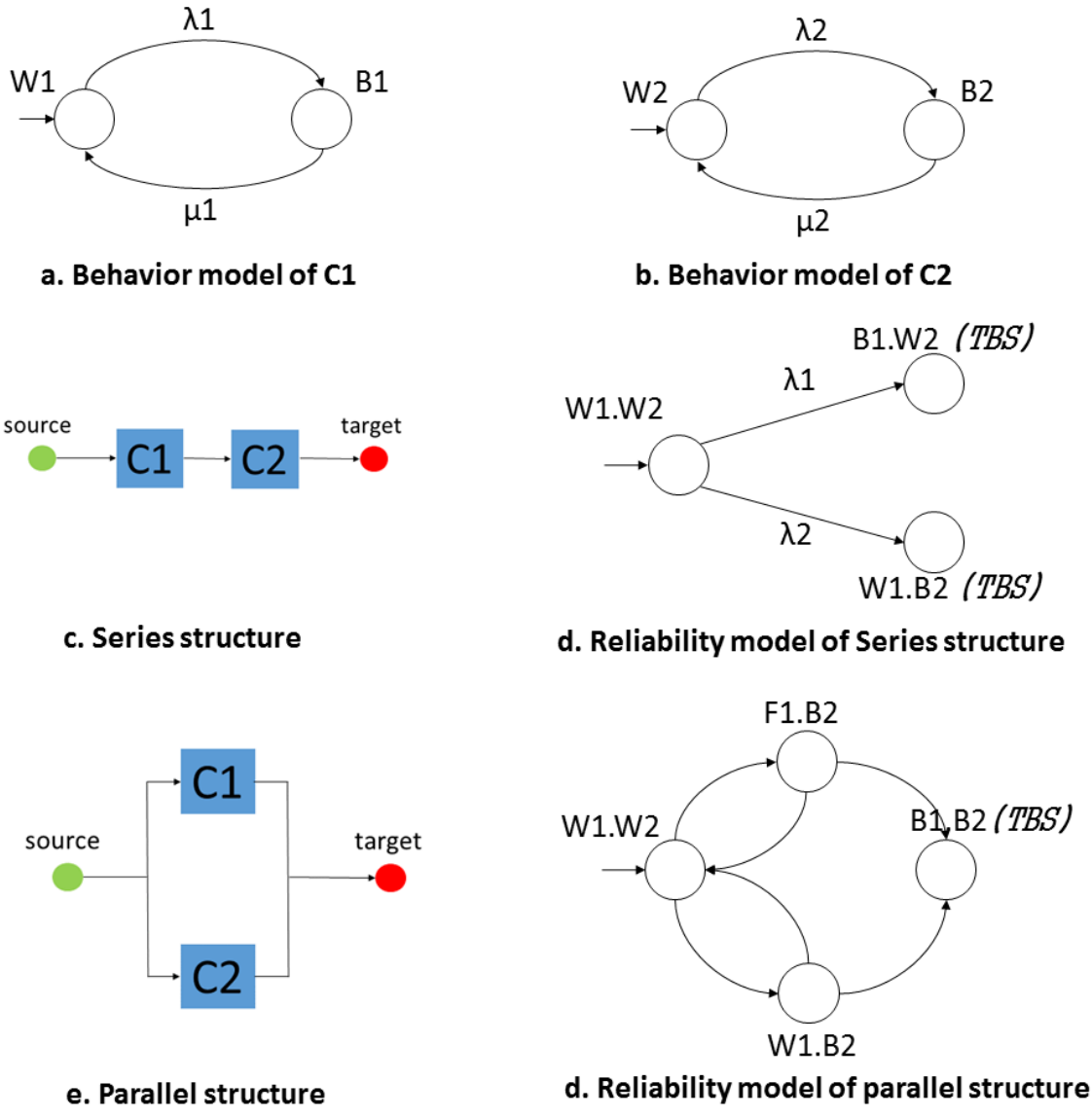


Figure 11 Basic reliability CTMC models from structure definitions

Thus the system features globally four possible states: 'W1.W2' meaning C1 and C2 and both working; 'W1.B2' meaning C1 is working but C2 is broken down; 'B1.W2'meaning C1 is broken down

however C2 is working; ‘B1.B2’meaning both C1 and C2 are broken down. And, the evolution of the system states are driven by the failure rates and the repair rates of the components (λi and μi).

A series structure can be built by connecting the output of C1 with the input of C2, as shown in Figure 11.c. This interconnection requires that both components are operational: if either component breaks down, the global system service cannot be achieved anymore. Similarly, if C1 and C2 could share the same inputs and feed the same output, a parallel structure can be built, as shown in Figure 11.e.

The reliability CTMC model of the series structure is represented in Figure 11.d. This model represents the possible scenarii leading to the global, total breakdown, as defined in section 1.2.2.

It can be observed that only when these two components are operational, i.e in their working state Wi, the system is globally operational. If one component breaks down, the whole system is broken down. So, the system states associations ‘W1.B2’, ‘B1.W2’ and ‘B1.B2’ are the system’s Total Breakdown States (abbreviated TBS in the figure).

The reliability CTMC model of the parallel structure is presented in Figure 11.f. This model represents the possible scenario leading to the global, total breakdown.

It can be observed that only when components C1 and C2 are both broken down, the system is totally broken down. So, the system state association B1.B2 represents the TBS.

Notice that the behavioral models obtained here are meant to assess **reliability**. Hence, they do not entirely feature all the possible behaviors. In theory, it should be possible to leave the TBS by repairing either one or both components, according to their local configuration. Still, in order to assess reliability, the global model should exclusively feature the scenarii leading to the total breakdown. As shown in Figure .c, for the parallel structure, repair actions can occur arbitrarily often, but only as long as the system is globally available.

Hence, reliability CTMC models are usually manually established from the system structure and from the knowledge about the individual component dysfunctional behaviors. This process requires a high level of expertise, as it is done manually, as advocated by[7, 38]. It can be apprehended for medium sized systems, but becomes error-prone as soon as the structure gets complex. The ability of systematically compose CTMC models appears to be fundamental in order to handle this complexity.

CTMC composition

This operation builds a global behavioral system model out its components. It is based on the theory of interactive Markov chain[39]. Ordinary CTMC models express event occurrences quantitatively, as rates. The notion of interactivity goes back up to events, and the possibility that the same event can impact the dynamics of more than one model. Besides that, component states are regrouped into global states

according to a Cartesian product approach. The transitions of the global model reflect all transitions of the underlying components. They reflect two kind of situations:

- asynchronous evolution, if a transition only occurs locally in a component CTMC model;
- synchronized evolution, if the transition is shared between two or mode component CTMC models, through the same event [39-41].

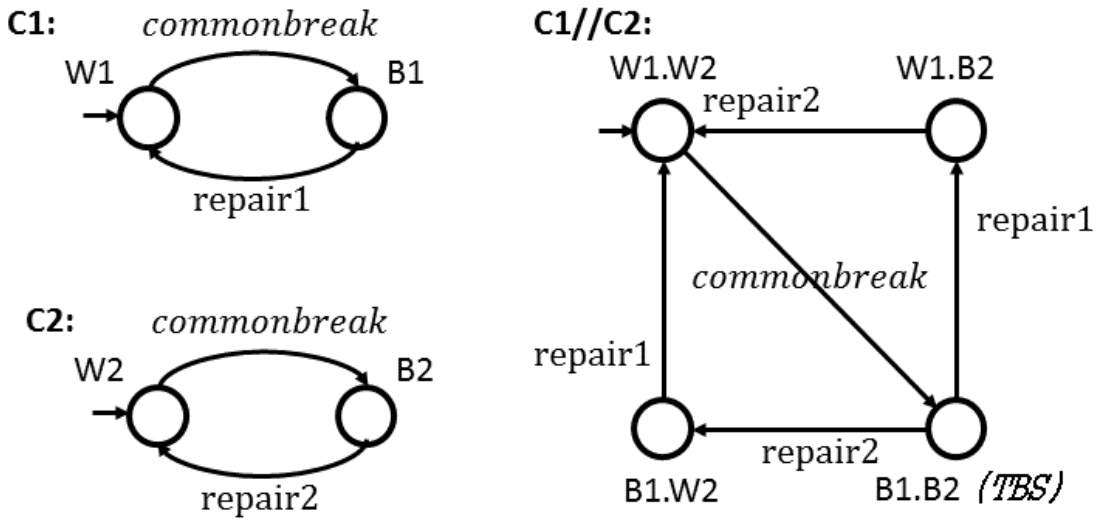


Figure 12 composition operation of interactive Markov chain

Example

Figure 12 shows a system built from two components C_1 and C_2. States W1 and W2 model working configurations, B1and B2 model broken down configurations. It is assumed that failure and breakdown events cannot be simultaneous. The model expresses the fact that components C_1 and C_2 are somehow interdependent, and that the same fault causes their common breakdown. This is denoted by label 'commonbreak' in the model. But each component can be repaired independently. The equivalent behavioral model is shown in Figure 2, denoted C1//C2. This model represents the whole system behaviors. The transition "repair1" and "repair2" are asynchronous. The transition "commonbreak" is taken synchronously by the two components.

State combination in CTMCs

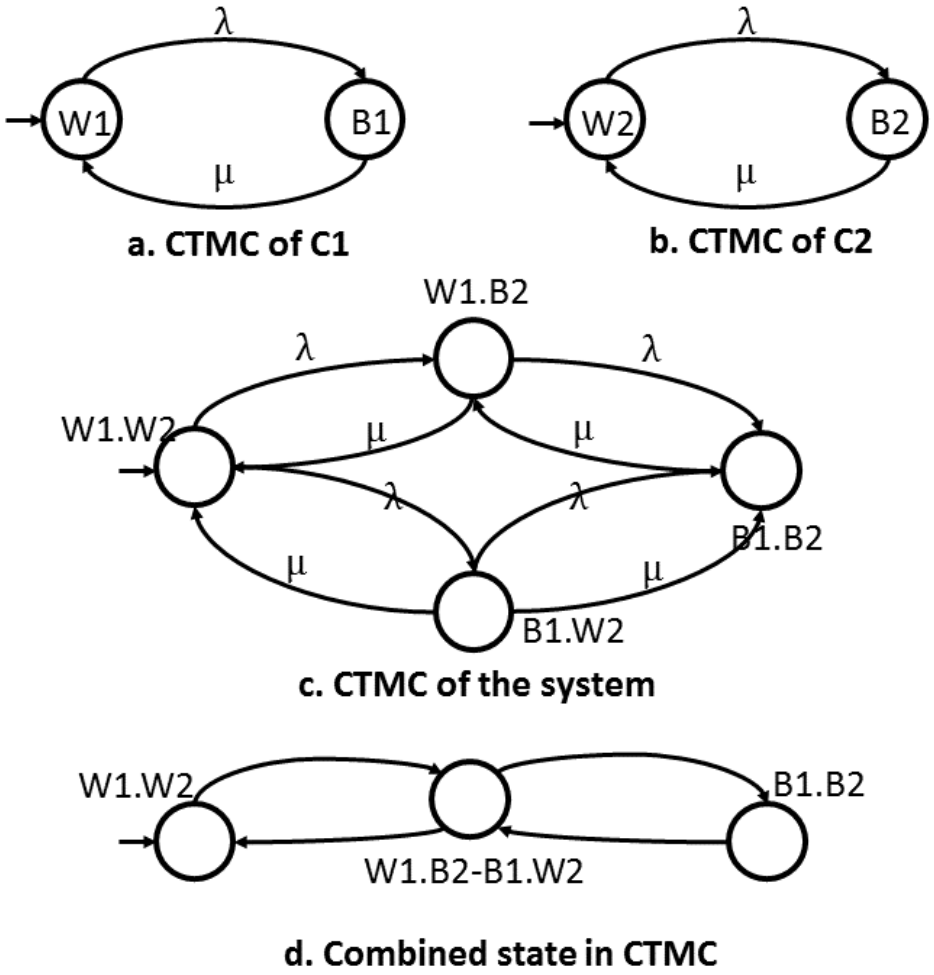


Figure 13 State combination

As CTMCs are quantitative models, it is sometimes possible to regroup states. For a system composed of two failure/repairable components, C_1 and C_2, having both a failure rate of "λ" and a repair rate of "μ". In order to deliver the service of the system, these two components are working independently, which means they do not influence each other. According to Figure 13 c, states B1W2 and W1B2 express respectively the fact that C_1 or $C_2$is broken down. This situation can be simplified, by expressing the fact that either C_1 or C_2 are faulty, using an equivalent state W1.B2-B1.W2 in Figure 13.d.

According to[42-44], transitions from state *x* to state *y* are fired with a probability $P_{x,y}$. Let *A* and *d* be CTMC states, and $d_1, d_2, \ldots d_n$ be states to be combined into *d*. In the resulting state-combined CTMC' the transition probabilities are calculated as follows:

$$P_{A,d} = P_{A,d1} + P_{A,d2} + \cdots + P_{A,dn}$$

$$P_{d,A} = (P_{A,d1}P_{d1,A} + \cdots + P_{A,dn}P_{dn,A})/P_{A,d}$$

Thus, the transition rate from W1.W2 to W1.B2-B1.W2 is:

$$P_{W1W2,W1B2-B1W2} = P_{W1W2,W1B2} + P_{W1W2,B1W2} = \lambda.dt + \lambda.dt = 2\lambda.\mathrm{dt}$$

The transition from W1.B2-B1.W2 to W1.W2 is:

$$P_{W1B2-B1W2,W1W2} = (P_{B1W2,W1W2}P_{W1W2,B1W2} + P_{W1B2,W1W2}P_{W1W2,W1B2})/$$
$$P_{W1W2,W1B2-B1W2}=(\lambda.dt.\mu.dt+\lambda.dt.\mu.dt)/2\lambda.\mathrm{dt} = \mu.dt$$

And the other transition rates respect the same principle [44-46].

Discussion

In conclusion, Continuous time Markov chain models offer attractive abilities for quantitative and qualitative modeling of component and system behavior. They are able to model sequences, and offer accurate mechanisms for identifying critical components within a given structure. They are useful but not entirely appropriate in order to handle **operational dependability** into MBSA. The shortcoming mainly comes from three points:

- they do not offer a unique modeling framework to natively model structure, component and system behavior;

- they are not systematically able to **specify** MBSA-related *behavioral requirements*. For instance, a repair sequence between two or more components can be preferred. Its expression within the system CTMC model is tedious and error-prone. Still, this is a fundamental requirement for **operational dependability;**

- the state combination and simplification may lead to losing necessary behavioral information and decrease the level of accuracy for the dependability assessment. Indeed, states are necessary for modeling the various system configurations, such as idle, working or failure. Unless components feature identical dysfunctional behaviors, these could by lost through state combination, and reasoning about them would not be possible anymore. Similarly, transitions should not be regrouped, so that it remains possible to distinguish different transition rates in a system or even event sequences. This is why this work advocates not to use this particular simplification step, and in the sequel states shall not be combined. The loss of insight is compensated by the modular modeling methodology, which is able to reduce the modeling workload. The same modular approach is able to naturally integrate operational requirements inside the system model.

The following section recalls the usage of Petri Nets in the MBSA domain.

B. Petri nets for system assessment

Petri net models are equally helpful in dependability assessment[47, 48]. Together with Reliability Block Diagrams, Fault Tree Analysis and Markov chains, stochastic Petri nets (by the creation of

stochastic activity networks) and Timed Petri nets (by real-time faulty systems describing) were applied to dysfunctional system modeling[49, 50]. This approach is further improved by system failure sequence analysis[51] by graphical description [52] by the association of minimal cut sets in the system failure logic expression[53] and by aging tokens creation[54].

This section recalls the dependability assessment solutions using Stochastic Petri nets. The Petri net interconnection operation is recalled first, which can generate the system model based on the component behaviors. The Petri net modeling principle is subsequently recalled. Then, the transformation principles between CTMC and Petri nets are recalled. A discussion concerning Petri Net MBSA is presented at the end of this section.

Interconnection of Stochastic Petri net models

According to the research motivation presented above, the global system behavior needs to be built out of its individual components. Thus, the composition of the individual models in order to generate the system model is a required step. Moreover, additional behavioral specifications need to be modeled and composed into the global result. Petri nets can be composed by using the interconnection operation[55]. As illustrated in Figure left side, a connecting arc is used to link from one place in LPN (Local Petri Net) A to one transition in LPN B. By this method, these two local models are connected to form a global model. Another illustration is shown in Figure right side, an Interaction Petri Net (IPN) is used to connect from one transition in LPN A to two transitions in LPN B. Similarly, Local Generalised Stochastic Petri Nets are composed by modules with interconnection[56]. Subnets of Coloured Petri nets are also able to be composed according to their hierarchy [57]. The Petri nets components composition semantics are created in [58]. In the aboves solutions, tokens are exchanged in order to achieve the composition. Differently, in [55, 56] the composition operation is not impacted by the component tokens.

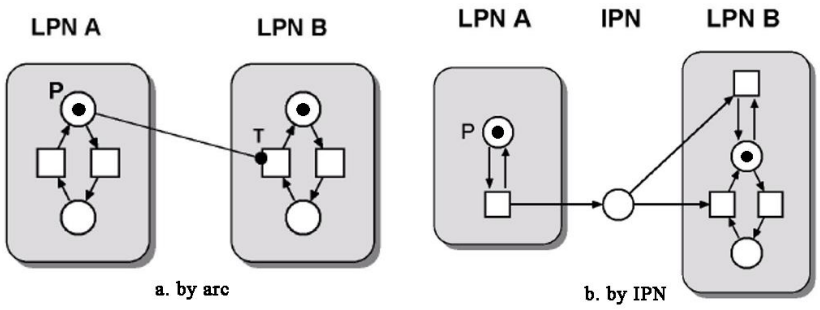


Figure 14 Interaction of Petri Net

Petri net modeling from the system structure

In order to to express the causality induced by the system structure, previous research works have focused on the dependability assessment using Petri nets[59-65]. The structural modeling principle of Petri Net is recalled in this part.

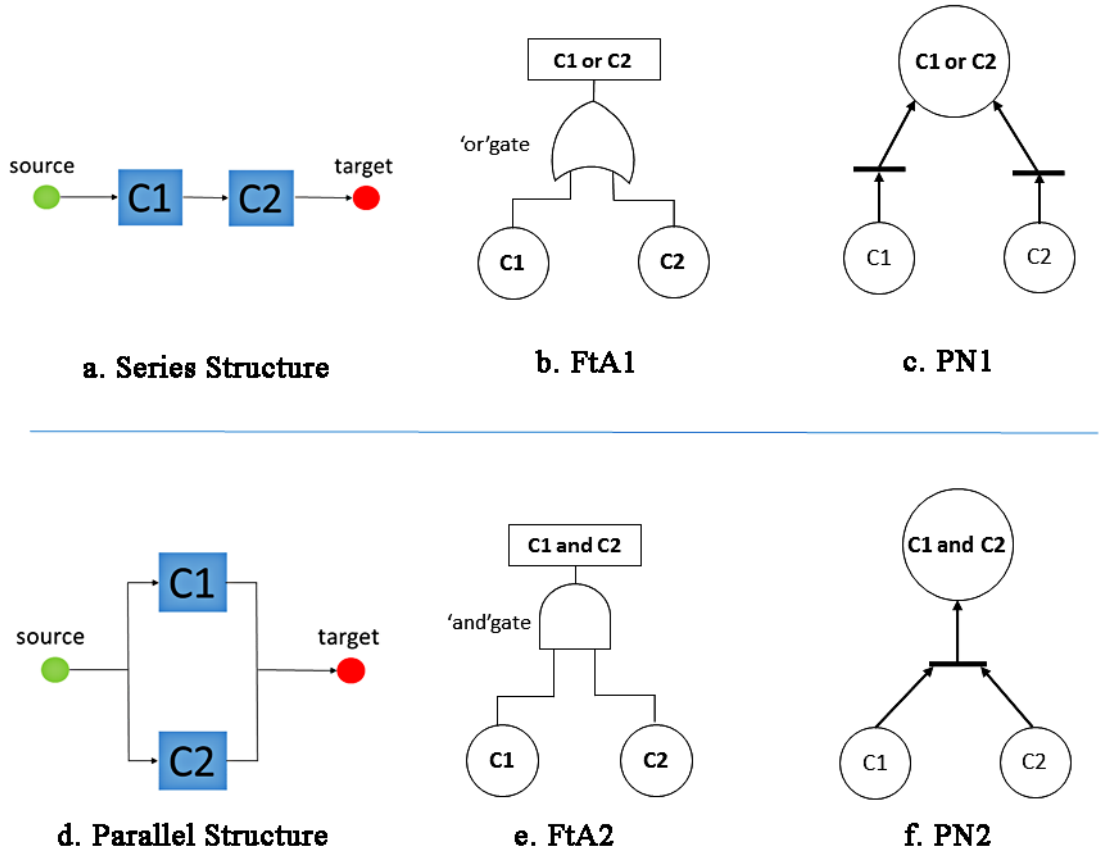


Figure 15 Petri Net model generation based on structure and FtA

Fault tree Analysis (FtA) and Petri nets can be combined in order to generate the system behavioral model. For example, the work of [60, 64, 65] uses Fault tree Analysis transformed into Petri nets and calculate the system indicators by the application of fuzzy Lambda-Tau methodology.

In Figure 15, the upper side presents the Petri net dysfunctional model corresponding to a series structure; the lower side presents the Petri net generation based on parallel structure system. There are two components A and B in the system, and the names 'C1' and 'C2' in the FtA are used to denote the failure state of the components. For the series structure, the 'or' gate FtA is established. And the 'or' gate of the FtA can be transformed into an 'or' Petri net structure (on the right side), expressing the fact that the failure of 'C1' or the failure of 'C2' is able to cause a total breakdown. For the parallel structure, the 'and' gate of the FtA can be transformed into an 'and' transition of Petri net, expressing the fact that the failure of 'C1' and the failure of 'C2' occurring both are able to cause a total breakdown.

Generalized Stochastic Petri nets

Generalized Stochastic Petri nets can be translated into Markov chains, which has the potential application for generating dependability models. A cold-redundancy model is used to illustrate this generation approach.

Example

Figure 16.a represents two components and one repairer. The repairer is only able to repair one component at a time. In case the component C1 is broken down, C2 starts to work instead. It is assumed that component C2 has the priority to be repaired over C1. After all these two components are successfully repaired, the repairer becomes idle.

The Generalized Stochastic Petri nets of components C1 and C2 are shown in

Figure <C1_PN> and <C2_PN>. The edges and places are explained as following:

P0: C1 is operating;	P4: C2 is operating;
P1: C1 is waiting for being repaired;	P5: C2 is waiting for being repaired;
P2: C1 is being repaired;	P6: C2 is being repaired;
P3: repairer is available for C1;	P7: repairer is available for C2;
t0: C1 is broken down;	t3: C2 is broken down;
t1: repairer begins to repair C1;	t4: repairer begins to repair C2;
t2: C1 is successfully repaired;	t5: C2 is successfully repaired;

Based on the component' models, according to the repair principle of the system (C2 has the priority to be first repaired, if these two components are both breakdown), the global model is able to be generated by the model-interconnection. The Generalized Stochastic Petri Net of this system is represented in figure. d System_GSPN.

An example of the global expected behavior shows for instance that in place P2 if the two components break down, C2 has the priority to be repaired.

The Generalized Stochastic Petri can be translated into a Reachability Graph, according to [66-68]. In this example, the approach proposed by [69, 70] is used. The Petri Net Analysis tool 'Tina' performs the reachability analysis.

The generated CTMC is shown in the <System_CTMC> subfigure. The transition labels conserve the same meaning as in <System_GSPN>. The CTMC obtained reflects the initial system' dysfunctional behavior: cold redundancy with two components, one repairer, C2 has the priority to be repaired.

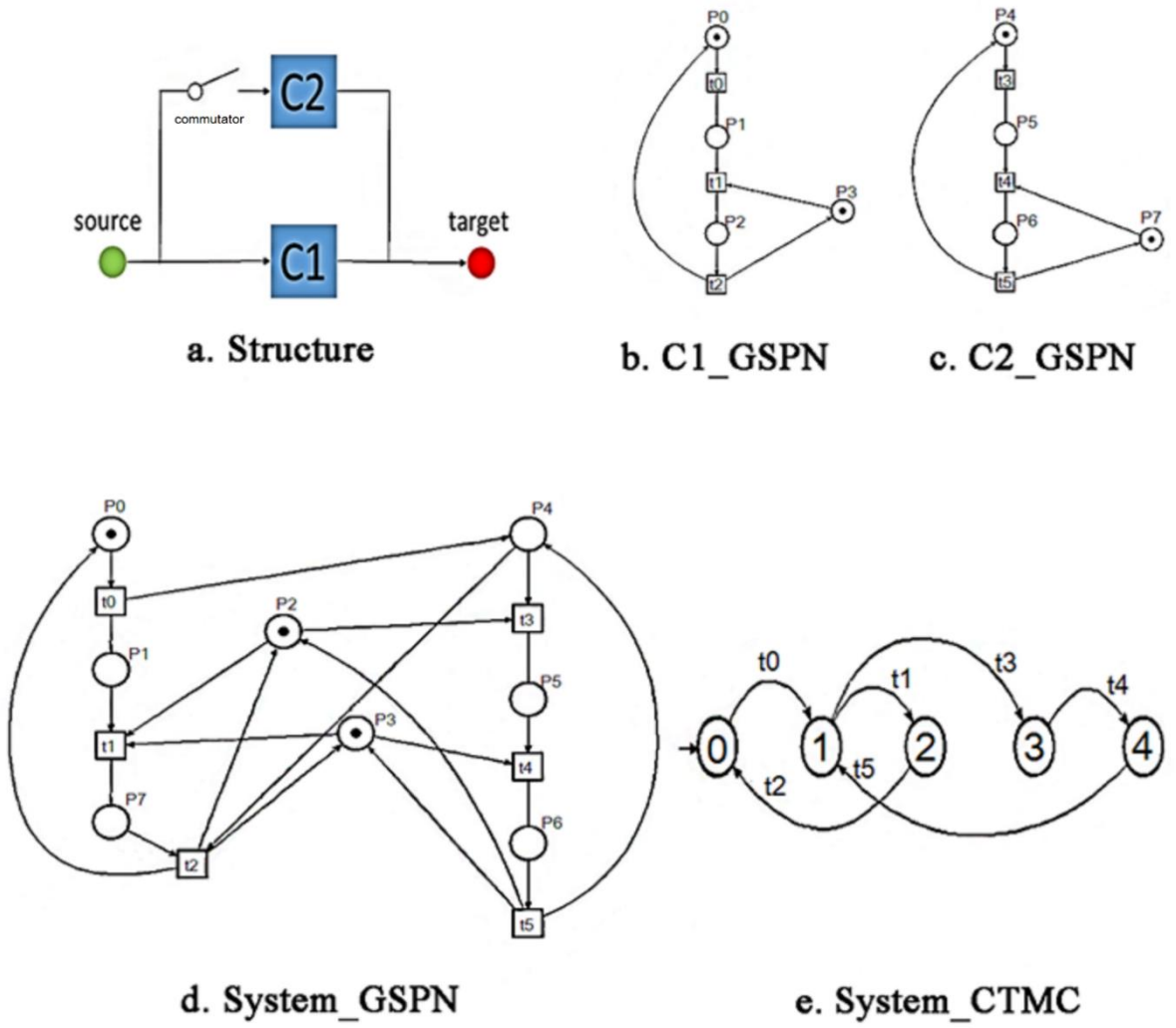


Figure 16 transformation between Stochastic Petri Net and CTMC

Petri nets offer a complete modeling framework encompassing dysfunctional aspects and, able to generate equivalent CTMC models ready to be exploited by MBSA. Yet, according to the research motivation stated in this document Petri nets conserve some drawbacks that are discussed below.

Discussion

The main advantage of Petri nets comes from their compositional ability for generating the system model, as well as their natural ability to integrate the system structure in the behavioral model. Yet the following points remain to be investigated:

- though Petri net are able to consider the modeling of failure sequences, the manual expression of the global interleaving is difficult and error-prone. The integration of specifications for desired repair sequences is an additional difficulty;

- the compositional process starting from the interconnection of individual Petri net models calls for

an important modeling overhead in order to guarantee the global behavioral coherence in presence of a connecting transition or in an 'Interaction Petri Net'. Besides, the Petri net interconnection operation has a strong impact on the model scale, which complicates the system assessment. According to the statement of the research motivation, this problem should be solved by the modular modeling methodology.

- though Petri nets can be translated into CTMCs in order to achieve MBSA, this transformation only contributes to the model generation; there is no solution for specifying the operational dependability.

1.3.2 Review of MBSA generation approaches

The MBSA generation approaches contribute both in scientific research and engineering projects. Three typical approaches are selected to be discussed in this section. BDMP is the theory extension of FtA and Markovian process, whose mechanisms (for the consideration of failure sequence) shows interesting potential for the specification of the operational dependability, by considering the failure sequence. The AltaRica framework owns a powerful modeling feature and is able to interact with other modeling approaches. In the GRIF framework, several widespread dependability analyzing methodologies are available as distinct packages.

A.Boolean Logic driven Markov Processes (BDMP)

BDMP[71] is a safety assessment model generation approach combining FtA and Markov process, which owns the function of assessment evaluation and Monte Carlo Simulation[72]. In BDMP, the dynamic systems are modeled as an easy built FtA- analogous model, with the leaves associated with the Markovian process, with the triggers handling the failure sequence of the components. The presentation of dynamic priority making BDMP much more similar as Dynamic Fault tree Analysis, the major advantage to Dynamic Fault tree Analysis is that BDMP is able to model the repairable systems with a variety of dependence[73]. Even the complex reconfiguration trajectory is also considerable by the enhanced BDMP(named Generalized Boolean logic Driven Markov Processes), in which Triggered Markov process and Switched Markov process are used[74]. The application of Boolean logic makes the BDMP more readable and enables further processing. The equivalent Markov chain of BDMP can be generated, so the assessment calculation is based, as usual, on the Markov chain. Related to BDMP, a formal supporting software tool is created using Figaro language, named KB3. KB3 aims to improve the quality, rapidity, and accessibility of dependability studies automatically and it is able to realize the functions of BDMP theory[75, 76].

In comparison, Petri nets are more flexible but BDMP provides more readability and scalability. Also, BDMP shows it ability in the modeling and characterizing safety and security interdependencies[5]. The power of BDMP is hence reflected in various applications.

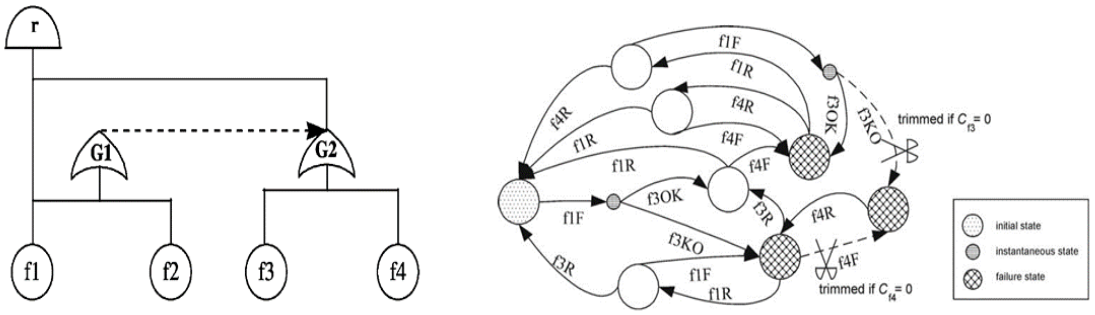


Figure 17 BDMP

Figure 17 (cited from [72]) presents an example used to explain the Markov chain model generation by BDMP. The left hand side shows the BDMP modeling and the right hand side is its Equivalent Markov chain Model. In this BDMP, the trigger is presented by a dotted arrow. 4 components labeled by f1, f2, f3 and f4 are represented in this model. The underlying logic of this BDMP model states that 'the system is broken when f1 is failed and (f3 or f4 is failed)'. Besides, whenever gate G1 asserts a failure, it instantaneously forces the same assertion on G2, normally triggered by Markov components f3 and f4. Component f2 acts only indirectly by causing a mode change for f3 and f4 when it fails. The Markov chain is presented using the following notation: xF means 'failure of x', xR means 'repair of x', xOK means 'successful startup of x', xKO means 'failure on demand of x'. It has an initial state, a failure state and a so called *instantaneous* state (the instantaneous state is corresponding with the trigger). According to this example, it can be seen that BDMP is able to consider of failure sequences, which has a potential interest for operational dependability.

BDMP is powerful, not only by easily generating the model suitable for assessment calculation and Monte Carlo simulation, but also by the ability to consider the various system situations, for example failure, failure of demand, reconfiguration and weather impacts (the weather influences the system reliability, such as bigger failure rate than the normal weather). Nevertheless, as the variety of situation consideration increases, the complexity of the modeling increases. Driven by the research motivation for the operational dependability, there exists a demand that various and complex repair-related situations of the system (for example, repair priority and system maintenance operation) should able to be easier considered in the model included in one model generation approach framework.

B.AltaRica

The AltaRica formal language is designed to describe the safety-critical systems by the form of Guarded Transition System[77], for handling with safety analyses[78] (assessment[79], simulation[80],

model checking[81] and verification[82]), with several integrated environments Cecilia OCAS (Dassault Aviation), Simfia (EADS Apsys),and Safety Designer (Dassault Systemes)[79].

Figure 18 (cited from[78]) presents the overview of AltaRica. AltaRica framework owns a powerful modeling ability. For example, AltaRica is able to model complex multi-state systems[83], multi-physical systems[84], and timed systems[85, 86]. Moreover, the association with the other modeling languages also enhances its modeling and safety analysis function domain. SysML is possible to be translated into AltaRica, which enlarges its modeling domain[87, 88]. The translation with NuSMV enables the function domain of verification and safety assessment using symbolic mothodlogy[89, 90]. The complementary use of AltaRica and Event-B enhances both the modeling and function domain, in which Event-B is responsible for system designing task and AltaRica is responsible for safety analysis task[91]. Additionally, the algorithm of mapping between AADL and AltaRica also enhance its modeling and function domain[92]. Several industrial applications cases confirm the truth that AltaRica has already reached industrial maturity. For example, the reliability assessment for the complex electrical system[93] and the complex Aircraft System[94, 95].

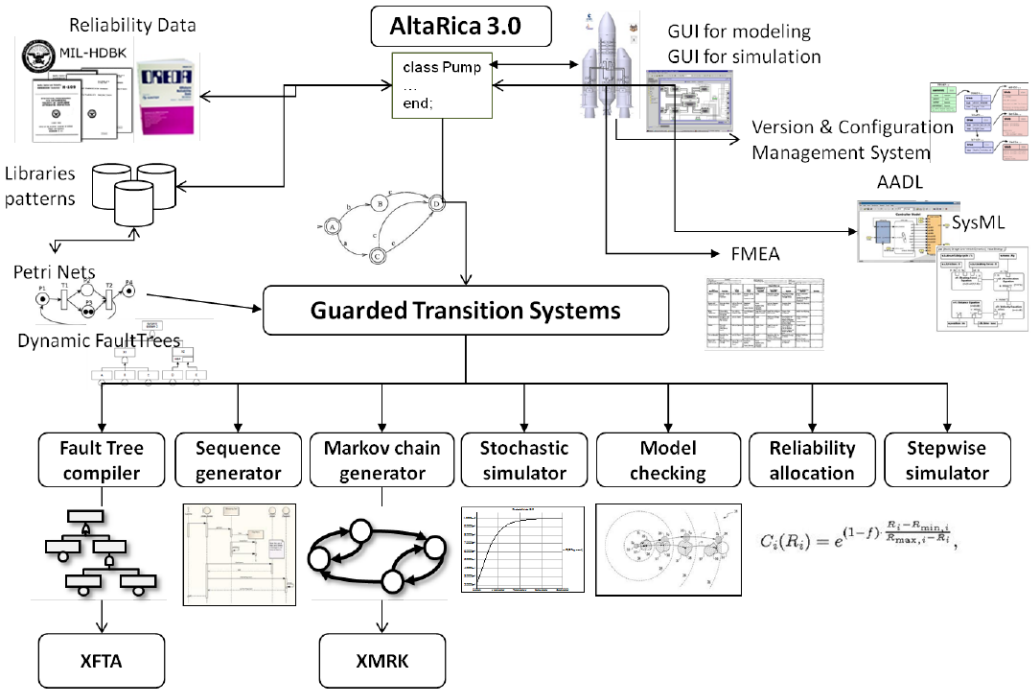


Figure 18 Overview of AltaRica 3.0 project

According to the overview of AltaRica, considering the aspect of safety analysis models generation, AltaRica owns the function of FtA compiler and Markov chain generation[78]. The algorithms of [96, 97] have demonstrated the formalism translation from Guarded Transition System to FtA, based on the theory of compiler from mode automata into Boolean equations[98]. And this proposal is improved by simplifying FtA and by automatical FtA generation solution[99]. Moreover, as SysML is able to describe the system behaviors according to the structure and requirements, by using Block Definition Diagram

to define the components, Internal Block Diagram to present the relationships of the components, State Transition Diagram to describe the states of the components, Activity Diagram to present to the transitions, etc[100]. The model transformation between from SysML and AltaRica is possible to enhance the modeling ability for the purpose of safety supporting (SysML mainly for modeling and AltaRica is mainly for handle the safety) [101], for example the aging and maintenance characteristic of the system is simulated, by using SystML to model and using AltaRica Data Flow[88]. However, the formal as well as the simplified faulty model generation approach from SysML suitable for all the systems is still not a finished work[101], which means there is no previous proposal able to present the faulty model generation and dependability assessment based on the faulty model by using SysMl transformation with AltaRica. Especially for specifying the various management requirements in the faulty model, the work will be resource-consuming and error-prone by this method. So, the major significant finding to emerge from these research works is to reveal the generation ability of AltaRica. Notwithstanding, these generated models (Markov chain and FtA) are normal, as well as they own the same drawbacks as what is stated in Petri Net, Markov Chain, BDMP or GRIF.

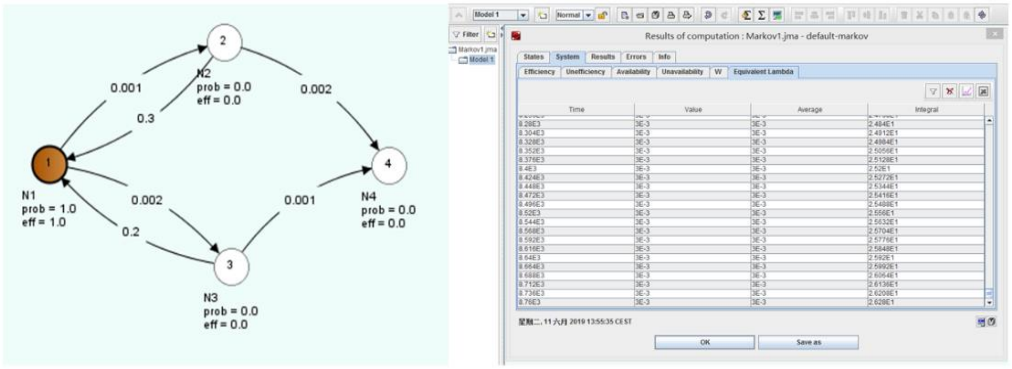
In summary, according to the research motivation, the AltaRica dependability assessment function does not consider repair trajectories in the model at hand which shows a limited support for the operational dependability aspect in its MBSA generation approach.

C. GRIF

Considering the demand for dependability analysis, the formal approach is supported by GRIF. GRIF is able to analyze the system components behaviors by failure and repair, as well as analyze the system structures. Its Markov chain package is applied for the time indicator calculation, mainly considering the failure and repair system behaviors. The Petri Net package is applied for the system behaviors describing. Fault tree Analysis package and Reliability Block Diagram package are applied for analyzing system structures.

C.1 The Markov chain package

According to the 'Reliability Markov chain modeling principle' introduced in the previous part, an example of reliability CTMC of a parallel structure (shown in Figure 11) is modeled in Figure 19, assuming: $\lambda 1=0.001$fph; $\lambda 2=0.002$ fph; $\mu 1=0.3$ rph; $\mu 2=0.2$ rph. It is able to calculate the necessary system assessment results, shown on the right side in Figure 19. For example, equivalent lambda is able to calculate the MTTF, according to equation $MTTF=1/\lambda$.



a. Series Structure CTMC

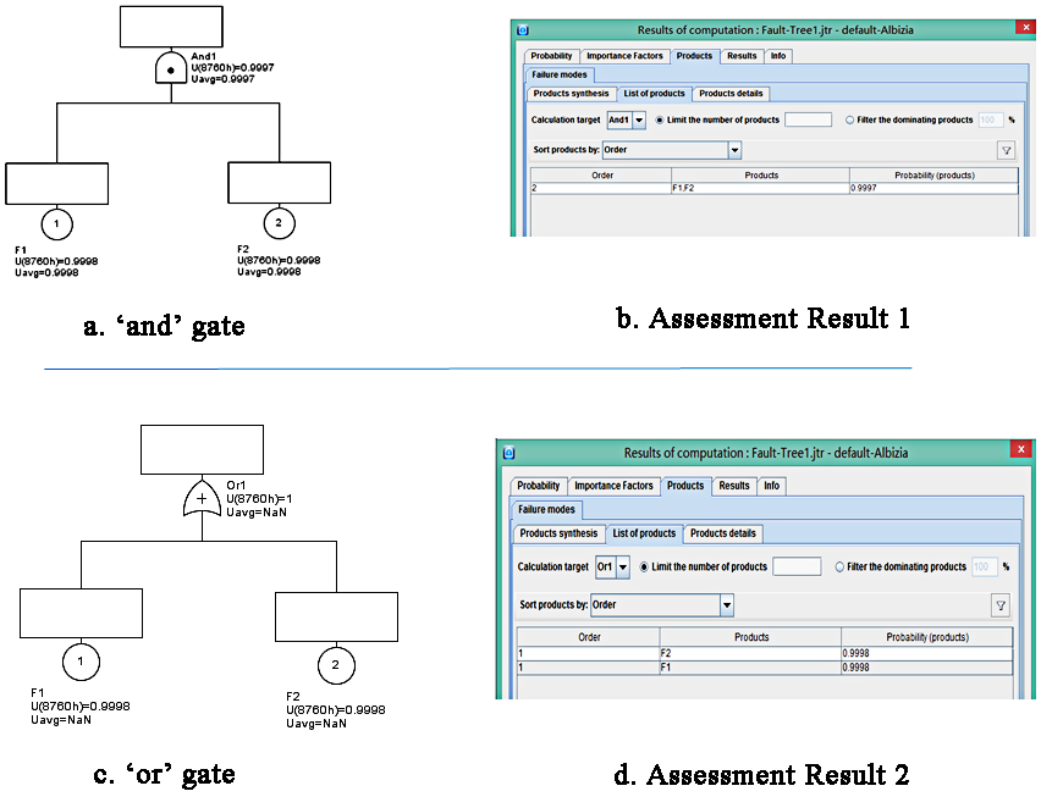
b. Reliability Assessment

Figure 19 Time indicators calculation by GRIF CTMC

The Markov chain package of GRIF is able to assess the system reliability by using CTMC model, but it is not able to handle easily the maintainability (for example, repair trajectory) during the reliability assessment.

C.2 FtA (Fault tree Analysis) package

In GRIF, the structure analyzing tool Fault tree Analysis uses conventionally ‘and’ gates for analysis of parallel structures and ‘or ’gates series structures.



a. ‘and’ gate

b. Assessment Result 1

c. ‘or’ gate

d. Assessment Result 2

Figure 20 Fault tree Analysis

Depicted in Figure 20, the upper side is: an FtA using a ‘and’ gate to analysis the parallel system structure, F1 and F2 stand for the failure of two components, the assessment result is shown on the right presenting the Lambda calculation of parallel structure, and the system failure logic expression is obtained: F1∧F2. The lower side is: a FtA using an ‘or’ gate to analyze the series system structure, the assessment result is shown on the right presenting the Lambda calculation of series structure, and the system failure logic expression is: F1∨F2. This failure logic expression offers the information explaining how the components breaking down leads the whole system breaking down, which is helpful for the ‘system total broken down state identifiable’ in reliability model generation. In GRIF, the FtA is only focused on the consideration of failure and repair rates, it is short of the consideration of repair sequences, and this shortage loses a considerable ability of operational dependability.

C.3 The RBD (Reliability Block Diagram) package

Similar to the FtA, the RBD is also applied for the structure analysis. The blocks are distributed in RBD by parallel or series configuration. These blocks represent the corresponding components of failure behaviors.[102]

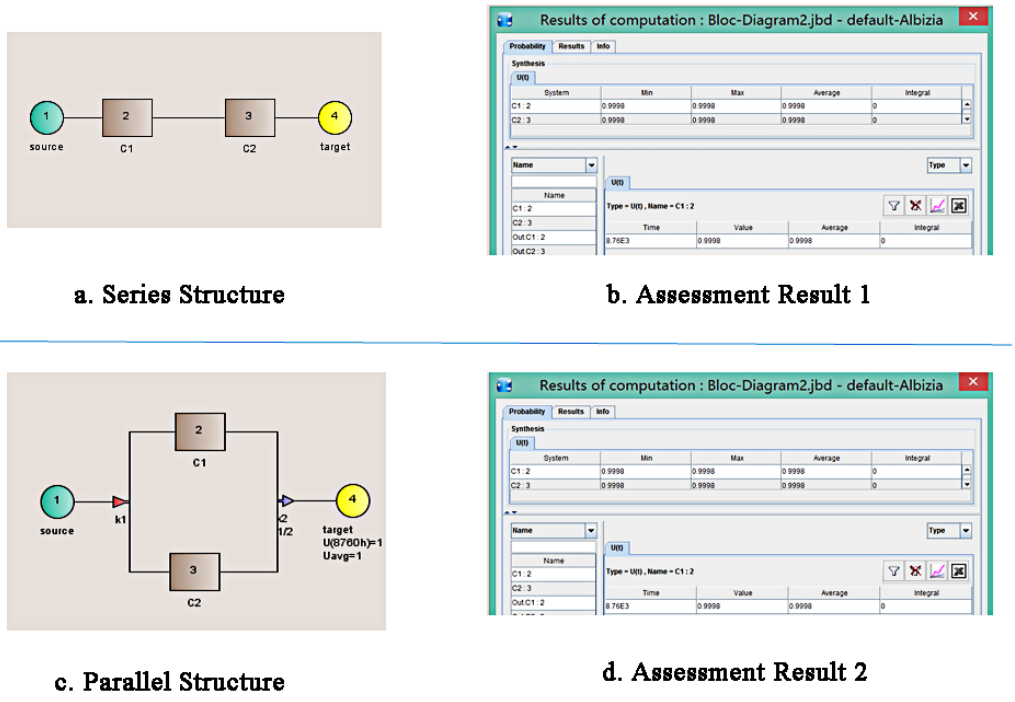


Figure 21 RBD assessment approach

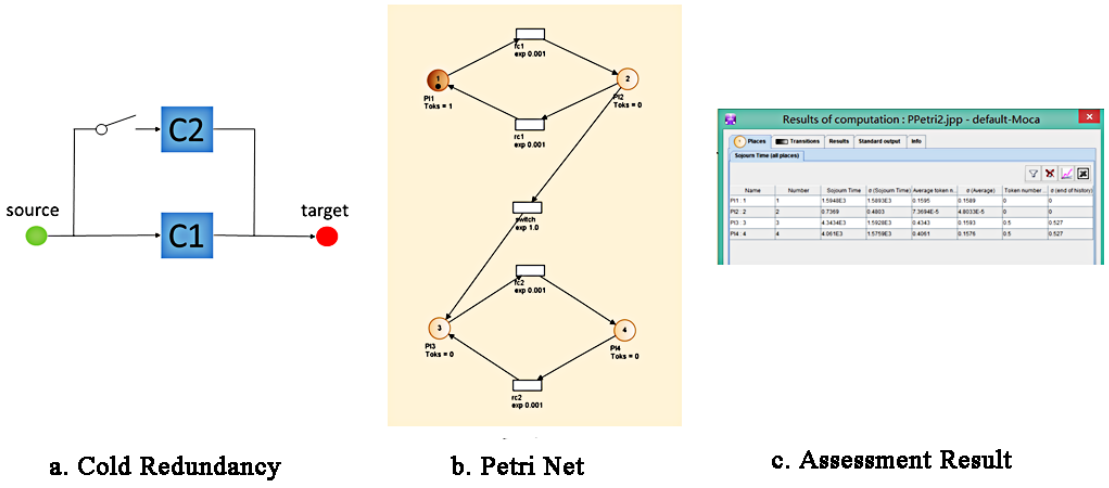
According to Figure 21, the upper side shows the RBD model presenting components C1 and C2 are working in the series collaboration, with source and target. The reliability assessment result is shown in the right side, presenting the reliability of this structure. In the lower side is the parallel system RBD model and its assessment result is shown in the right side, and the result presents reliability of the structure.

The RBD model is similar to FtA, but focuses the behaviors which successfully realizes the global service. They are also able to consider the failure and repair behaviors of the components to achieve the reliability assessment function. But the consideration of repair sequences (such as the repair priority) is unavailable.

C.4 The Stochastic Petri Net package

An example is used to explain how the Stochastic Petri net is applied for the system assessment by GRIF. Depicted in Figure 23, the system is shown in the left part, component C1 is in the main working location, when C1 breaks down the system switches to starting C2 working, while these two components both own the ability to be repaired.

In the middle part of this figure, there is the Stochastic Petri nets for component C1 and C2 respectively. PI1: Component C1 is working; PI2: Component C2 is broken down; PI3: Component C2 is working; PI4: Component C2 is broken down; fC1: failure of Component C1; r C1: repair operation of Component C1; fC2: failure of Component C2; r C2: repair operation of Component C2; switch: after component C1 is broken down the switcher turns to let component C2 starts to work. And the assessment result is shown is the right part. In this example, there is no repair of C1 after switching.



a. Cold Redundancy b. Petri Net c. Assessment Result

Figure 22 Petri Net Assessment Approach for Redundancy system

D. Discussion

BDMP, AltaRica and GRIF are selected as the typical approaches for the statement, comparison, and discussion in this section. GRIF is constituted by several independent packages and each package is applied for one traditional and useful modeling theory, for example, FtA package. BDMP is creative in the modeling generation theory by the extension of FtA and Markovian Process. Especially, by the application of trigger for component failure sequence, BDMP partly contributes to the consideration of operational dependability. AltaRica is a powerful model generation tool, not only owns the coding agility for describing the complex behaviors of the system, but also owns the generators to transform its

Guarded Transition System into other models (for example, Guarded Transition System can be transformed with SysML) to enlarge its function.

Besides GRIF, BDMP, and AltaRica, similarly there are also some other generation approaches which have already been successfully applied in the industrial projects. Some of them are : BlockSim[103] handles with DES model generation by using defined blocks and defined system layers, JMP[104] provides a statistical analyse tool which can be used for dependability analysis, Hip-HOPS[105] generates the model using hierarchy and flow diagram is used to describe the behaviors in each hierarchy, SAML[106] handles both the failure mechanism by qualitative and safety assessment by quantitative, PRISM for model checking[107], etc. However, these approaches are also not able to handle the operational dependability, for the similar reason. GRIF packages handle conventional model generation approaches. They could integrate a specification of the operational dependability inside the base models, but are unable to generate the global model for further assessment (the reason is stated in the discussion part of FtA, RBD, CTMC and Petri Net). For example, if the components are set to operate as 'first failed first to be repaired', the GRIF packages have no solution to treat this situation.

Table 1 synthesize the model generation approaches in the MBSA context, featuring contributions for operational dependability. For the first four items (CTMC, FtA, RBD and PetriNet) are the very usual applied model for the system failure scenario analysis, which have already been stated in the introduction of each approach. AltaRica is able to build these models by its powerful coding functions, but BDMP can generate the CTMC by its modeling features.

The last five items are related to the necessary elements for our research motivation (the consideration of operational dependability). The structure should be formally analyzed in order to generate the formal dependability model, thus structure analysis is necessary. Unique transition rates and N-state behavioral models should be considered, for the reason that the full faulty behaviors of the system should be considered by model. Repair trajectory specified by the operational requirements should be expressed in the model to handle with the operational dependability by qualitative. The effectiveness should be quantitatively presented in order to judge the operation requirements set onto the system is suitable or not, which is a necessary aspect for the operational dependability by quantitative. By these items, the model is able to handle the operational dependability, by the calculation of the time indicators, to resubmit the system with a qualitative improvement of the dependability regarding the operations. These five items will be discussed as follows.

The 'unique transitions', 'N-states' and 'structure analysis' are the necessary items for the proposed operational MBSA model generation. The 'unique transitions' feature means the unique rates of all the behaviors should be considered in the model, which is a prerequisite for the operational dependability, for the reason that the event trajectory of requirements can be specified only in case that each unique transition is considered in the model. BDMP and AltaRica satisfy this item, by offering a detailed modeling ability for every unique rate in the system describing the model. However, if the system is constructed by a large number of components with a complex structure, and if the component behaviors

have unique rates (as it has already been discussed in the previous part, it is meaningless to simplify the model by state combination method), the modeling task will be very difficult for the system analysts. The ‘N-states’ feature means that not only behaviors like failure and repair should be expressed, but also necessary reliability-related behaviors, such as downtime maintenance, failure of the component turning on or component failure sequence, should be considered in the model. This is another prerequisite for the operational dependability, for the reason that several operation related states should be established in order to present the operational dependability. BDMP satisfies this item by modeling in detail the behaviors of the components and using triggers to present failure sequences of the components, but if there is a demand for the consideration of the downtime maintenance behavior, its choice must be reconsidered. AltaRica is able to describe the system behaviors by language coding, thus its agility leads this item to be achievable. Nevertheless, it is also very difficult for the engineers to manually code model manual, as the component number and structure complexity increase. The ‘structure analysis’ amounts to using the correct and formal methodology to analyse the system structure in order to illustrate the system failure scenario. This is necessary for the Reliability, Availability and Maintainability model generation, without which the operational dependability model could not have the function of dependability assessment. FtA is one of the formal structure analysis tools. This item is easy to be achievable, BDMP is based on the modeling methodology of FtA, and AltaRica is able to transform the model into FtA.

The ‘repair trajectory’ and ‘efficiency evaluation’ are the main items for the consideration of operational dependability in the dysfunctional model. The ‘repair trajectory’ means the repairing sequence of the components, which should be specified in the model in order to contribute to the operational dependability by qualitative. The ‘efficiency evaluation’ means evaluating the system efficiency when partially broken down or partially repaired. This item is able to identify which ‘repair trajectory’ is better and how healthy the system recovers by this ‘repair trajectory’, which contributes to the operational dependability by quantitative. As it is already stated in each model generation approaches, there is no current solution for these two items in one unique framework. BDMP is based on the methodology of FtA (FtA is good at considering how the system does fail). Though it is able to consider the failure sequence by using the trigger, the methodology of FtA has the limitation that the ‘repair trajectory planning’ is short of consideration. Besides, efficiency evaluation in BDMP needs to be handled separately. As for AltaRica, it is of the same shortage, and besides, the specification integration is not directly integrated. However, though these approaches are not able to fully satisfy the research motivation by one framework, the association of models by AltaRica and the generation evaluation tool by BDMP inspires the solution proposed: associate the adequate functions from different approaches in order to realize the operational dependability.

Table 1 Review of the model generation approaches

Name / Function		BDMP	AltaRica
Models	**CTMC**	**√**	**√**
	FtA	**-**	**√**
	RBD	**-**	**√**
	Petri Net	**-**	**√**
structure	**Structure analysis**	**√**	**√**
Faulty behaviors	**Unique transitions**	**√**	**√**
	N-state behaviors	**-**	**√**
Operational Requirements	**Repair trajectory (by qualitative)**	**-**	**-**
	efficiency evaluation （by quantitative）	**-**	**-**

Overall, according to the previous statement, there is no existing model generation approach able to fully achieve the consideration of operational dependability included in one framework. Especially, the management of repair operations and the evaluation of repair efficiency is unavailable. Thus, the contribution of this research will fill these gaps.

Chapter 2
Operational Reliability Model Generation

This chapter mainly contains three parts:
First, the proposal for the reliability model generation;
Second, the proposal for the consideration of operational dependability in quality;Third, the consideration of the operational dependability in quantity, which is realized by the Capacity Calculation Fault Tree

Contents

2.1 Framework Introduction

This chapter introduces the proposal of the MBSA model generation approach by the consideration of **operational reliability**. Conventionally, the dependability assessment merely relies on the CTMC model representing the faulty behaviour of the component at hand. The processing of this model provides all the conventional indicators that can be expected.

In contrast, operational dependability is able to provide a model enriched with behavioral information, mostly related to *how* the reactions to failures can be conducted.

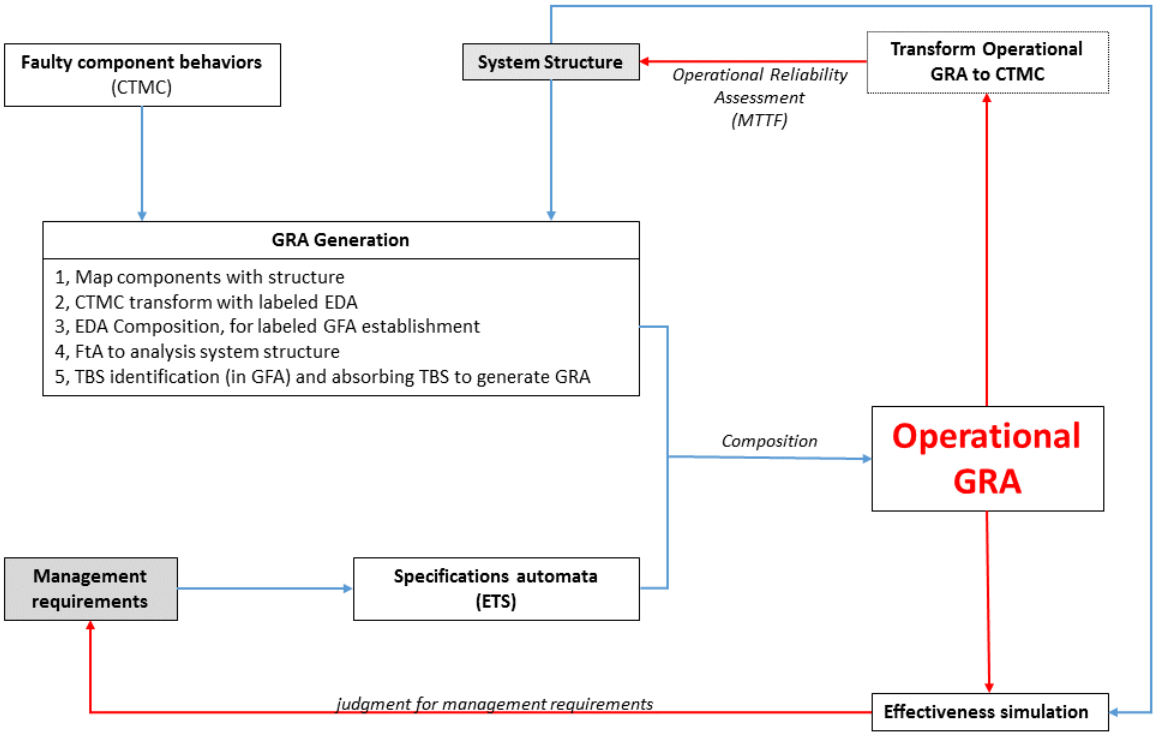


Figure 23 framework of proposal

Figure 23 depicts the modular modeling framework of this proposal. According to the research motivation, a MBSA generation approach with the consideration of operational dependability is needed. By the assessing function of the target model, an improvement of the dependability for a system is able to be resubmitted. By specifying the management requirements in the target model and calculating the effectiveness caused by the management requirements, the operational dependability is able to be considered. In this proposal, the target model is termed 'Operational GRA' in the figure. There are three inputs of this proposal:

- the faulty components behaviors: the CTMC models of the components;

- the system expected structure for delivering the service;
- the management requirements, such as the requirements for planning the repair sequence of the components.

Based on the system design and faulty components behaviors, a Global Reliability Automaton (GRA) is generated. This GRA model (no the operational one) generation is issued by the following steps:

1. map the component CTMC with the structure, in order to fully consider the system;
2. transform the CTMC into a labeled Event Driven Automaton (EDA). Labels represent Boolean assertions used to describe the sets of characteristics of each states;
3. compose local EDA into a global model, the GFA (Global Faulty Automaton);
4. express the failure logic expression from the FtA and the system structure;
5. identify the Total Broken Down State (TBS) according to the failure logic expression and make the TBS an absorbing (sink) state in the GRA.

Based on management requirements, specification automata are designed. The composition of GRA and specification automata establishes the Operational GRA. By the simulation of the event scenario in Operational GRA and by a proposed calculation methodology for the system's effectiveness, the dynamic effectiveness of the system caused by the management requirements is able to be simulated. Simulation will be needed when the resulted operational GRA is too big. The resulted value of the effectiveness simulation handles the judgment that which management requirements are suitable for the system, thus the optimal requirements are able to be set onto this system. Thanks to the genericity of the GRA, it can be fed to existing dependability assessment tools. By the assessment result (time indicators are calculated), the dependability level of the system design is evaluated, which is helpful for the engineers to resubmit an improvement of the dependability of the system.

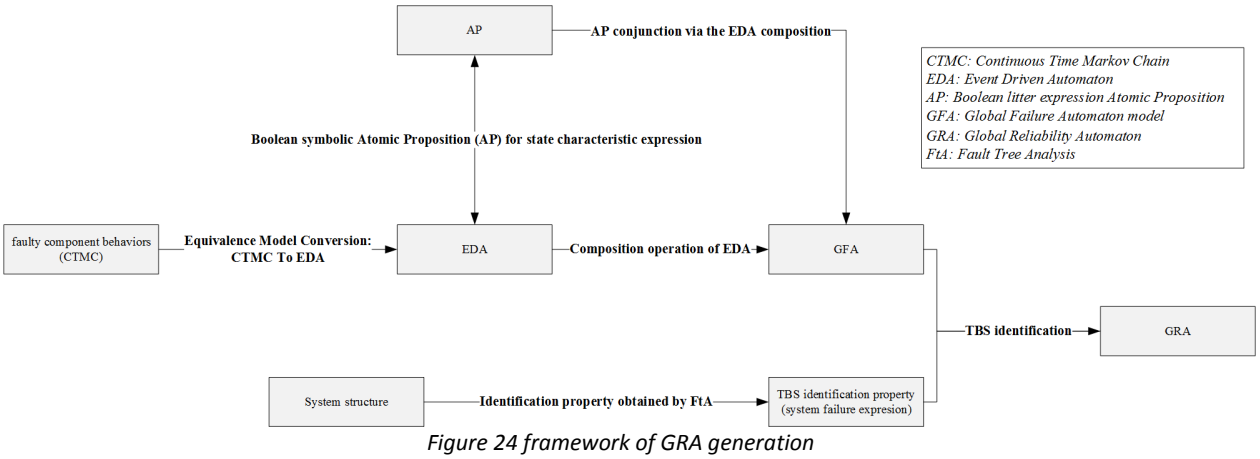


Figure 24 framework of GRA generation

According to Figure 24, the GRA model generation approach is proposed in Section 2.2, by five

steps. Continuous Time Markov Chain (CTMC) model is used as the safety assessment model. The occurrence rates of the transitions in CTMC are mapped to the transition events in the Event Driven Automaton (EDA)[108, 109]. This step relies on previously published results [110]. Their application is presented in 2.2.1.1. The consideration of N-states behaviors and unique events are stated, and this is issued by the composition of component-models. Thus, GFA (Global Faulty Automaton) is established. This composition operation is presented in 2.2.1.2. According to its definition, the reliability model starts with the initial state and stops at the system total broken down state (TBS). The TBS states in GFA should be identified and absorbing in order to generate the GRA. However, when there is a need of state identification, state names are not explanatory enough, and there is no existing formal approach for the state identification only by using the state names. So, AP (Boolean Symbolic Atomic Proposition) is applied as a further expression of the state characteristic in the component-model, which is stated in 2.2.1.3. As an improvement of the composition operation, the component- AP conjunction formula via the model composition is studied in 2.2.1.4. Thus, in GFA, the causality resulted from the service delivering can be expressed by the states in the form of AP. In 2.2.1.5, the formal FtA (Fault tree Analysis) is used for analyzing the system structure, which provides the 'failure logic expression' of the system. As each state is associated with AP, the 'failure logic expression' acts as the reference of judging which state stands for the system totally break down. Therefore, the TBS (Totally Breakdown State) can be identified. According to the definition of Reliability Model, the Reliability Model begins from system initially working state and stops at the system totally breaks down state (TBS), thus the TBS in GFA should be operated to be absorbing to generate GRA, which is proposed in 2.2.1.6. Moreover, this model generation approach is fully illustrated by an example in 2.2.2.

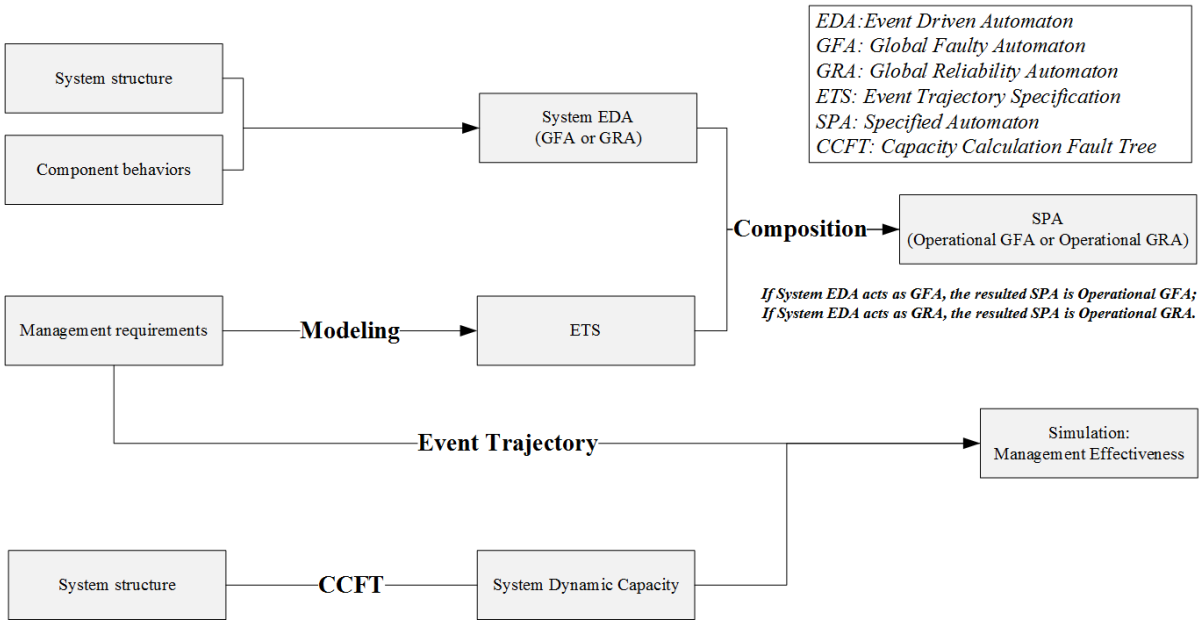


Figure 25 framework of operational dependability

As illustrated by Figure 25, Section 2.3 presents the proposal for the operational dependability. For specifying the event trajectory caused by management requirements, the implementation method for the

consideration of operational dependability in the MBSA model is presented. Management requirements are set according to the various situations of the system, which forms the repair trajectory in the model. Thanks to the specifying solution in the Discrete Event System Theory[111, 112], ETS (Event Trajectory Specification Automaton) are able to be identified. By the composition of System EDA (resulted from the system structure and component behaviors) and ETS, the SPA(Specified Automaton) is established. This specifying method is suitable for both GFA and GRA. So, if the system EDA acts as GFA, the resulted SPA is the Operational GFA, and if the system EDA acts as GRA, the resulted SPA is the operational GRA. This specifying methodology is presented in 2.3.1. Besides, several conventional applied management requirements are selected to be specified in 2.3.2. As for the redundancy systems, components are possible to be breakdown as well as to be repaired, the production capacity of the system changes. A first quantitative analysis methodology (named CCFT: Capacity Calculation Fault Tree) of the system production capacity is studied in 2.3.3. By the application of CCFT, according to the event scenarios executed by the management requirements in the system model, a simulation method for the effectiveness of the management requirement is studied in 2.3.4. In this simulation method, the 'economic cost', 'time cost' and 'system capacity' are analyzed to judge 'which management requirement is suitable for the various situations of the system'. Additionnally, a second quantitative issue is open thanks to the conventional assessment tools that can be applied.

2.2 Global Reliability Automaton Generation

2.2.1 Generation approach

The following paragraphs present the reliability model generation approach, able to handle the specific features previously emphasized: unique transitions (unique rates associated to each expected event), N-state behaviors (ability to model functional states other than failed and repaired), and formal structure analysis in order to establish the model.

2.2.1.1 Behavioral Equivalence Model Conversion: CTMC VS EDA

This section introduces the transformation between CTMC models and EDA. Recall that this transformation is perfomed on order to reach the ability of composing behaviors: dysfunctional behaviors and operational behaviors. A global reliability automaton (GRA) is obtained, modeling both dysfunctional and operational behaviours. The final assessment requires a back transformation, from GRA back to a global CTMC in order to recover the ability of computing time indicators.

It must be emphasized that CTMC models are usually **quantitative**. Transitions hold rates which are numbers, and it is up to the human designer to keep track of the **signification** of each rate: the actual event whose occurrence is expressed. So in the sequel, in order to obtain a behavioral model (the EDA),

it is important to make the event-rate association explicit. This is why, an enriched CTMC model definition is indroduced, as illustrated in Figure 1(b).

- **Enriched CTMC definition**

An enriched CTMC is a tuple:

$$CTMC=(Q_{mc}; E_{mc}; \delta_{mc}; q0_{mc})$$

where Q_{mc} is the state set, $E_{mc} = E \times \mathbb{R}^+$ is the event set, associating each event $e \in E$ and a non negative rate $r \in \mathbb{R}^+$. The 'unique transition' feature considered in this work, constraints this set:

$$\forall (e_1, r_1) \in E_{mc}\ \forall (e_2, r_2) \in E_{mc} : r_1 \neq r_2 \rightarrow e_1 \neq e_2$$

which amounts to say that each event is always associated with the same rate. For this reason, the occurrence rate of any event can be modeled by a function, denoted R_{xy}. The transition function δ_{mc} is defined as:

$$\delta_{mc} : Q_{mc} \times E_{mc} \rightarrow Q_{mc}$$

The state $q0_{mc} \in Q_{mc}$ is the initial state of the CTMC.

- **Event-Driven Automaton (EDA) definition**

An Event-Driven Automaton (EDA) is a tuple:

$$EDA= (Q_{EDA};E_{EDA};\delta_{EDA};q0_{EDA};qm_{EDA})$$

Where Q_{EDA} is the state set, E_{EDA} is a set of events, δ_{EDA} is the transition function, $q0_{EDA}$ is the initial state and qm_{EDA} is a marked state.

- **Enriched CTMC to EDA translation**

An EDA is systematically obtained from an enriched CTMC by applying the following rules[110]:

- $Q_{EDA} = Q_{mc}$: the sets of EDA states and CTMC states are the same;
- $E_{EDA} = \{e | \exists (e, r) \in E_{mc}\}$: the events of the target EDA are exactly those appearing in the enriched CTMC;
- $\delta_{EDA}(q, e) = q'\ iff\ \delta_{mc}\left(q, \left(e, R_{xy}(e)\right)\right) = q'$: the transitions of the EDA correspond to the transitions of the enriched CTMC;
- $q0_{EDA} = q0_{MC}$ the two models have the same initial state;
- qm_{EDA} is a marked state, further designated by the designer, as shown in the sequel.

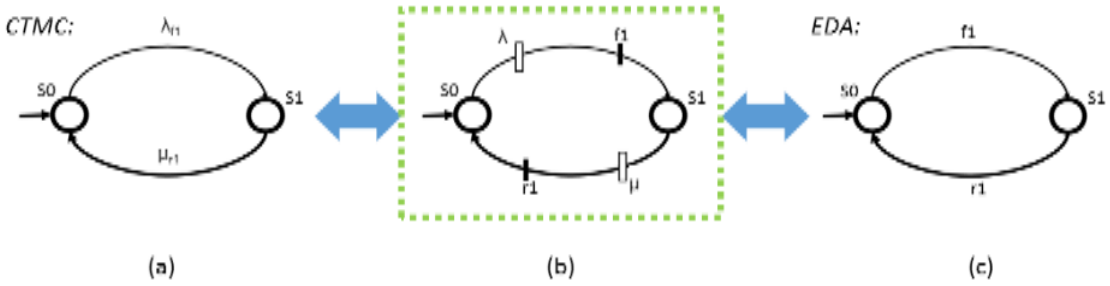


Figure 16 Model translation between CTMC and EDA

Figure 16, the models (a), (b), and (c) illustrate the chain of possible translations between the quantitative CTMC, the enriched CTMC and the EDA. It represents the same failure-repairable system, with an initial working state S0 and a system break down state S1, the failure event (denoted by f1) occurs with rate λ, and the system repair operation (denoted by r1) occurs with the rate μ. Concerning this example, the following statements hold.

The states sets are identical:

$$Q_{mc} = \{S0, S1\} = Q_{EDA}$$

The transitions in the enriched CTMC are

$$\text{In CTMC (b)} : \begin{cases} \delta_{mc}\big(S0, (f_1, R_{xy}(f_1)\big) = S1 \\ \delta_{mc}\big(S1, (r_1, R_{xy}(r_1)\big) = S0 \end{cases}$$

The transition expression becomes δ_{EDA} according to the above transformation rules, as illustrated in Figure 26 (c): the transitions are driven by events, and each event happens by its rate.

$$\text{In EDA (c)} : \begin{cases} \delta_{EDA}(S0, f1) = S1, by\ the\ occurrence\ rate\ R_{xy}(f_1) \\ \delta_{EDA}(S1, r1) = S0, by\ the\ occurrence\ rate\ R_{xy}(r_1) \end{cases}$$

2.2.1.2 Composition operation of EDA

For the reason that CTMC is able to be transformed into EDA, the composition of faulty component behaviors which is described by CTMC can be issued by its equivalent EDA composition operation. This part studies the formal method of the automata composition operation.

As the definition of EDA is already given: $EDA = (Q_{EDA}; E_{EDA}; \delta_{EDA}; q0_{EDA}; qm_{EDA})$. The composition of EDA is based on the automata composition operation in the discrete event system theory[111]. Assume two automata EDA1=(Q1; E1;δ1;q01;qm1) and EDA2=(Q2; E2;δ2;q02; qm2), the composition operation (denoted by '//') is defined :

$$EDA1//EDA2= (Q1\times Q2;\ E1 \cup E2;\ \delta;\ q01\times q02;\ qm1\times qm2;) \quad (2\text{-}3)$$

$$\delta((x.y),t) = \begin{cases} (x',y) & \text{if } t \in E1\backslash(E1 \cap E2) \text{ and } \delta1(x,t) = x' \\ (x,y') & \text{if } t \in E2\backslash(E1 \cap E2) \text{ and } \delta2(y,t) = y' \\ \phi & \text{if other } eslse \end{cases}$$

Assume one system is constructed by two components. If a very serious accident happens to this system, this accident causes the failure of both these two components, which is denoted by 'commonbreak' in the model, while each component has its own unique repair and failure performs. As illustrated in Figure , the two components model are depicted as C1 and C2, and the composition result is C1//C2. Here C1//C2 is able to present the behaviors of the whole system, thus C1//C2 is the GFA model of this system.

For component C1， $C1=(Q_{C1}; E_{C1};\delta_{C1};q0_{C1};qm_{C1})$

where, $Q_{C1}=\{W_{C1},B_{C1}\}$

$E_{C1}=\{repair1; failure1\}$

$\delta_{C1}(W_{C1}, failure1) = B_{C1}$; $\delta C1(B_{C1}, repair1) = W_{C1}$

$x0_{C1}= W_{C1}$

$qm_{C1}=B_{C1}$

For component C2, $C2= (Q_{C2}; E_{C2};\delta_{C2};q0_{C2};qm_{C2})$

where, $Q_{C2}=\{W_{C2},B_{C2}\}$

$E_{C2}=\{repair2; failure2\}$

$\delta_{C2}(W_{C2}, failure2) = B_{C2}$; $\delta_{C2}(B_{C2}, repair2) = W_{C2}$

$x0_{C2}= W_{C2}$

$qm_{C2}=B_{C2}$

Based on the composition equation, $C1//C2= AC(Q_{C1}\times Q_{C2};\ E_{C1} \cup E_{C2};\delta_{C1//C2};\ q0_{C1}\times q0_{C2};\ qm_{C1}\times qm_{C2})$

$Q_{C1}\times Q_{C2}=\{(W_{C1}.W_{C2}); (W_{C1}.B_{C2}); (B_{C1}.W_{C2}); (B_{C1}.B_{C2})\}$

$E_{C1} \cup E_{C2}=\{repair1 ;repair2;failure1;failure2\}$

$x0_{C1}\times x0_{C2}=(W_{C1}.W_{C2})$

$qm_{C1}\times qm_{C2}=(B_{C1}.B_{C2})$

The transitions are: $E_{C1}\backslash(E_{C1}\cap E_{C2})$={repair1;failure1} and $E_{C2}\backslash(E_{C1}\cap E_{C2})$={repair2;failure2}. Transition function of the composition result is shown in Table 2.

Table 2. Transition function of the composition result

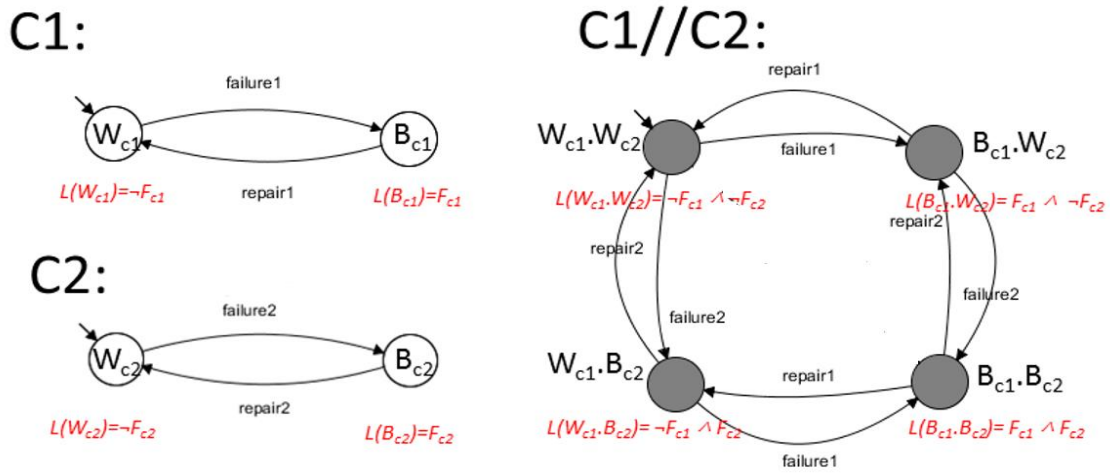


Figure 27 EDA composition and AP conjunction

In C1//C2	Locally to C1 or C2
$\delta_{C1//C2}((B_{C1}.B_{C2}), repair1)=(W_{C1}.B_{C2})$	$repair1 \in E_{C1}\backslash E_{C1}\cap E_{C2}$
$\delta_{C1//C2}((B_{C1}.W_{C2}), repair1)=(W_{C1}.W_{C2})$	$\delta_{C1}(B_{C1}, repair1)=W_{C1}$
$\delta_{C1//C2}((B_{C1}.B_{C2}), repair2)=(B_{C1}.W_{C2})$	$repair2 \in E_{C2}\backslash E_{C1}\cap E_{C2}$
$\delta_{C1//C2}((W_{C1}.B_{C2}), repair2)=(W_{C1}.W_{C2})$	$\delta_{C2}(B_{C2}, repair2)=W_{C2}$
$\delta_{C1//C2}((W_{C1}.W_{C2}), failure1)=(B_{C1}.W_{C2})$	$failure1 \in E_{C1}\backslash E_{C1}\cap E_{C2}$
$\delta_{C1//C2}((W_{C1}.B_{C2}), failure1)=(B_{C1}.B_{C2})$	$\delta_{C1}(W_{C1}, failure1)=B_{C1}$
$\delta_{C1//C2}((W_{C1}.W_{C2}), failure2)=(W_{C1}.B_{C2})$	$failure2 \in E_{C2}\backslash E_{C1}\cap E_{C2}$
$\delta_{C1//C2}((B_{C1}.W_{C2}), failure2)=(B_{C1}.B_{C2})$	$\delta_{C2}(W_{C2}, failure2)=B_{C2}$

2.2.1.3 Boolean symbolic Atomic Proposition (AP) for state characteristic expression

In either a CTMC or a corresponding EDA, the state name can be meaningless because a lack of tracking. The state meaning can become even harder to interprete after a EDA composition operation. State names turn out to be not explanatory enough. It is important to conserve the ability to target sets of states according to their significicance: failure, repairing, starting, etc. This task becomes tedious as soon as the models at hand are issued by a composition operation.

In this section, a formal state identification method is proposed, by using the Boolean symbolics to associate with state and using the failure logic expression as the identification property.

The Boolean symbolic Atomic Proposition (AP) is applied in order to associate simple assertions to the states and to compose such assertions. These assertions allow the expression and the association of specific properties for each state, as Boolean propositions. Furthermore, an association function is proposed to present the relationship between the state and its corresponding AP. To be uniformed with the previous EDA definition, the Boolean EDA with the addition of AP is defined:

$$\text{Boolean EDA} = (Q; E; \delta; q0; qm; AP; L)$$

Q, E, δ, q0, qm,δ remind to be the same as defined before.

AP is a set of atomic propositions which is defined as the form of Boolean letters, for example, a Boolean symbol ***F*** (means 'failed') stands for the 'system is broken down', and the negation of it ¬ ***F*** stands for its opposite meaning that this system is working healthily. Moreover, in some cases, the positive or the negative meaning are denoted by the value ('1' or '0').

$L:Q \rightarrow 2^{AP}$ is the association function corresponding with state and AP, here 2^{AP} is the power set of AP. Because AP is Boolean, the conjunction of the elements in 2^{AP} should be connected by logic 'and' (denoted by '∧').

Assume the AP set is :

$$AP = \{ap_0, ap_1, ap_2 \ldots\ldots ap_n\}$$

Then the power set of AP will be: 2^{AP}={∅, ap0， ap1， ap2， ap0∧ap1, ap0∧ap2, ap1∧ap2, ap0∧ap1∧ap 2…… }

It is to be noticed that to different states can share the same AP, because they feature the same property. If one state has more than one property, a combined proposition, (for example, ap0∧ap1 in 2^{AP}) is associated with this state. Besides, each state name is unique, but it is normally possible that two or more states share the same AP, because they share the same characteristic possibly. For example, if a model has three states: state 1 stands for system is idle (waiting for work), state 2 stands for system is healthily working and state 3 stands for system is broken down. State 3 will be associated with '***F***' (means system is failed), but state1 and state 2 share the same AP '¬ ***F***' (means system is not failed).

According to Figure 27, the AP applied component models are defined:

$C1=(Q_{C1}; E_{C1}; \delta_{C1}; q0_{C1}; qm_{C1}; AP_{C1}; L_{C1})$

and

$C2=(Q_{C2}; E_{C2}; \delta_{C2}; q0_{C2}; q0_{C2}; AP_{C2}; L_{C2})$.

Each component has two possible properties which can be mapped to its states: “HealthilyWorking” and “Broken down”. To establish the AP set, $\boldsymbol{F_{C1}}$ and $\boldsymbol{F_{C2}}$ are set to expressed respectively the components “broken down” property. Their negation $\neg\boldsymbol{F_{C1}}$ and$\neg\ \boldsymbol{F_{C2}}$ stands for the components healthily working.

$$AP_{C1}= \{\boldsymbol{F_{C1}}, \neg\boldsymbol{F_{C1}}\}$$

$$AP_{C2}= \{\boldsymbol{F_{C2}}, \neg\ \boldsymbol{F_{C2}}\}$$

With the help of association function L_{C1} and L_{C2} the state can be associated with AP elements according to each state meaning, depicted in Table 3.

Table 3. AP association function

Local Component:C1	
State Name	label
W_{C1}	$\boldsymbol{L_{C1}(W_{C1})=\neg F_{C1}}$
B_{C1}	$\boldsymbol{L_{C1}(B_{C1})=F_{C1}}$

Local Component:C2	
State Name	label
W_{C2}	$\boldsymbol{L_{C2}(W_{C2})=\neg F_{C2}}$
B_{C2}	$\boldsymbol{L_{C2}(B_{C2})=F_{C2}}$

2.2.1.4 AP conjunction via the EDA composition

This part introduces the principle of AP conjunction based on EDA composition. Because the system-states is the Cartesian Product result of the component states, these resulted system states own all the characteristics of their factoring component states. The AP of the component states are connected by a ‘and’ logic to produce the AP of system-state.

The composition operation of the AP applied EDA is defined:

$EDA1//EDA2=AC\ (Q1\times Q2;\ E1\cup E2;\ \delta;\ q01\times q02;\ qm1\times qm2;\ AP1\cup AP2,;\ L1\wedge L2)$

$AP1\cup AP2$ means the union of the Atomic Proposition set. $L1\wedge L2$ is the AP association function for EDA1//EDA2, which means it is the ‘and’ logic result of the components associate function. As it is known: $L1:Q1\rightarrow 2^{AP1}$ and $L2:Q2\rightarrow 2^{AP2}$, the global mapping function L is defined as:

$$L:\ Q1\times Q2\rightarrow 2^{AP1\cup AP2} \qquad (2\text{-}4)$$

Assume that two component states x and y, which $x \in Q1$, $y \in Q2$. And $(x.y) \in Q1 \times Q2$ is to denote the system-state composed from x and y. L(x), L(y),L(x.y) are to denote state AP from EDA1, EDA2, and EDA1//EDA2. The further explanation of equation (2-4) is:

$$L(x.y)=L(x) \wedge L(y)$$

The composition operation of components C1 and C2 is illustrated in Figure .

The AP union is: $AP_{C1} \cup AP_{C2}=\{ \boldsymbol{F_{C1}}, \neg \boldsymbol{F_{C1}}, \boldsymbol{F_{C2}}, \neg \boldsymbol{F_{C2}} \}$. The AP conjunction result in C1//C2 is shown in Table 4.

Table 4. AP conjunction by the CTMC composition operation

C1//C2	
State Name	AP Association Function Result
$W_{C1}.W_{C2}$	$\neg \boldsymbol{F_{C1}} \wedge \neg \boldsymbol{F_{C2}}$
$B_{C1}.W_{C2}$	$\boldsymbol{F_{C1}} \wedge \neg \boldsymbol{F_{C2}}$
$W_{C1}.B_{C2}$	$\neg \boldsymbol{F_{C1}} \wedge \boldsymbol{F_{C2}}$
$B_{C1}.B_{C2}$	$\boldsymbol{F_{C1}} \wedge \boldsymbol{F_{C2}}$

2.2.1.5 State Identification property obtained by FtA

The state identification is the act of designating the subset of states of a Boolean EDA matching a desired property, expressed as a Boolean predicate. Let P be a desired Boolean property expressed over the global AP alphabet. For a given EDA, the state identification with respect to P is the set of states Q_P defined as:

$$Q_P = \{q \in Q \; s.t. L(q) \Rightarrow P\}$$

As AP is applied for describing the state characteristic, to comprehensively establish the identification property, AP elements are used. In the identification property, AP are connected with the conjunction ('and' logic, symbol '$\wedge$') and disjunction ('or' logic, symbol '$\vee$'), and also labeled with the negation ('$\neg$'). The logic comparison of the state AP and the state identification property is able to point out which states are identified: if the AP of one state is a sufficient condition of the property, it means this state satisfies the desired condition P, and hence this state is identified by this property. The various logic combination of AP is able to establish all the complex Boolean algebra to be the state identification property, included in which TBS identifying property is.

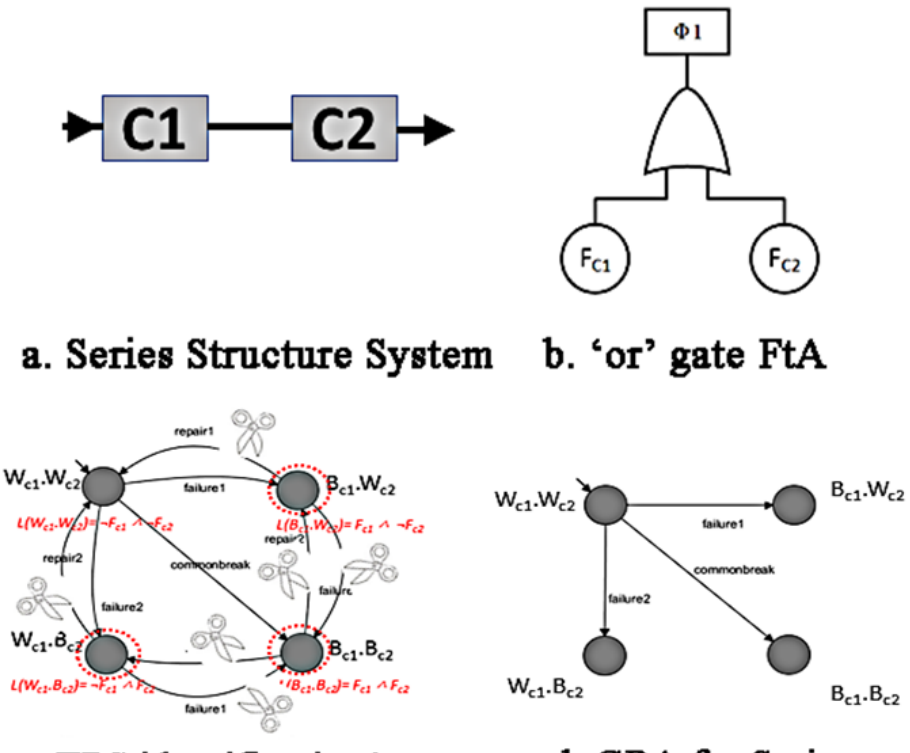


Figure 2 TBS identification for Reliability CTMC establishment

As pointed out in Figure 8, FtA is used in the proposed framework in order to map a failure logic expression to any component-based architecture. The aim here is to go beyond structure towards behavior, and to establish a mapping from the failure logic expression, corresponding to a fault tree, inside the global EDA's behaviour. This is done by identifying formally the internal EDA configuration corresponding to the failure expression. This configuration is actually a set of states where the failure formula is satisfied.

The input of the FtA leaves express the failure of a component, such as 'FC1' in Table 3. Thus, the faliure logic expression resulted from the failure logic expression of the FtA is the combination of component APs, which explains how the components breakdown propagates to the whole system breakdown according to the causality of the system service. And these APs are connected by 'and' logic ('$\wedge$') resulted from the 'and' gate in FtA, and connected by the 'or' logic ('$\vee$') resulted from the 'or' gate in FtA. The 'failure logic expression' is used as a state identification property. In particular, the Total Breakdown State (TBS) is able to be identified in the global EDA obtained. Based on the definition of the reliability model, it features both an initial state and a final state, the TBS. So, the TBS in GFA should be operated to be absorbing state (by deleting the output transitions according to the transition function), in order to further generate the GRA.

This part is the statement of the FtA modeling principle for the structure analysis, and the failure logic expression is the TBS identification property. However, not only FtA, all the other structure analysis tools which are able to obtain the failure logic expression are suitable for our proposal. The 'parallel' and 'series' are two basic elements in the system structure. The upper side of Figure 28 introduces FtA is applied for the Series Structure, and the two components C1 and C2 is working by the series collaboration form. The resulting FtA for series structure is by the 'or' gate connecting the two leaves, and these two leaves are the AP element $\boldsymbol{F_{C1}}$ and $\boldsymbol{F_{C2}}$ (already listed in Table 3). By FtA, the failure logic expression is obtained: $\Phi 1=\boldsymbol{F_{C1}}\vee\boldsymbol{F_{C2}}$. In the lower side, the system is of the Parallel Structure. The equivalent FtA for the parallel structure is the 'and' gate connecting $\boldsymbol{F_{C1}}$ and $\boldsymbol{F_{C2}}$. The logic failure expression is: $\Phi 2=\boldsymbol{F_{C1}}\wedge\boldsymbol{F_{C2}}$. The failure logic expressions are the TBS state identification property for Series and Parallel structure.

2.2.1.6 TBS identification

The TBS identification is performed on the global EDA, issued from the composition of all components of the structure at hand. In parallel, the FtA yields a failure logic formula Φi. A The following is the identification formula. As presented previously, the TBS idenfitication amounts to computing the set of states satisfying Φi. For an identified state set IS(Φi) $\subset Q$, its states satisfy the identification property Φi according to the following rule:

$$IS(\Phi i)=\{x|\ x\in Q,\ L(x)\Rightarrow \Phi i\} \qquad (2\text{-}5)$$

IS(Φi) is a subset of state set Q: IS(Φi)⊆Q. If no state AP satisfies the identification property, IS(Φi) is an empty set IS(Φi. An algorithm to judge the state AP to be a sufficient condition of the identification property is studied as the following.

Algorithm: state identification
A state AP (expressed by n elements) is already known as: $F_1 \wedge F_2 \wedge F_3 \wedge \ldots \wedge F_i \ldots \wedge F_n$; Assume: F_i is the arbitrary element of the AP: If F_i is in its negation form: $F_i=0$ If F_i is in its positive form: $F_i=1$; An identification property Φis set, using $F_1,F_2,F_3 \ldots\ldots F_n$; Based on the value of { F_i }: If the value of Φ is caculated to be '1': AP is the sufficient condition ofΦ, this state is identified to be TBS If the value of Φ is caculated to be '0' : AP is not the sufficient condition ofΦ, this state is identified not to be TBS ;

For example, in Figure 28 upper side, according to the failure logic expression $\Phi 1=\boldsymbol{F_{C1}} \vee \boldsymbol{F_{C2}}$ one of the identified states is $B_{C1}.W_{C2}$. The AP is : $L(B_{C1}.W_{C2})= \boldsymbol{F_{C1}} \wedge \neg \boldsymbol{F_{C2}}$. Based on equation (2-5), it is known $L(B_{C1}.W_{C2})$ is the sufficient condition of Φ1, thus $L(B_{C1}.W_{C2}) \vDash \Phi 1$. The proof is: based on the Algorithm, in the state AP, $\boldsymbol{F_{C1}}$ is the positive form, thus $\boldsymbol{F_{C1}}=1$, and $\boldsymbol{F_{C2}}$ is the negation form, thus $\boldsymbol{F_{C2}}=0$. The value calculation of Φ1 is : $\Phi 1=\boldsymbol{F_{C1}} \vee \boldsymbol{F_{C2}}= 1 \vee 0=1$. Φ1 is true, which means the state AP makes the state identification true and this state is identified.

According to Table 4 and equation (2-5), the identified state set is: in series structure IS(Φ1)={ $B_{C1}.W_{C2}$； $W_{C1}.B_{C2}$； $B_{C1}.B_{C2}$} and in parallel structure IS(Φ2)={ $B_{C1}.B_{C2}$ }. The identified states are TBS marked in the red circle in Figure 228. The principle of Reliability model generation is to operate the TBS to be absorbing in GFA. Based on Table 2, the output transitions of the TBS should be deleted, for example, $B_{C1}.W_{C2}$ is one of the TBS states in Figure 228 upper side part and the output

transitions are $\delta_{C1//C2}$ ((B_{C1}. W_{C2}), repair1)= (W_{C1}.W_{C2}) and $\delta_{C1//C2}$ ((B_{C1}.W_{C2}), failure2)=(B_{C1}.B_{C2}) and they should be deleted. Finally, after deleting the output transitions of TBS, the corresponding GRA model is shown in Figure 228.

2.2.2 Example: GRA Generation

According to the framework introduced in the previous part, an example for the GRA generation is studied in this section. Two major points are presented here, the first is the fact that 'N-state behaviors' are able to be considered. The second is the application of the method presented for structure analysis by using FtA. To be readable and easily understandable, only three components are used in a series-structure system (a more complex benchmark will be presented in the next chapter). As for the 'N-state behaviors' demand, the component model is able to be modeled by various states (not only working and breakdown states). These various states are able to be considered in GFA, by the composition operation of component-models that is similar to the 'unique events'. So, if the composition is achievable as well as 'unique transition' is achievable, the 'N-state behaviors' is also achievable. Also for the reason that 'to be readable and easily-understandable', only 'unique transition' is selected to be shown in this section. The 'N-state behaviors' will be presented in the next chapter.

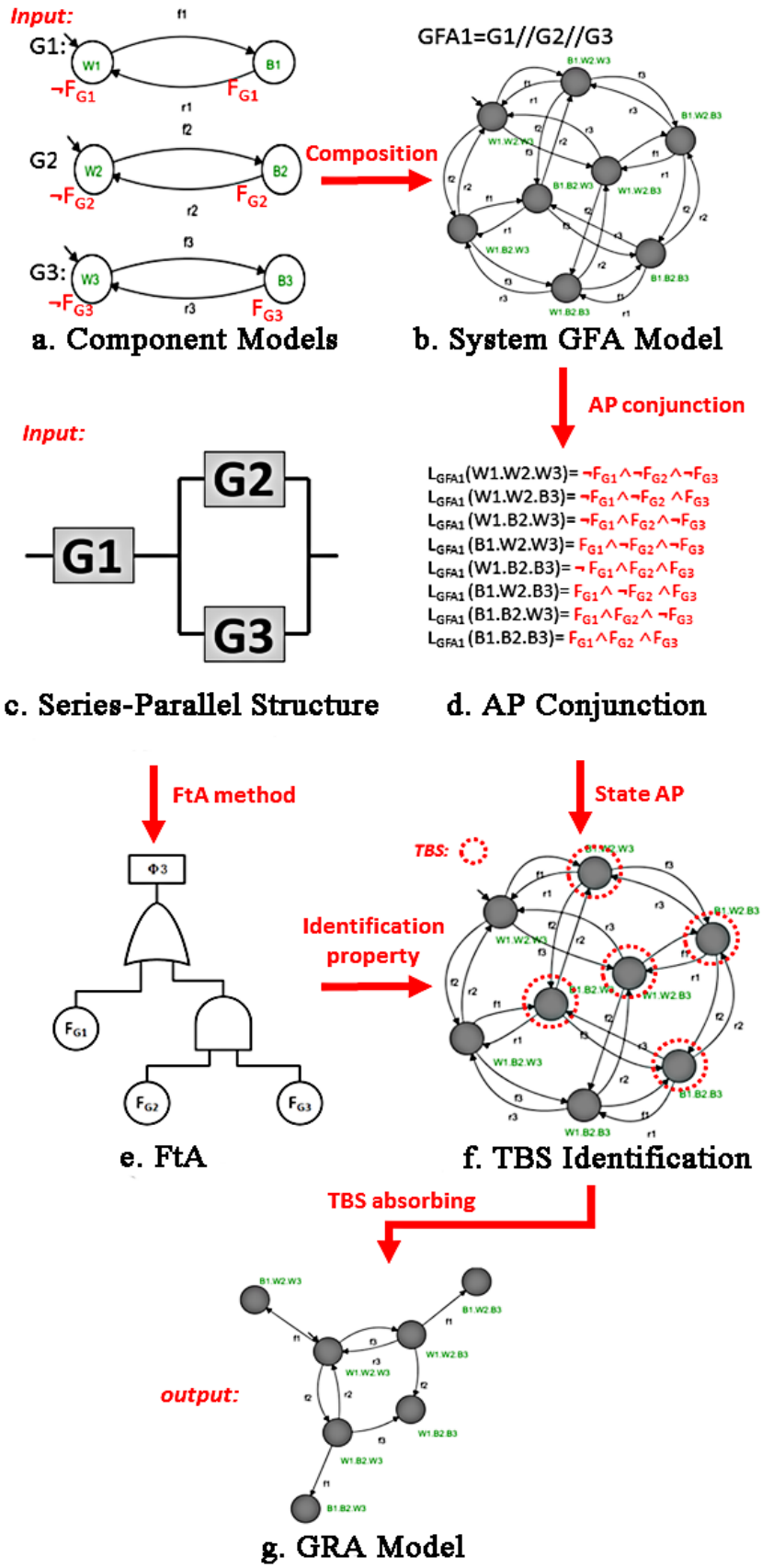


Figure 29 Benchmark: Consideration of Higher-level Reliability

Three components G1, G2 and G3 are designed: states W1, W2, and W3 stand for the system healthily working states; states B1, B2, and B3 stand for the system breakdown states; each component has a unique failure behavior of f1, f2, and f3, and a unique repair operation of r1, r2 and r3. Boolean AP set {F_{G1}; F_{G2}; F_{G3}} are used to denote the broken down of the components, while the negative symbolic of these AP presents the components healthily working.

$G1= (Q_{G1}; E_{G1};\delta_{G1};q0_{G1}; qm_{G1};AP_{G1};L_{G1})$

$=(\{W1,B1\};\{f1,r1\};\delta_{G1}(W1,f1)=B1, \delta_{G1}(B1,r1)=W1;W1; B1;\{ F_{G1}, \neg F_{G1}\}; L_{G1}(W1)= \neg F_{G1},$ $L_{G1}(B1)= F_{G1})$

$G2= (Q_{G2}; E_{G2};\delta_{G2};q0_{G2}; qm_{G2};AP_{G2};L_{G2})$

$=(\{W2,B2\};\{f2,r2\};\delta_{G1}(W2,r2)=B2, \delta_{G1}(B2,r2)=W2;W2; B2;\{ F_{G2}, \neg F_{G2}\}; L_{G2}(W2)= \neg F_{G2}$ $L_{G2}(B2)= F_{G2})$

$G3= (Q_{G3}; E_{G3};\delta_{G3}; q0_{G3}; qm_{G3};AP_{G3};L_{G3})$

$=(\{W3,B3\};\{f3,r3\}r;\delta_{G3}(W3,f3)=B3, \delta_{G3}(B3,r3)=W3;W3; B3;\{ F_{G3}, \neg F_{G3}\}; L_{G3}(W3)= \neg F_{G3},$ $L_{G3}(B3)= F_{G3})$

As it is introduced that the GFA model describes the system by all the possible cases (every detailed component behaviors are considered), and the GFA is the composition result of the component-models. So, GFA1=G1//G2//G3 (this composition operation is issued by Supremica[113]). The AP conjunction result via the EDA composition is shown in red bold in Figure 29 <AP conjunction>, based on equation (2-4).

For the reason that the reliability model starts with the initial state and stops at the system totally breakdown state, the GRA is issued from the GFA, by identifying the system total breakdown states (TBS) and absorbing them. To know how the components breaking down leads to the total breakdown of the system, the FtA is applied to analysis the structure to obtain the 'failure logic expression'. The 'failure logic expression' is the TBS state identification property, as every state is already associated with AP. Shown in Figure 28 <FtA>, FtA uses an "and" gate connecting the breakdown(denoted by AP) of G2 and G3 because of the parallel structure; using an "or" gate connecting the breakdown of G1 and the previous "and" gate because of the series structure, thus the logic failure expression is $\Phi 3=F_{G1} \vee (F_{G2} \wedge F_{G3})$.

In order to ensure the formal correctness and provide an automatic processing ability, a GRA Generation Tool was created in this research. This software mainly has 4 functions:

1, TBS Identification, based on equation (2-5)

2, TBS absorbing, delete the output transitions of TBS state to generation the GRA

3, Taken access part from the initial state. As the same output transitions have been deleted, it is possible that several states become unreachable from the initial state. These states are meaningless in GRA. By this function, the unreachable states are deleted.

4, Output GRA model. The GRA model is saved as 'xml' document.

By using this program to treat the state of GFA1, the identified states are IS(Φ3)={ B1.B2.B3; B1.B2.W3; B1.W2.B3; B1.W2.W3; W1.B2.B3}, and they are marked in the red circle. These TBS are further absorbed to generate GRA1 and then the unreachable states are deleted. GRA1 is shown in Figure 29 <Reliability Model>.

2.3 Consideration of Operational Dependability

In the previous sections, operational reliability was considered by targeting the TBS inside the GFA and by considering it as a final state. This excludes any repair operation for a totally broken-down system. Yet it is possible to extend the current framework towards handling dependability. Repair trajectories are part of the EDA models and thus can be easily included when targeting the TBS. Hence, the consideration of operational dependability is achieved by planning possible repair trajectories in the global EDA model. Other management requirements can be specified in order to express repair priorities, maintenance for the service, etc. The resulting model is expressed as an additional EDA, to be composed with the global GFA model. The operational dependability can be assessed, qualitatively and quantitatively, relying on the resulting model.

It is important to emphasize that this is one of the important highlights of this proposal, in relation to the corpus of methods for assessing operational safety. Moreover, its formal aspect offers the ability to introduce the formal verification of properties such as liveliness, reachability, not developed at the current status of this work but totally reachable as perspectives. Besides, this notion of dealing with the structure on the one hand and the operational constraints on the other hand, remains close to existing engineering methods and tools.

2.3.1 Management requirements specifying approach

In the domain of Discrete Event System Theory application, the specification automaton is designed to contain the specifying information, and by the help of composition operation of system model and ETS (Event Trajectory Specification Automaton), the management requirements are able to be specified.

The specification automaton is defined:

$$ETS=(Q_{ETS};E_{ETS};\delta_{ETS};q0_{ETS};qm_{ETS})$$

Q_{ETS} is the state set, E_{ETS} is the transition set, δ_{ETS} is the transition function, $q0_{ETS}$ is the initial state and qm_{ETS} is the marked state.

Then result of the composition operation (denoted by '//')[111] is named to be specified automaton (denoted by 'SPA'):

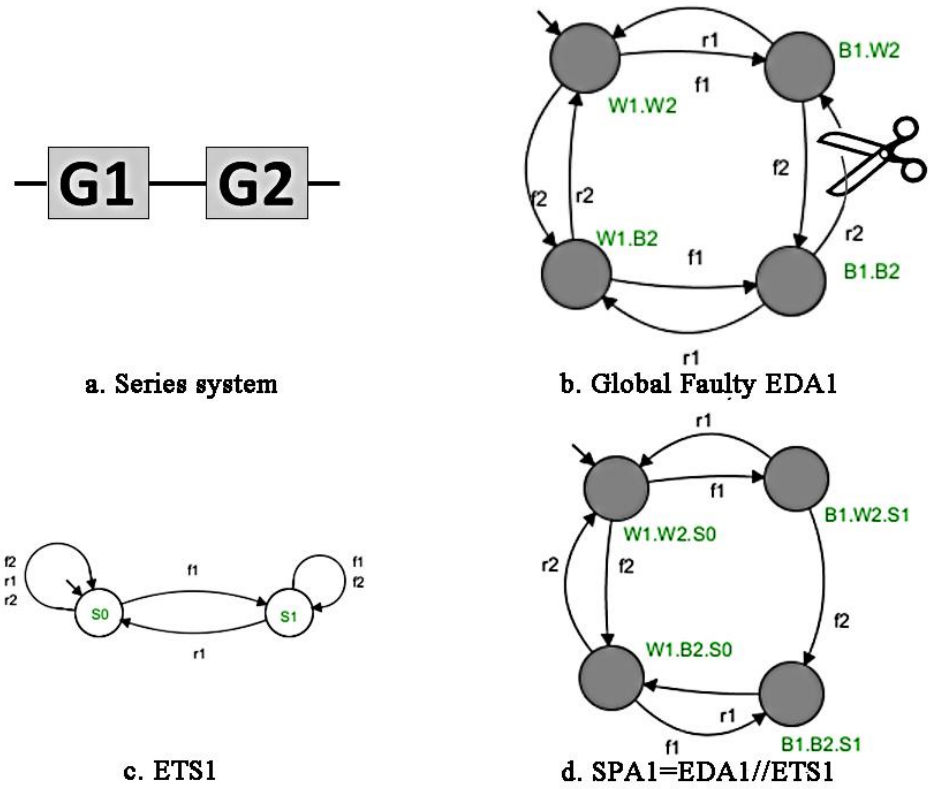
$SPA=(Q_{SPA};E_{SPA};\delta_{SPA};q0_{SPA};qm_{SPA})$

$= EDA // ETS$

$=(Q_{EDA}\times Q_{ETS};E_{EDA}\cup E_{ETS};\ \delta_{SPA};\ q0_{EDA}\times q0_{ETS};\ qm_{EDA}\times qm_{ETS})$

Where, Ac stands for taking the accessible part, and symbol '×' is to denote Cartesian product between the state sets or transition event sets[114].The composition operation is is already presented in equation (2-3). Thus, the transition function of EDA and ETS is further explained:

$$\delta_{SPA}((x.y),e)=\begin{cases}(\delta_{EDA}(x,e)\ .\delta_{ETS}(y,e)) & if\ e\in E_{EDA}\cap E_{ETS}\\ (\delta_{EDA}(x,e)\ .\ y) & if\ e\in E_{EDA}\backslash E_{EDA}\cap E_{ETS}\\ (x\ .\ \delta_{ETS}(y,e)) & if e\in E_{SP}\backslash E_{EDA}\cap E_{ETS}\\ undefined & otherwise\end{cases}$$



a. Series system

b. Global Faulty EDA1

c. ETS1

d. SPA1=EDA1//ETS1

Figure 30 "G1 has the priority to repair" Model Specifying

Consider the system depicted in Figure 0 as an example. Assume a management requirement "The important component owns the priority to be repaired first". When both the two components are broken down, G1 has the priority to be repaired first.

Shown in Figure , on the right top part, is the GFA model of this system：

$$EDA1=(Q_{EDA1};E_{EDA1};\delta_{EDA1};q0_{EDA1};qm_{EDA1})$$

Q_{EDA1}= {W1.W2, W1.B2, B1.W2, B1.B2} these states respectively standing for: both G1 and G2 are working; G1 is working but G2 is broken down; G1 is broken down but G2 is working; Both G1 and G2 are broken down. E_{EDA1}= {f1, f2, r1, r2} these events respectively standing for: the failure event of G1; the failure event of G2; the repair operation of G1; the repair operation of G2. According to the management requirement, when both components are broken down (state: B1.B2), the first repair operation should be distributed to G1, so the transition $B1.B2 \xrightarrow{r1} W1.B2$ is legal, but the transition $B1.B2 \xrightarrow{r2} B1.W2$ is illegal. The illegal transition should be dismissed to satisfy this requirement.

Shown in Figure 0 the lower left part: the specification automaton is:

$$ETS1=(Q_{ETS1};E_{ETS1};\delta_{ETS1};q0_{ETS1};qm_{ETS1})$$

Here, Q_{ETS1}={S0,S1}; E_{ETS1}={f1,f2,r1,r2};$q0_{ETS1}$={S0}; qm_{ETS1}=∅ (here, the marked state is not necessary) ETS1 is designed as: the self-loop events of S0 {f2,r1,r2}, drives no state evolution from S0 to S1, considering these events do not break the management requirement; from S0 to S1 is event {f1}, considering "in case f1 happens"; the self-loop of S1 is {f1,f2}, where there is no repair operation r1 or r2; from S1 to S0 is the repair operation r1, considering after f1 happens it should repair G1 and then turns to initial (the repair of G2 is forbidden here). By this design, the illegal transition (repair G2, when both components breakdown) is dismissed and the management requirement will be satisfied, which is shown in Table 5.

The composition result of ETS1 and EDA1 is:

$SPA1=(Q_{SPA1};E_{SPA1};\delta_{SPA1};q0_{SPA1};qm_{SPA1})$

= EDA1 // ETS1

$=Ac(Q_{EDA1}\times Q_{ETS1};E_{EDA1}\cup E_{ETS1};\ \delta_{ETS1};\ q0_{EDA1}\times q0_{ETS1};\ qm_{EDA1}\times qm_{ETS1})$

And $Q_{EDA1}\times Q_{ETS1}$={W1.W2.S0, W1.B2.S0, B1.W2.S1, B1.B2.S1}, $E_{EDA1}\cup E_{ETS1}$={f1,f2,r1,r2}, $q0_{EDA1}\times q0_{ETS1}$=W1.W2.S0. As there is no necessary to denote the marked state in this application, $qm_{EDA1}\times qm_{ETS1}$=∅.

Based on composition definition, and for the reason that the event set: E_{EDA1}=E_{ETS1}, the transition function belongs to the case $e \in E_{EDA} \cap E_{\mathrm{ETS}}$. Transition function is: $\delta_{SPA1}((\mathrm{x.y}),\mathrm{e}) = (\delta_{EDA1}(x,e)\ .\delta_{\mathrm{ETS1}}(y,e))$, $here\ e \in E_{EDA1} \cap E_{\mathrm{ETS1}}$. Its transitions are listed in Table 5. This SPA1 satisfies the management requirement by there is no repair event (r2) from state(B1.B2.S1) to state(B1.W2.S1), which means when G1 and G2 both breakdown, the only allowed repair is r1. A proof is issued by the calculation of $\delta_{SPA1}((\mathrm{(B1.B2).S1}),\mathrm{r2})$. According to ETS1, if the specified result makes $\delta_{SPA1}((\mathrm{(B1.B2).S1}),\mathrm{r2})$ to be no state, the management requirement is satisfied and ETS1 works.

As it is known: δ_{ETS1}（S1,r2）=∅ ;

δ_{EDA1}（（B1.B2）,r2）=B1.W2 ;

Then:

$\because \quad r2 \in E_{EDA1} \cap E_{\mathrm{ETS1}} = \{f1, f2, r1, r2\}$

$\therefore \quad \delta_{SPA1}((\mathrm{x.y}),\mathrm{r2}) = (\delta_{EDA1}(x,r2)\ .\delta_{\mathrm{ETS1}}(y,r2))$

Here, x=(B1.B2) and y= S1

$\delta_{SPA1}((\mathrm{(B1.B2).S1}),\mathrm{r2}) = (\delta_{EDA1}((B1.B2),r2)\ .\delta_{\mathrm{ETS1}}(S1,r2))$

$=((B1.W2)\,.\emptyset)\ \notin E_{SPA1}$ here, state$((B1.W2)\,.\emptyset)$ does not exist

In the upper deduction, it is proofed that there is no repair transition (r2) from (B1.B2.S1) to (B1.W2.S1) in SPA1, which means when G1 and G2 both breakdown, the repair of G2 is forbidden and the only allowed repair is G1.

In summary, if there is management requirement specifying demand, the specification automaton ETS is designed, and the composition of EDA and ETS is able to specify the management requirement in the model. What should be emphasized is: the composition of the EDA and ETS is able to specify the corresponding management requirement onto the EDA, so the composition of ETS with GFA or the composition of ETS with GRA are both able to handle the specifying task. For the establishment of Operational Reliability Model, one solution is to first compose the ETS with the GFA model and then identify TBS to establish the Reliability model; the other solution is to first identify the TBS in the GFA to establish GRA and then compose the ETS with GRA. These two solutions illustrate the same specification result.

Table 5. Management requirement satisfying transition function of SPA1

SPA1 transition function	Transition event	EDA1 transition function	ETS1 Transition function
δ_{SPA1}（（W1.W2.S0）,f1）=B1.W2.S1	f1	δ_{EDA1}（（W1.W2）,f1）=B1.W2	δ_{ETS1}（S0,f1）=S1
δ_{SPA1}（（W1.W2.S1）,r1）=W1.W2.S0	r1	δ_{EDA1}（（W1.W）,r1）=W1.W2	δ_{ETS1}（S1,r1）=S0
δ_{SPA1}（（W1.W2.S0）,f2）=W1.B2.S0	f2	δ_{EDA1}（（W1.W2）,f2）=W1.B2	δ_{ETS1}（S0,f2）=S0
δ_{SPA1}（（W1.B2.S0）,r2）=W1.W2.S0	r2	δ_{EDA1}（（W1.B2）,r2）=W1.W2	δ_{ETS1}（S0,r2）=S0
δ_{SPA1}（（W1.B2.S0）,f1）=B1.B2.S1	f1	δ_{EDA1}（（W1.B2）,f1）=B1.B2	δ_{ETS1}（S0,f1）=S1
δ_{SPA1}（（B1.B2.S1）,r1）=W1.B2.S0	r1	δ_{EDA1}（（B1.B2）,r1）=W1.B2	δ_{ETS1}（S1,r1）=S0
δ_{SPA1}（（B1.W2.S1）,f2）=W1.B2.S1	f2	δ_{EDA1}（（B1.W2）,f2）=W1.B2	δ_{ETS1}（S1,f2）=S1

2.3.2 Example: expressing operational management specifications

As it is not possible to present all the management requirements for all the applicated situations, this section offers an example for three widely used managements: Downtime Maintenance (denoted by ‘DM’), Priority to Repair (denoted by ‘PtR’) and First Failed First Repaired (denoted by ‘FFFR’). However, in reality, system-designers can design their own ETS for various specifying demands.

This example is applied to a series-parallel system, G1 has the PtR for its relatively important position because of the series linking, G2 and G3 are repaired satisfying FFFR because G2 and G3 are lying in similar positions. The whole system needs a DM (‘test’ behavior for begin to maintenance and ‘recover’ behavior for system recover to work), these management operation requirements are specified for the Reliability model obtained in the previous part (the reliability model in the benchmark in Figure 29).

2.3.2.1 Modeling downtime maintenance scenarii

According to Figure 31, the specification of such a scenario genarally amounts to updating an existing failure/repair specification which does not express maintenance yet. Thus, a new specification state is added into it, and this state is used to model a maintenance operation. It is usual to express maintainance scenarii exclusively with respect to the existing scenarii already modeled: the specification can run either one or the other. The resulting specification, the Downtime Maintenance Evant trajectory

Specification (DM ETS) is an automaton enriched with the new maintenance scenario. Thus, from the initial state, the system can switch to the maintenance state. After maintenance, driven by a 'recover' operation, the system switches back to its operational state, from where either failures or further maintenance may occur.

Hence, according to Figure 64 <DM>, the specification automaton is designed by the principle of 'keep the model as original and adding a maintenance state into the model': First step, add a maintenance state from initial state, which has the function of 'adding a maintenance state into the model'; Second step, design the transition between states S0, S2, S3, S4 are {f1,f2,f3}and {r1,r2,r3}, this step is to describe the model " From initial state, it stochastic happens the failure of G1, G2 and G3 as well as the repair operations ", which has the function of 'keep the model as original'. Based on equation (2-1), the composition result of this specification automaton and the original system-automaton is shown in Figure 64 <SPA=System EDA//PtR//DM//FFFR>, this result satisfies the management requirement by adding a state (W1.W2.W2.maintenance.S0.S0) standing for the system Downtime Maintenance operation driven by 'test' and 'recover' event.

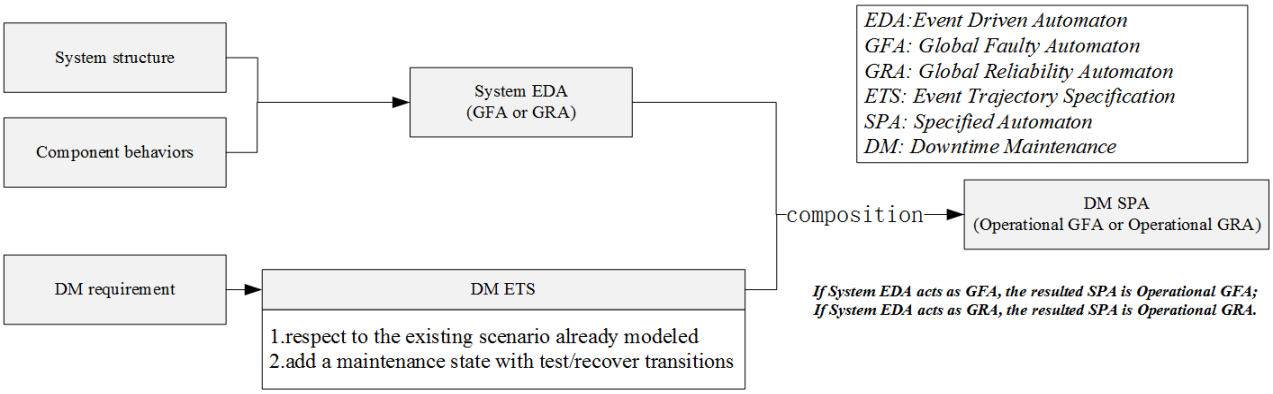


Figure 31 Downtime maintenance specifying approach

2.3.2.2 Repair priority for one single component

Let $C_1, C_2, \dots C_N$ be N components. Let $C_j, 1 \leq j \leq N$ be a most prioritary component. This repair policy recurrently states that:

- if C_j is broken down, and $\{C_i, i \neq j\}$ are operational, then C_j may be repaired;
- if C_j is broken down, and some $\{C_i, i \neq j\}$ are also broken down, C_j may be repaired;
- if C_i is operational and some $\{C_i, i \neq j\}$ are broken down, any C_i may be repaired, and this same rule re-applied until the set of broken components is empty.

According to Figure 32, the modeling of the priority to repair specification has mainly 4 steps: first, add a priority state with the input transition of the failure of the most priority state, which identify the

case that the most priority component is brookendown; Second, use the self-loop transitions of the priority state to forbidden all the repair operations of all the other components; Third, set the output transition of the priority state to be the repair operation of the most priority state, which guarantees the only allowed repair operation should be distributed to the most priority component; Fourth, respect all the other exsiting scenario in the EDA model to be unchanged, inorder to guarantee the other failure/repair scenario to be uninfluenced. These rules are already illustrated in Figure 2, using the example of a series structure.For this example, let component G1 have the priority to be first repaired. The specification automaton 'Priority to repair' <PtR> in Figure 64 models this aspect: in case the failure of G1 happens (the transition: $S0 \xrightarrow{f1} S1$, in PtR), no repair operations of G2 or G3 should be achieved and the only allowed repair is r1.

After the composition operation, the specified model <SPA=System EDA//PtR//DM//FFFR> satisfies this PtR Management requirement, in all possible cases: component G1 breaks down, the first repaired component is always G1, until component G1 is repaired, G2 and G3 are allowed to be repaired.

2.3.2.3 'First Failed First Repaired' requirements

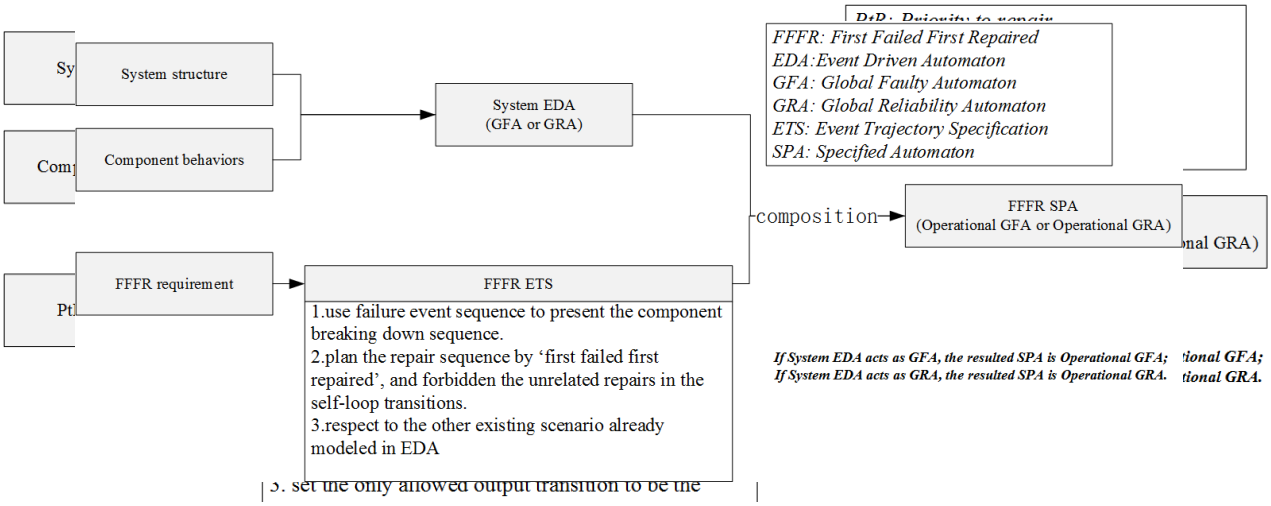


Figure 53 First failed first repaired specifying approach

modeled in EDA

Figure 42 Priority to repair specifying approach

This repair policy is very similar to the previous one, except that $j, 1 \leq j \leq N$ can change each time C_j is repaired. The specification model becomes a little more complex, as it requires to express all the possible shuffles between failure and repairs. According to Figure 33, the solution is mainly issued by using failure sequence to present the breaking down sequence of the components and then planning the related repair sequence.

It is illustrated in the 'First failed first repaired' <FFFR> part of Figure 64. Components G2 and G3 are respecting "First Failed First Repaired", the specification automaton is depicted in <FFFR>Figure 64. The FFFR priority mechanism is designed as: from the initial state there happen two failure events of G2 and G3. The failure sequence $S0 \xrightarrow{f2} S1 \xrightarrow{f3} S3$. For state S3, G2 fails before G3, so the output repair operation is designed as $S3 \xrightarrow{r2} S2 \xrightarrow{r3} S0$, organizing G2 repaired before G3; The failure sequence $S0 \xrightarrow{f3} S2 \xrightarrow{f2} S4$. For state S4, G3 fails before G2, so the output repair operation is designed as $S4 \xrightarrow{r3} S1 \xrightarrow{r2} S0$, organizing G3 repaired before G2. Moreover, the self-loop of state always contains the behaviors {f1, r1} of G1, and this self-loop illustrates the specification automaton only consider G2 and G3, without any consideration of G1 (no transition function executed for f1 and r1). The composition result shown in <SPA=System EDA//PtR//DM//FFFR> satisfies this management requirement.

- **Discussion**

The framework presented above features several advantages:

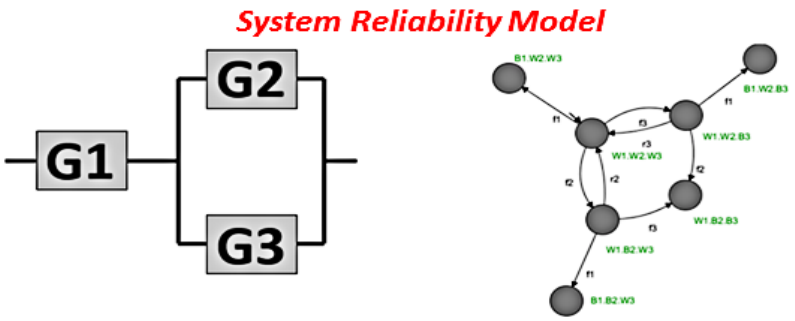
- offer an event-based dynamics point of view, which allows focusing on behavior prior to switching to the quantitative figures and their subsequent analysis. We advocate the fact that this approach is less error-prone in the expression of the system CTMC model;

- it is able to automatically build complex models up from simple component models. This can be achieved thans to the component product operation;

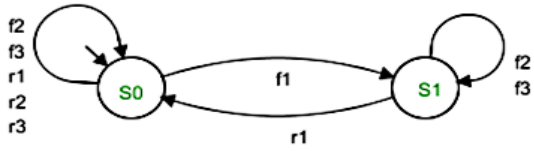
- it is able to 'weave' specific desired behaviors into the global system model. These behaviours express operational requirements such as repair policies, which have been illustrated above. Other policies can be easily imagined as long as they can be represented as EDAs. This operation also brings an important improvement as it is automatic and produces correct-by-construction results. They are derived faster, with much less engineering effort and reduced risk of manually introducing errors in the system model;

- thanks to the event-based representation, interesting properties such as reacheability or liveness can be mechanically checked or even enforced.

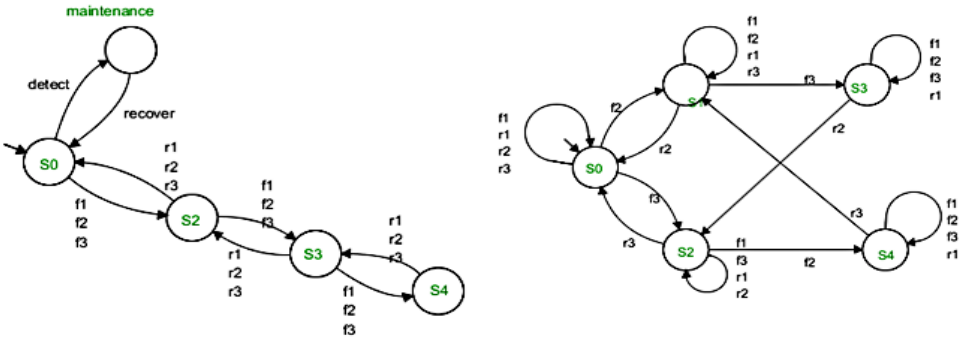
Combination explosion in the size of the models is the main limitation of the proposed approach. Still, it offers engineers a safe tool for modeling and building, able to reach higher complexities than by the conventional manual approach.



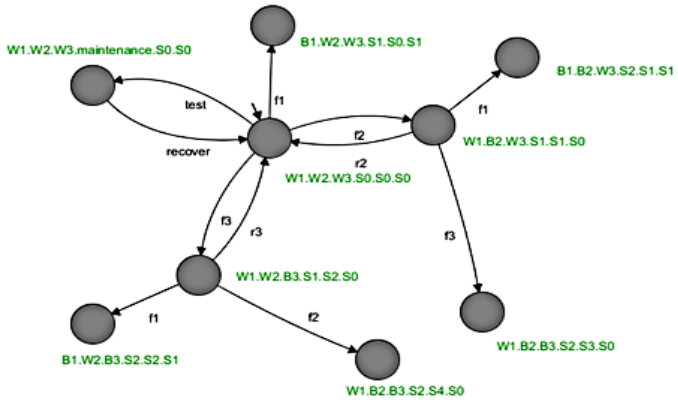
a. Series-Parallel System b. System EDA(GRA)



c. PtR



d. DM e. FFFR



f. Operational GRA=SPA=System EDA//PtR//DM//FFFR

Figure 64 Management specifying benchmark for DM, PtR and FFFR

2.4 Capacity Calculation Fault Tree (CCFT)

2.4.1 Introduction

As it has already been mentioned ‘the management requirements are set for the purpose to offer the optimal operation for the system’. But a problem appears in order to answer the question: ‘To handle with the optimal operation, which management requirement is more suitable for the system?’. In this section, this problem restricted to the service capacity enhancement (production aims) and is solved by applying a new quantitative simulation method named by CCFT (Capacity Calculation Fault Tree). The CCFT is able to calculate the system capacity with the consideration of system partial breakdown, by which it is able to simulate ‘based on a certain management requirement, to what level the system recovers’.

The redundancy designing of the system guarantees that in case several components break down, the system is still able to deliver its service with a lower production capacity. As this kind of system becomes more complex, it is difficult to calculate the system capacity considering several components have broken down. Thus, a formal solution to solve this problem is required. FtA (Fault tree Analysis) is an efficient tool to analyze the system structure. The system production capacity calculation approach (CCFT) is issued by the association of FtA and the component production capacity. This approach is not only able to analyze the safety-critical system as the normal FtA does, but is also able to calculate the system dynamic production capacity with the consideration that the system is partially broken down. It integrates into the proposed engineering approach dedicated to MBSA, completing the palette of tools which can be used in synergy with a quantitative tool enforcing the overall utility.

The modeling principle of CCFT is proposed, in 2.3.3.2.

2.4.2 Modeling Principle of CCFT

The same as FtA, the CCFT model is illustrated by two basic gates ‘and gate’ and ‘or gate’. Depicted in Figure 35, two wind turbines G2 and G3 are working in the parallel form in a miniplant denoted by ‘sys1’, and each turbine has its electric power production. The system is shown in the left side (the production of G2 and G3 is respectively 3MW/h and 1.5MW/h), in the middle is the system structure, and in the right side is the CCFT of this system. In the left leaf of CCFT, FG2 denotes the Failure Boolean Logic of G2 (FG2=1 stands for component G2 broken down; FG2=0 stands for component G2 not broken down). Here, ‘WG2=3MW/h| FG2=0’ means if FG2=0 (G2 is not broken down), G2 is able to contribute to its production by the power value (denoted by WG2) of ‘3MW/h’, and ‘WG2=0W/h | FG2=1’ means if FG2=1(G2 is broken), G2 is not able to contribute its function, and its production power is ‘0’. It is the same explanation for the right leaf. These two leaves are connected by an “and gate” as the normal FtA. So the failure Boolean logic presentation of the system Sys1 is:

Fsys1=FG2∧FG3. In case that system is not broken down (Fsys1=0), the system production capacity(Wsys) is the summation value of WG2 and WG3. If the system is broken down (Fsys1=1), the production capacity is '0'. Table 6 presents the CCFT calculation result of the Wsys1 according to the possible broken-down-situations of the system.

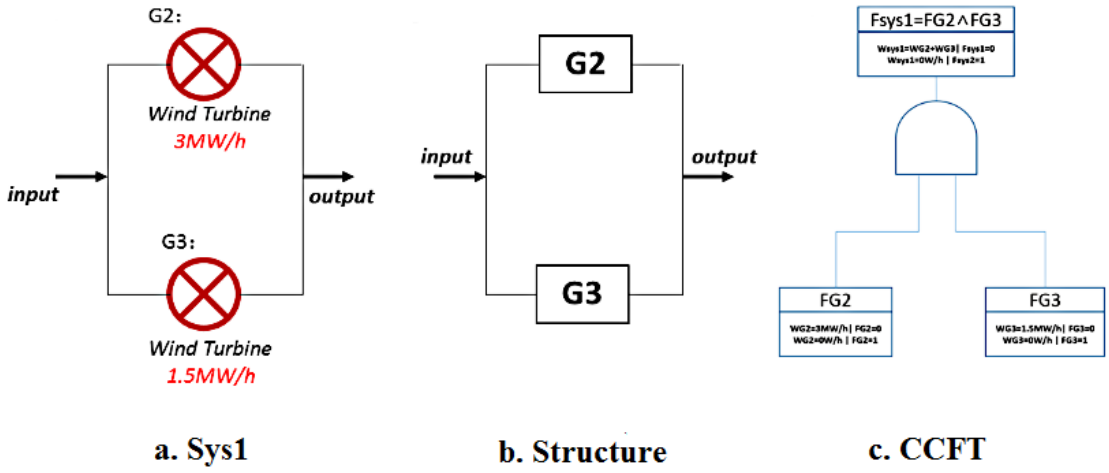


Figure 35 'and' gate of CCFT for parallel structure

Table 6 CCFT Validation of system Sys1

FG2	WG2(MW/h)	FG3	WG3(MW/h)	Fsys1	Wsys1(MW/h)
0	3	0	1.5	0	4.5
0	3	1	0	0	3
1	0	0	1.5	0	1.5
1	0	1	0	1	0

Shown in Figure 736, the system (denoted by Sys2) is constructed by a wind turbine G1 and a converter G4 working in the series structure, the left side is the system presentation (the production of G1 and G4 is respectively 1MW/h and 0MW/h), in the middle is the system structure, and in the right side is the CCFT. The same as normal FtA, to model the series structure, the leaves of G1 and G4 are connected by a 'or' gate. To be emphasized, the converter G4 is not a power producer whose output power is always '0' (whether FG4=1 or FG4=0, WG4=0). If the system is not broken down (wind turbine G1 and converter G4 are not broken down), the system is able to contribute to its function. If the system is broken down (Fsys2=1), it is not able to deliver its function. The calculation result is shown in Table 7 CCFT Validation of system Sys2.

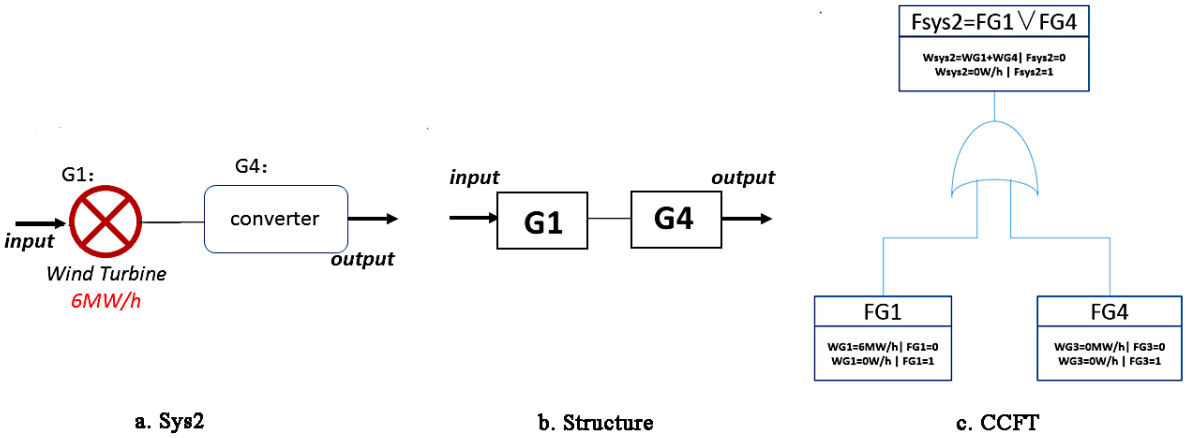


Figure 76 'or' gate of CCFT for series structure

Table 7 CCFT Validation of system Sys2

FG1	WG1(MW/h)	FG4	WG4(MW/h)	Fsys2	Wsys2(MW/h)
0	6	0	0	0	6
0	6	1	0	1	0
1	0	0	0	1	0
1	0	1	0	1	0

The following is the algorithm of CCFT to calculate the system capacity. For the components, if the component breaks down, the component production capacity is zero. If the component is working healthily, the component contributes to its production capacity. Based on FtA, if the input is several components have broken down, the system is broken down or healthily working will be known. So, In case the system is healthily working, the production capacity of the system is the summation of the component capacities. In case the system is broken down, the production capacity of the system is zero.

Algorithm: CCFT calculation
Assume: The system is constructed by several component:G1,G2,G3…Gi…Gn System has the production capacity: Wsys Component Gi has the production capacity: W_Gi Component Gi has failure Boolean symbolic: F_Gi The system failure logic expression is obtained by FtA: φ For the components: If Gi is broken down: F_Gi=TRUE W_Gi=0

If Gi is healthily working:

F_Gi=FALSE

For the system:

If the components Boolean Symbolic {F_G1… F_Gi ... F_Gn } makes φ=TRUE:

Wsys=0

If the components Boolean Symbolic {F_G1… F_Gi ... F_Gn } makes φ=FALSE:

Wsys=$\sum_{i=0}^{n}$ W_Gi

2.4.3 Effectiveness Simulation method for the management requirements

The effectiveness of the management requirement is able to tell whether the management requirement is suitable for the system or not, and it is of three elements 'economic cost', 'time cost' and 'system capacity'. The 'economic cost' means the money paid for the repair operation. The 'time cost' means the downtime of the components (not working time). In this section, an example of power plant system is studied in order to present the simulation methodology.

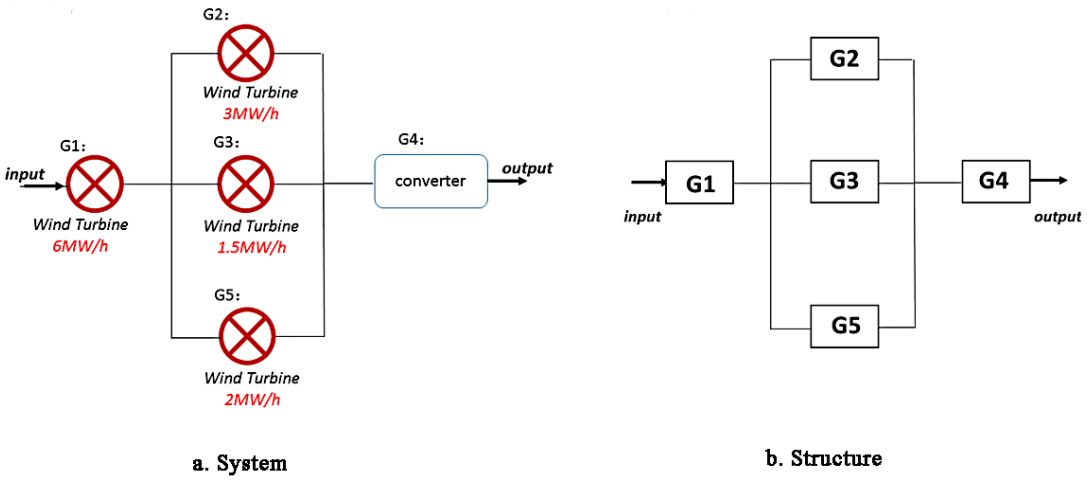


Figure 37 power plant system

Depicted in Figure 37, there is a power plant system (left side is the system presentation, and the right side is the structure), there are 4 wind turbines and a converter, the average output power of the wind turbines are 6MW/h, 3MW/h, 1.5MW/h and 2 MW/h (the converter does not produce the capacity). These components are possible to be broken down. It is easier to know the full production capacity (all the components are working healthily) of this system: Wsys=WG1+WG2+WG3+WG5=12.5MW/h. However, the hot redundancy designing of G2, G3 and G5 contributes to the system dependability to be: in case one or two components break down, the system is still able to deliver the power production service with a lower production capacity. For example, if G2 is broken down, the system is partially but not totally broken down, and the production capacity now is: WG1+WG3+WG5. In this system, each

repair of these components (G2, G3 and G5) should pay for a repair cost, which forms the 'economic cost'. Each failure and the repair operation of these three components is assumed to has a time period, 'Tf' is used to denote the 'time of failure', 'Tr' is used to denote the 'time of repair', which forms the 'time cost'. These parameters of the components are declared in Table 8.

Table 8 Parameters of the components in power plant system

Name / Parameter	G1	G2	G3	G4	G5
Production capacity(MW/h)	6	3	1.5	0	2
Economic cost (k€)	10	8	7	1	6
Time cost	Tr1,Tf1	Tr2,Tf2	Tr3,Tf3	Tr4,Tf4	Tr5,Tf5

2.4.5 Integration into the original MBSA framework

The component-models are shown in Figure 38, the specification ETS of 'FFFR35' ('First failure first repaired for component G3 and G5) and 'PtR2' ('Compared to G3 and G5, G2 has the priority to be first repaired') are designed. With the help of the CCFT (Figure), the failure logic expression of this system is obtained and the operational GRA model is achieved according to the proposal of this research (shown in Figure lower part).

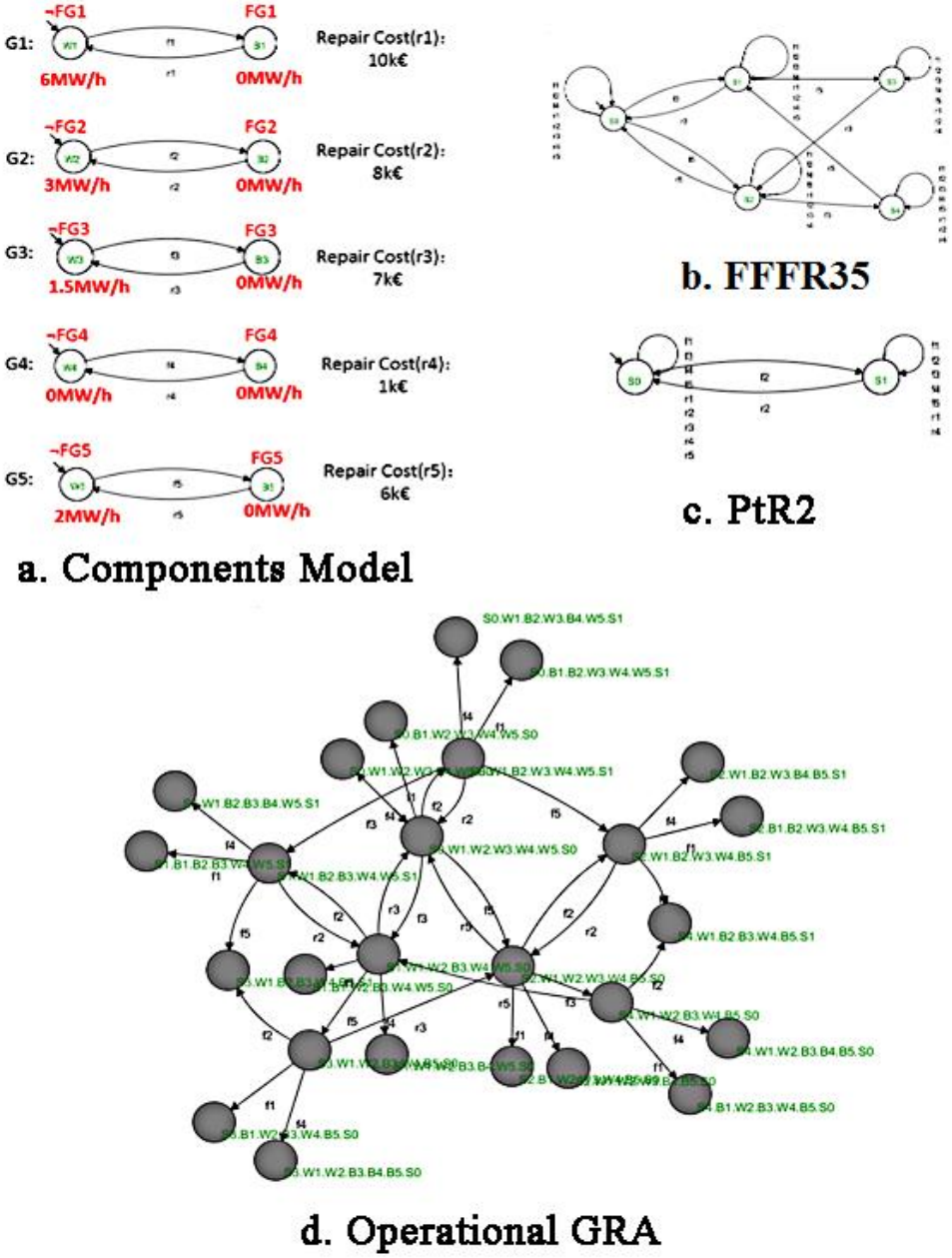


Figure 38 GRA generation with specifications

It is interesting to notice that the operational GRA obtained features a number of states which becomes to be visually difficult to handle. This is natural: the state space obtained through the composition operation has a size equal to the product of the sizes of the original automata. This yields an important limitation on the manual exploitation of such models. Beyond five components, with two specifications, the resulting GRA becomes hard to handle manually. This enforces the utility of simulation techniques such as CCFT in order to gain more insight on the resulting GRA behavior.

The application of CCFT is to calculate the 'system capacity' (shown in Figure 39), an 'and gate' is applied to connect the leaves of G2, G3 and G5, the result of this 'and gate' is the input of a 'or gate',

while this ‘or gate’ is applied to connect the leaves of G1 and G4. The CCFT calculation result is presented in Table 9 CCFT Result.

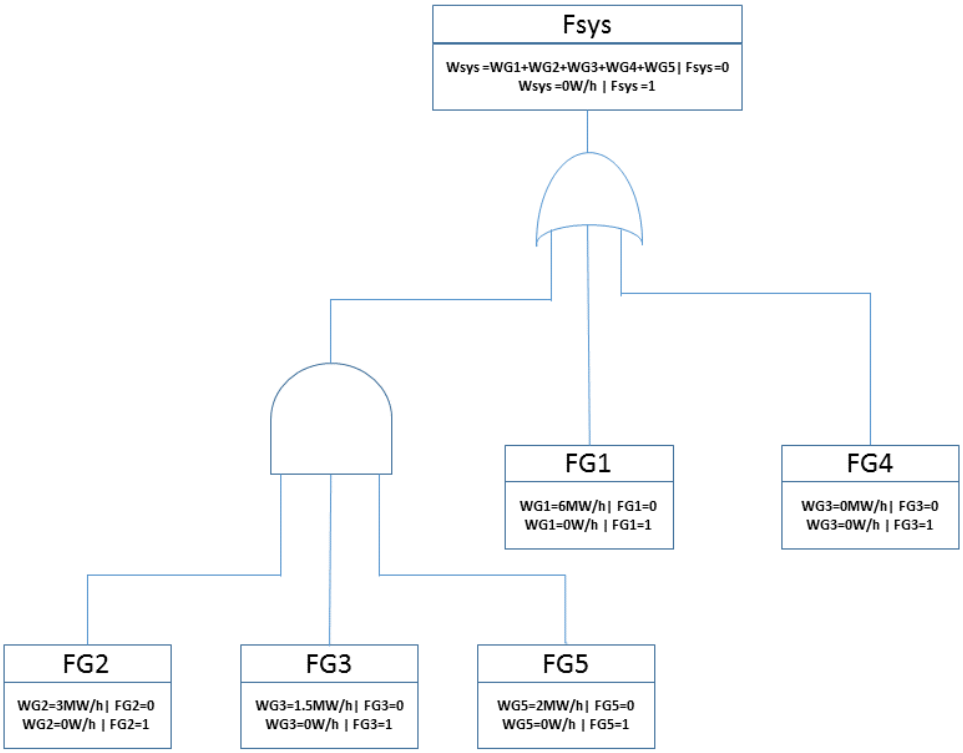


Figure 39 CCFT modeling

Table 9 CCFT Result

FG1	WG1 (MW/h)	FG2	WG2 (MW/h)	FG3	WG3 (MW/h)	FG4	WG4 (MW/h)	FG5	WG5 (MW/h)	Fsys	Wsys (MW/h)
0	6	0	3	0	1.5	0	0	0	2	0	12.5
0	6	0	3	0	1.5	0	0	1	0	0	10.5
0	6	0	3	0	1.5	1	0	0	2	1	0
0	6	0	3	0	1.5	1	0	1	0	1	0
0	6	0	3	1	0	0	0	0	2	0	11
0	6	0	3	1	0	0	0	1	0	0	9
0	6	0	3	1	0	1	0	0	2	1	0
0	6	0	3	1	0	1	0	1	0	1	0
0	6	1	0	0	1.5	0	0	0	2	0	9.5
0	6	1	0	0	1.5	0	0	1	0	0	7.5
0	6	1	0	0	1.5	1	0	0	2	1	0
0	6	1	0	0	1.5	1	0	1	0	1	0
0	6	1	0	1	0	0	0	0	2	0	8
0	6	1	0	1	0	0	0	1	0	1	0
0	6	1	0	1	0	1	0	0	2	1	0
0	6	1	0	1	0	1	0	1	0	1	0

1	0	0	3	0	1.5	0	0	0	2	1	0
1	0	0	3	0	1.5	0	0	1	0	1	0
1	0	0	3	0	1.5	1	0	0	2	1	0
1	0	0	3	0	1.5	1	0	1	0	1	0
1	0	0	3	1	0	0	0	0	2	1	0
1	0	0	3	1	0	0	0	1	0	1	0
1	0	0	3	1	0	1	0	0	2	1	0
1	0	0	3	1	0	1	0	1	0	1	0
1	0	1	0	0	1.5	0	0	0	2	1	0
1	0	1	0	0	1.5	0	0	1	0	1	0
1	0	1	0	0	1.5	1	0	0	2	1	0
1	0	1	0	0	1.5	1	0	1	0	1	0
1	0	1	0	1	0	0	0	0	2	1	0
1	0	1	0	1	0	0	0	1	0	1	0
1	0	1	0	1	0	1	0	0	2	1	0
1	0	1	0	1	0	1	0	1	0	1	0

As each state is already associated with the Boolean Logic (AP) in GRA, the system production capacity of each state is able to be calculated by CCFT according to the AP of each state. For example, in Operational GRA, state (S1.W1.B2.B3.W4.W5.S1) has the Boolean expression: ¬FG1∧FG2∧FG3∧¬FG4∧¬FG5. Using '0' or '1' to represent the 'true' or 'false' Boolean logic of this state is: FG1=0;FG2=1;FG3=1;FG4=0;FG5=0. According to the calculation algorithm, the system production capacity of this state is 8MW/h in the table. Based on the calculation algorithm, a software tool is created for CCFT calculation. In this tool, the input is the state name, and the output is system capacity.

Hence, besides the name and the AP, the model state is enriched with the system production capacity by the application of state-AP and CCFT. According to the transition function of the model, the reachable states of the repair scenario (repair scenario is executed by management requirement) is known and the system capacities in the reachable states are able to be calculated by CCFT. According to the cost and the repair trajectory, 'how much money is paid' and 'how long time it takes' for the management requirement is able to be calculated. So, the effectiveness of management requirement is able to be simulated.

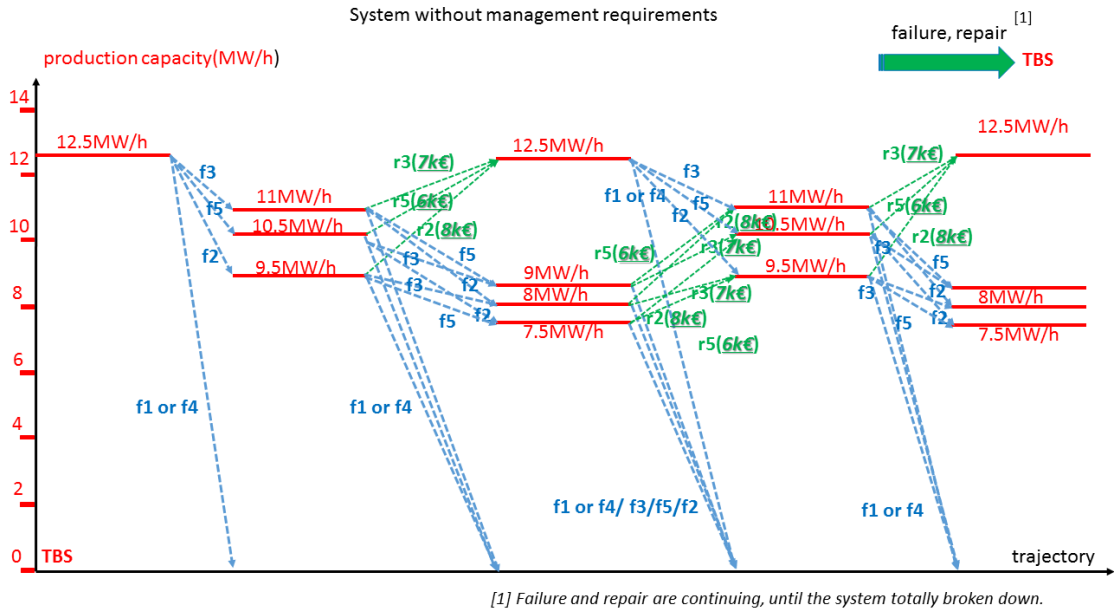


Figure 40 System dynamic capacity simulation (without management requirement)

In Figure 40, the system dynamic production capacity value without management requirement, as well as the failure occurrence and repair cost payment, are calculable. This quantitative analysis is a circle (failure and repair operation are continuous) until the system totally breaks down(reaching TBS in the model).

After the model is specified by FFFR35 and PtR2, all the scenarios executed cases are quantitative analyzed (shown in Figure 41). In it, figure (1) and (2) are the simulation result of FFFR35. From figure (2) to (6) are the simulation result of FtR2. The figure (7) is an example of the comparison simulation with and without PtR2, in case that system is in state (S1.W1.B2.W3.W4.B5.S1) meaning G2 and G5 have broken down. The current system-capacity of state (S1.W1.B2.W3.W4.B5.S1) is 7.5 MW/h. By the PtR2 executed, the G2 is first to be repaired (r2 is prosecuted) and the reachable state is S2.W1.W2.W3.W4.B5.S0 (in this state the system capacity is 10.5MW/h). Because the cost of r2 is 8k€, the PCR1 (Production capacity Recovered) is 10.5-7.5 MW/h=3MW/h by the payment of 8k€. Without PtR2, r5 will be firstly prosecuted and the PCR2 is 2MW/h by the payment of 6k€. Though the cost is cheaper without PtR2 executed, PCR is higher with PtR2 executed (PCR1>PCR2). So, in case the system is required of a lager recovery of PCR, the PtR is needed. The component that is able to contribute a larger production capacity should have the priority to be first repaired. The figure (8) is an example of the comparison simulation with and without FFFR35. With or without FFFR35, the economic cost (G3 and G5 are both repaired) is always to be 13k€. According to Table 8 the period time of f5, f3, r5 and r3 are respectively Tf5, Tf3, Tr5 and Tr3. With FFFR35, the DT (Downtime) of G3 and G5 are: DT_{G3}=Tf5+Tr3 and DT_{G5}=Tr3+Tr5 (Shown in (8) in red), and the Summation of Downtime is SDT1= Tf5+2Tr3+Tr5. Without FFFR35, the DT (Downtime) of G3 and G5 are: DT_{G3}=Tf5+Tr5+Tr3 and DT_{G5}=Tr3+Tr5 (Shown in (8) in black), and the Summation of Downtime is SDT2=Tf5+2Tr3+2Tr5. It

is obviously that SDT2>SDT1. The application of FFFR is helpful to reduce the Total Down Time of the components. So, in case the components are of relatively similar location in the system structure, similar production capacity and similar economic cost, the FFFR is needed.

The ‘economic cost’, ‘time cost’, and the corresponding ‘system production capacity’ are able to be simulated, according to the system management requirements. The quantitative simulation of the effectiveness of the management requirement is a further contribution to the consideration of operational dependability. The system operator is able to choose a suitable management requirement, by the help of this simulation approach.

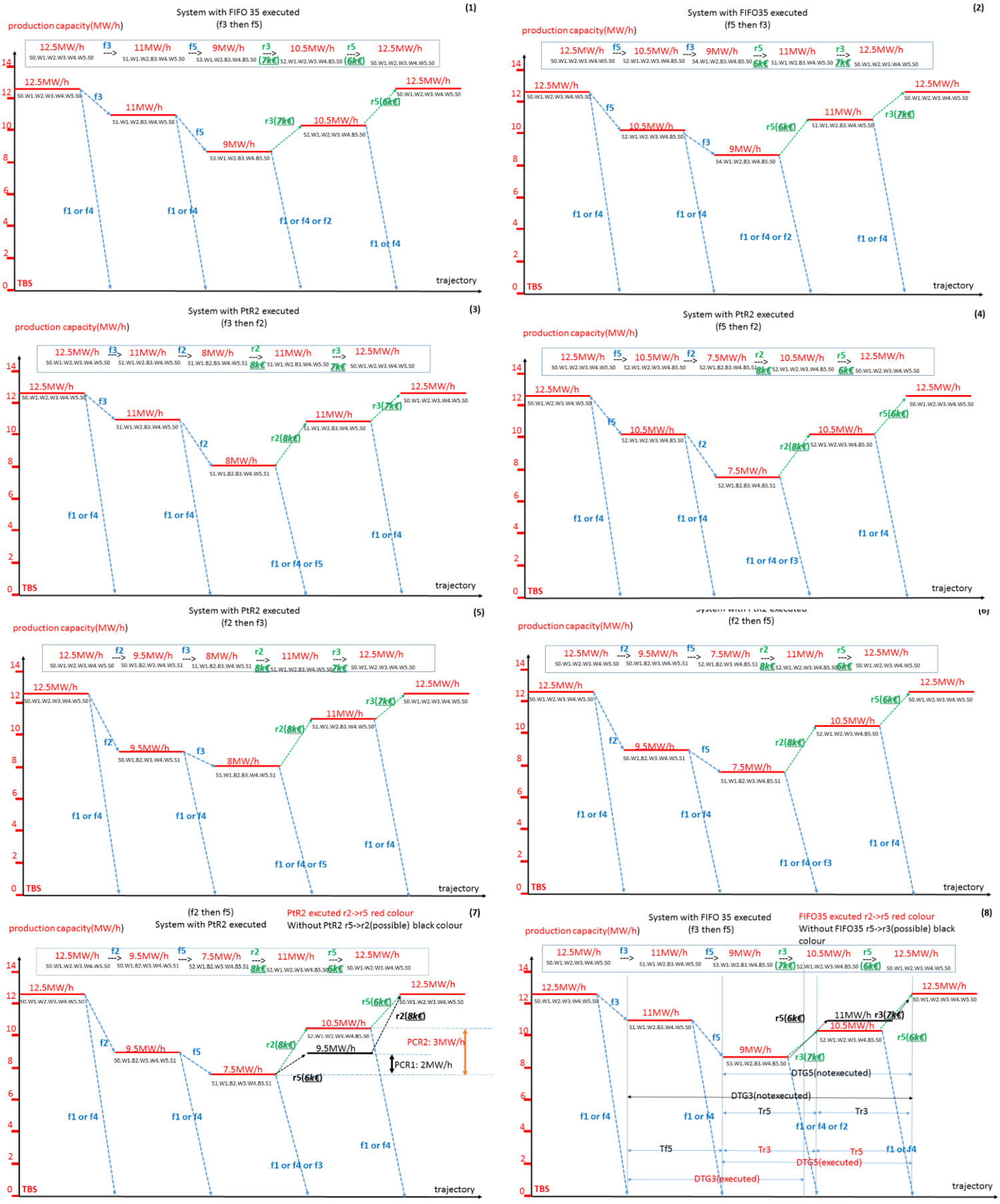


Figure 81 simulation of the system production capacity with management requirements

2.5 Conclusions

Component models expressing various scenarii are able to be expressed and exploited in order to assess the system reliability model. This is successively achieved by composition, expressing the failure logic expression resulted from FtA, by linking it as a set of atomic propositions (AP) to a subset of states where these propositions are asserted and finally by identifying the TBS identified and making it a sink state. These steps have shown their utility in the assessment of operational reliability.

This approach can be extended towards operational maintainability and availability. The various management requirements can be expressed as shown in this chapter. Compared to the normal model generation approaches, a change or addition of the specification always lead a completely resubmitting the model, which is both resource-consuming and error-prone. The advantage of this specification solution is to apply the modular modeling method: the management requirements specifying method has strong compatibility, which means it is suitable for various systems, one kind of management requirement always remains to be the same specification module establishment.

For some cases, as the model has already been specified, another management requirement is needed as a supplement. Such requirements may sometimes be conflicting with each other. Yet, the modeling approach introduced remains totally open to the automatic formal verification in order to prevent such situations. Indeed, the event-based models handled can be naturally fed to formal verification tools. Thus, this research offers a multiple management requirements specifying function by composing the specification automata, where each specification automaton presents one unique management requirement. This approach contributes to the ability that there is no need to redesign a new specification automaton which is able to contain all the information of additional management requirements. It is only needed to continuously compose the new module (specification automaton) with the previous model.

A quantitative simulation approach coherent with both MBSA and our modeling proposal is presented, on order to handle the effectiveness of the management requirement. The effectiveness is proposed by the calculation of ‘money cost’, ‘time cost’ and ‘system capacity’ according to the management. This simulation approach is for judging which management requirement is suitable.

There exist various management requirements according to different situations. This research provides an open specifying issue, the safety analysts are able to design their own specification automaton according to demand.

Chapter 3
Case of Study

This chapter mainly contains two parts. First, the operational reliability model generation of a miniplant system is studied. Second, the effectiveness simulation of the management requirements.

Contents

3.1 System Introduction

A miniplant system (shown in Figure 92) is studied in this chapter. This miniplant is modified from the benchmark of [115] to be more complex as well as more suitable for studying the proposal in this research. This energy delivery system is constructed by three sub-systems by the series collaboration form. Each sub-system have several faulty-repairable components. There are the time-duration for failing and repairing, the money cost for the repair operation, and the production capacity for the electric power, in the components.

Table 10 Data of miniplant

Name / parameter	Sys1	Sys2			Sys3				
	G1	G2	G3	G4	G5	G6	G7	G8	G9
characteristic	f1,r1	f2,r2	f3,r3	f4,r4, start,stop	f5,r5	f6,r6	f7,r7	f8,r8	f9,r9
Capacity (MW/h)	15	10	10	10	5	3	3	2	1
Economic cost(k€)	10	10	10	10	8	5	5	5	5
Time cost	Tf1, Tr1	Tf2, Tr2	Tf3, Tr3	Tf4, Tr4 Tstart, Tstop	Tf5, Tr5	Tf6, Tr6	Tf7, Tr7	Tf8, Tr8	Tf9, Tr9

In Sys1: there is only one components G1, with the production capacity of 15MW/h. G1 is possible to fail and to be repaired, and the money cost for repair operation is 10k€.

In Sys2: G2, G3 and G4 are repairable, by the repair money cost of 10k€. G3 is the cold redundancy of G2. If G2 breaks down, G3 begins to work. G4 needs reposing after it has been working for a certain time duration, so G4 has three states: idle state, working state and broken down state. When component G4 is in the idle state, the component does not break down, but it also does not contribute to the production (the capacity is 0MW/h at this state). G4 has the priority to be first repaired, comparing to all the other components in the whole system.

In Sys3: form G5 to G9, the components are working in the hot redundancy collaboration form. Among these components, the money cost to be repaired for G5 is 8 k€, for the other components are 5 k€. When the capacity of Sys3 is lower than 8MW/h, this sub-system is not able to work functionally, which should be regarded as breakdown. G5 has the priority to be first repaired, comparing to the other components in Sys3 (suitable for: G6, G7, G8, G9; not suitable for: G1, G2, G3, G4). G6 and G7 respects

the repair principle ‘first failed, first repaired’, comparing to the other components in Sys3 (suitable for: G5, G8, G9; not suitable for: G1, G2, G3, G4).

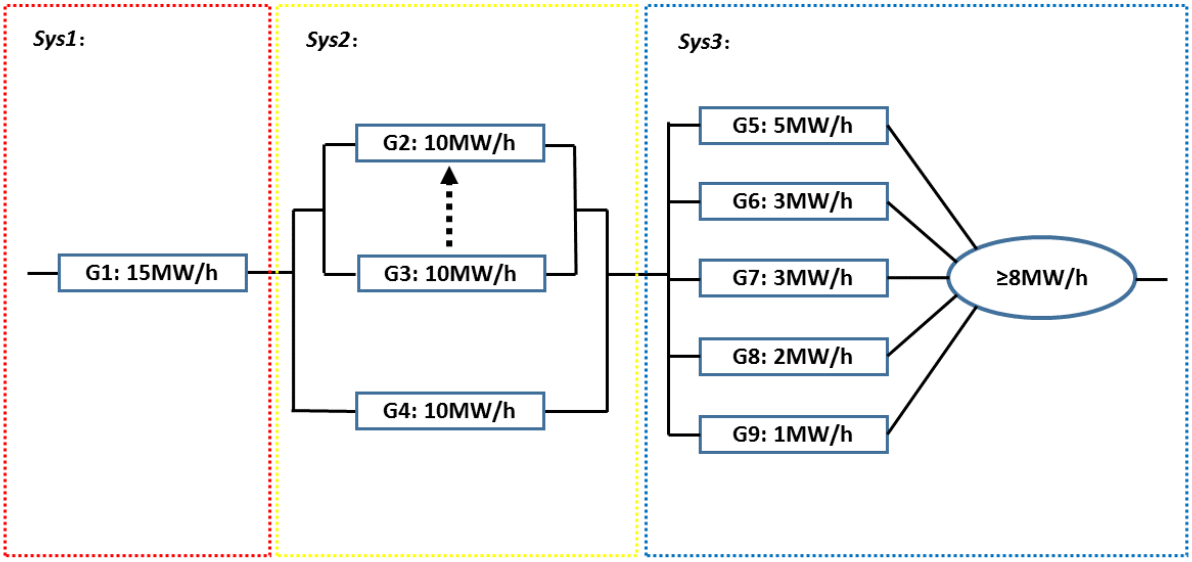


Figure 92 Mini Plant system

3.2 Model Generation

Based on the proposal of this research, mainly four steps are needed for the model generation. They are: ‘components modeling’, ‘CCFT modeling’, ‘specification automata modeling’, and ‘GRA generation’.

3.2.1 Components Modeling

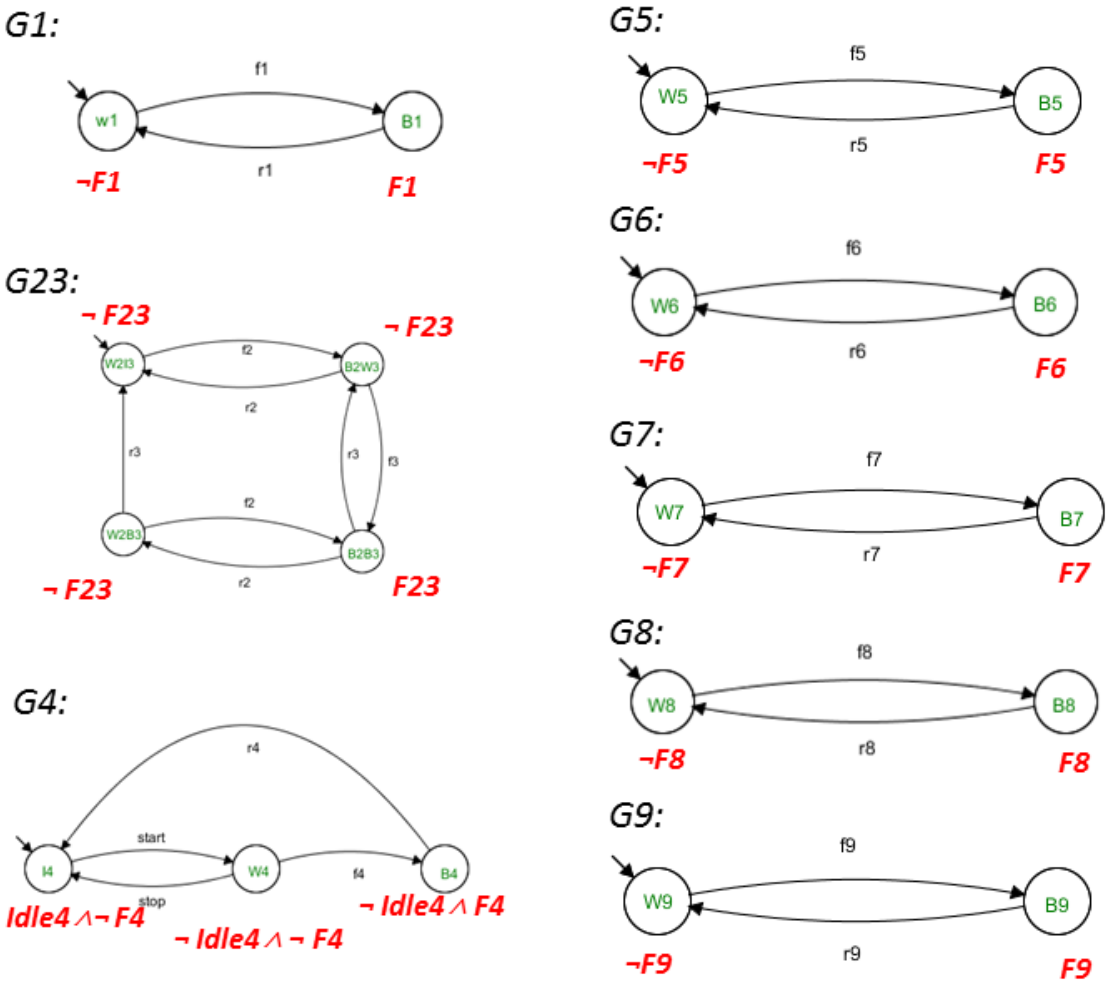


Figure 103 components in miniplant

Depicted in Figure 103, G1, G5, G6, G7, G8, G9 have two states presenting for system is healthily working and system is broken down, and each state is associated with AP. As it has already been introduced ‘G3 is the cold redundancy of G2’. Model G2G3 respectively presents the behaviors of G2 and G3. Only in case that G2 has already broken down, G3 is possible to break down. The AP element ‘F23’ is used for presenting G2 and G3 are both broken down (based on the characteristic of cold redundancy, when G2 and G3 are both broken down, the cold redundancy system is regarded as broken down), and ¬F23 is used for presenting the opposite (cold redundancy system is able to contributes to its function). G4 has three states: I4 is the idle state (initial state), W4 is the working state,

and B4 is the system broken down state. Based on the meaning of these states, two Boolean letters are set: ‘F4’ means G4 is broken down, and ‘idle4’ means G4 is idle. ‘¬ Idle4 ∧ F4’ is used to associate with B4, ‘¬ Idle4 ∧ ¬ F4’ is used to associate with W4, and ‘Idle4 ∧ ¬ F4’ is used to associate I4.

3.2.2 CCFT Modeling

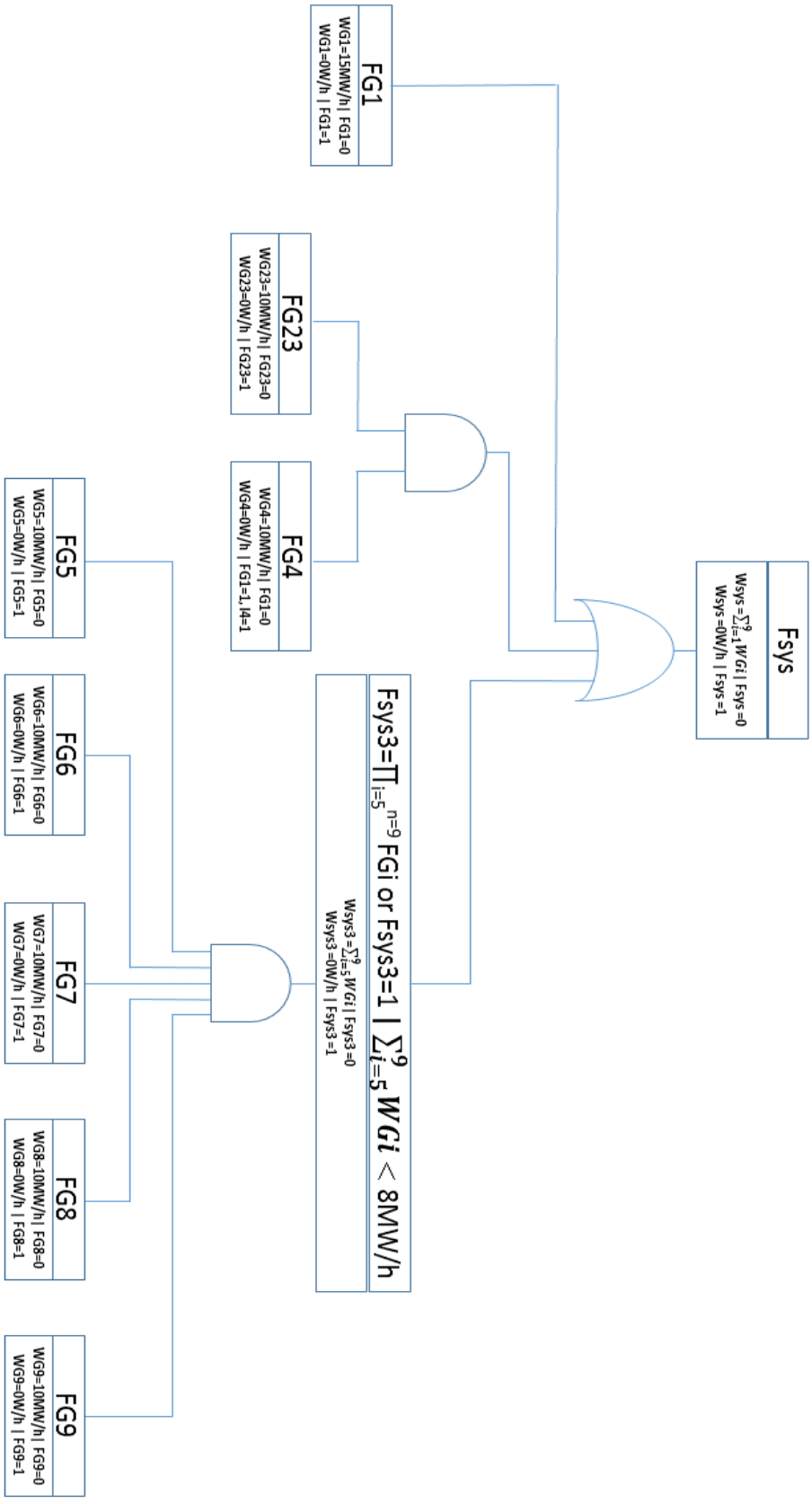


Figure 114 CCFT for miniplant

The CCFT modeling is presented in Figure 1144. The leaves of G1 and G23 are modeled, following the same principle introduced in the proposal part. In the leaf of G4, if the input is I4=1 (meaning component G4 is idle), though the component is not broken down, the production capacity is zero in this condition (WG4=0MW/h). For the leaf of Sys3, the Fsys3 calculation is 'Fsys3=$\prod_{i=5}^{n=9}$ FGi or Fsys3=1 | $\sum_{i=5}^{9} WGi \leq$ 8MW/h'. Here, 'Fsys3=$\prod_{i=5}^{n=9}$ FGi' means Fsys3=FG5∧FG6∧FG7∧FG8∧FG9, which is the result of the 'and' gate. 'Fsys3=1 | $\sum_{i=5}^{9} WGi$< 8MW/h' means: in case the summation of the components-capacity is smaller than 8MW/h, Sys3 is regarded as broken down, which is already stated in the system introduction part.

3.2.3 Specification Automata Modeling

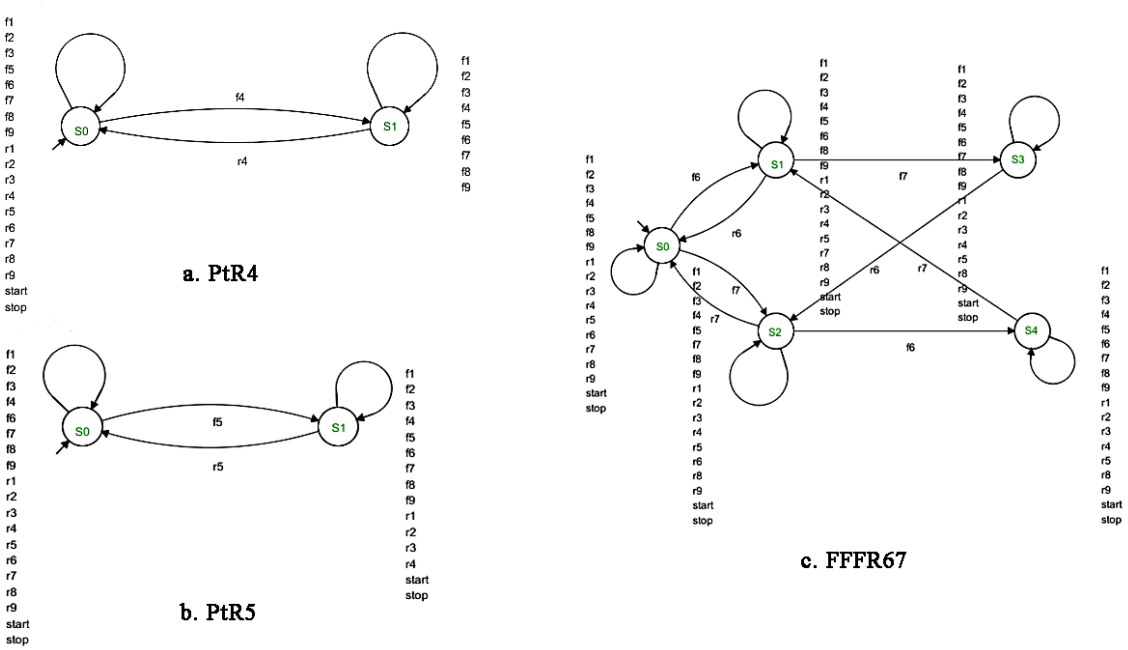


Figure 125 Specification Automata based on the management requirements

Shown in Figure 125, the specification automaton for the management requirement 'G4 has the priority to be firstly repaired, comparing to all the other components in the system' is designed as PtR4. It has already been stated in the system introduction part 'G5 has the priority to be first repaired, comparing to the other components in Sys3 (suitable for: G6, G7, G8, G9; not suitable for: G1, G2, G3, G4)'. In PtR5, the self-loop of state S1 includes 'r1,r2,r3,r4'. This designing ensures that G5 has the priority to be first repaired, comparing to the G6, G7, G8, and G9 (G1, G2, G3, and G4 are not included). It is the same reason for the self-loop of state S3 and S4 in FFFR67, as it has already been stated 'G6 and G7 respects the repair principle first failed first repaired, comparing to the other components in Sys3 (suitable for: G5, G8, G9; not suitable for: G1, G2, G3, G4) '.

3.2.4 GRA Generation

The GFA model is obtained by: GFA= G1//G2//G3//G4//G5//G6//G7//G8//G9, and the specified GFA is obtained by: SPGFA=GFA//PtR4//PtR5//FFFR67.

Based on the CCFT in Figure 114, the failure logic expression is obtained:

$$\varphi = FG1 \vee (FG23 \wedge FG4) \vee Fsys3$$

which is the identification property for TBS. Fsys3 is divided into two parts, the first part is Fsys3'= FG5∧FG6∧FG7∧FG8∧FG9, the second part is Fsys3''=1| $\sum_{i=5}^{9} \boldsymbol{WGi}$<8WM/h. Thus, Fsys3=Fsys3' ∨Fsys3''. So, the TBS identification can be achieved by two steps. The first step is the TBS identification method which is introduced in the Reliability Model Generation(in equation(2-5)): TBS'=IS(φ'), and φ'= FG1∨(FG23∧FG4)∨(FG5∧FG6∧FG7∧FG8∧FG9). The second step is to identify the states, whose AP is not the sufficient condition of φ', but they lead a low capacity of Sys3 (less than 8MW/h). The identification property is φ''= {Fsys3=1| $\sum_{i=5}^{9} \boldsymbol{WGi}$<8WM/h}. As it is already introduced in the algorithm of CCFT, the input is the state name and the output is the system capacity. By this software, the system state names which lead the sub-system Sys3 to a low capacity are known. The summation of the states identified by φ' and the states identified by φ'' is the TBS set of the System.

After the TBS are automatically identified, the TBS are made absorbing in order to to generate the GRA model.

It is important to notice that the resulting model has a size which becomes prohibitive for a visual comprehension. Indeed, the by taking into account the number of states of each component, or specification model, we obtain a total number of 3840 states. Understanding the modeled behavior by a visual approach becomes extremely difficult, and at any rate error prone. This is why simulation appears to be very useful in order to help engineers assess the behavior of the GRA model. CCFT simulation is applied in the sequel in order to gain this insight.

3.3 Simulation

This simulation is mainly for testing the ability of GRA model for failure scenario expression and for the consideration of operational dependability. In 3.3.1, the failure scenario expression is simulated by the criterions of N-state behaviors and event sequences of components G2, G3 and G4, by using Supremica. In 3.3.2, the operational dependability modeling is simulated by the management requirements (PtR4, PtR5, and FFFR67), by using Supremica. In 3.3.3, the effectiveness of the management requirements is simulated to proof that these management requirement set onto the system are reasonable, by using CCFT software.

3.3.1 Simulation: Consideration of Failure Scenario Expression

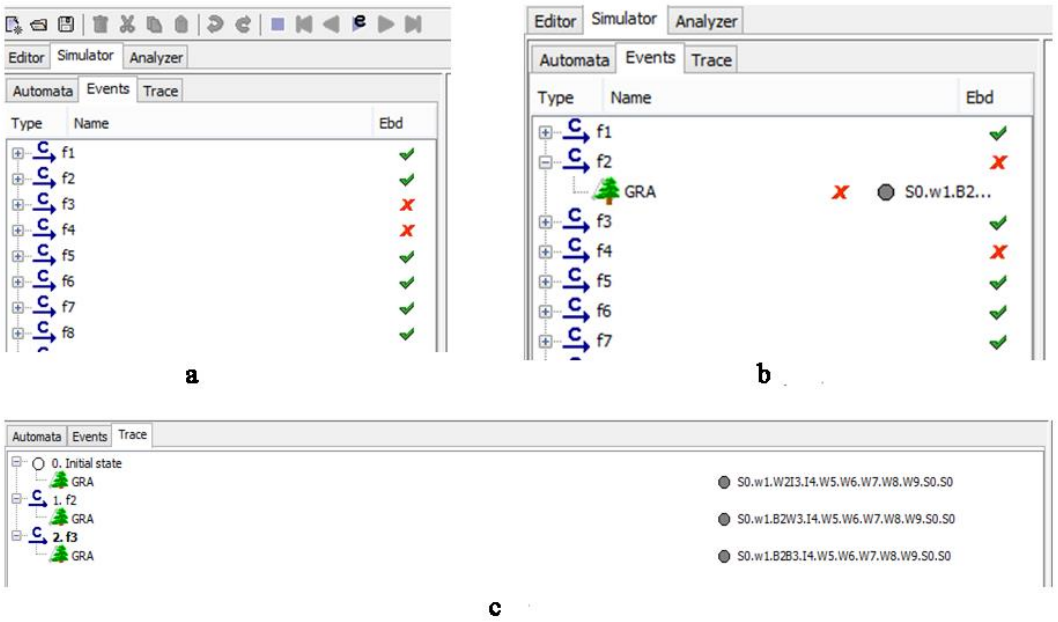


Figure 136 Simulation for G2G3

Figure 136 shows the simulation result of G2 and G3. It is already introduced G3 is the cold redundancy of G2. Depicted in (a), f2 is possible to happen, but f3 is not possible. Depicted in (b), after f2 has already happened, f3 is possible to happen. (c) shows the failure sequence and the reaching states of the cold redundancy design. (a), (b) and (c) are able to detailed present the behaviors of G2 and G3 in the system-model. The simulation result of the failure sequence proofs that the GRA is able to consider the event sequences of the system, according to causality of the service.

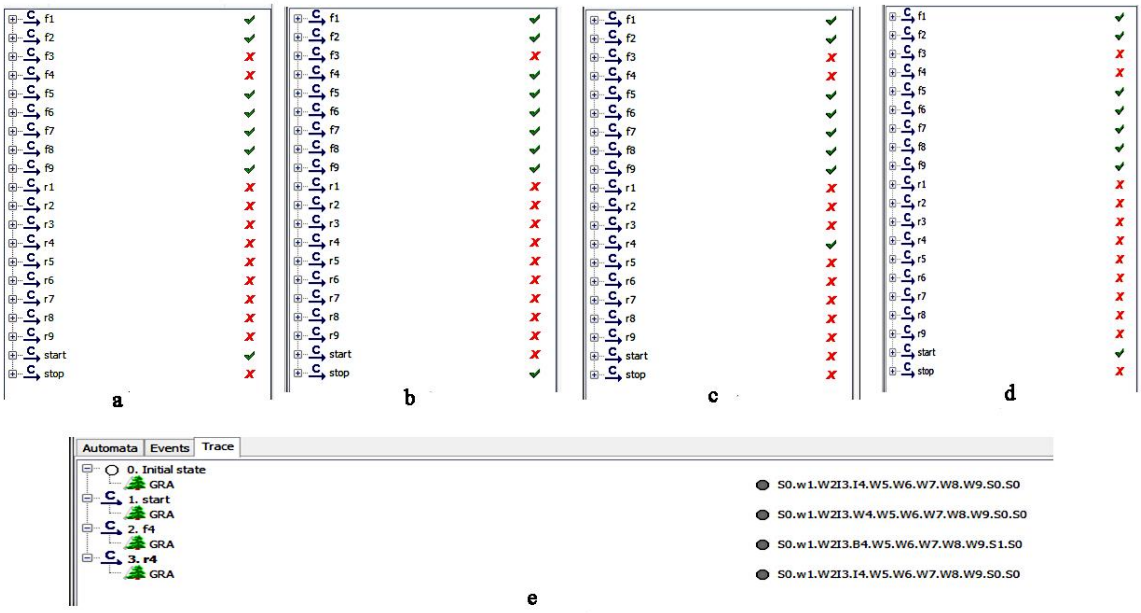


Figure 47 Simulation for G4

Figure 47 shows the simulation result of G4. Subfigure (a) shows that event 'start' is may happen. From (b) to (d) is the behaviors of G4: start (begin to work) -> f4, (failure happens) ->r4(able to be repaired). (e) shows the failure sequence and the reaching states. The behaviors of G4, not only the failure and repair behaviors, but also the start and stop behaviors, are considered in the system model. This simulation result proofs that the GRA is able to consider the N-state behaviors of the system, according to the causality of the service.

3.3.2 Simulation: Consideration of Operational Dependability Modeling

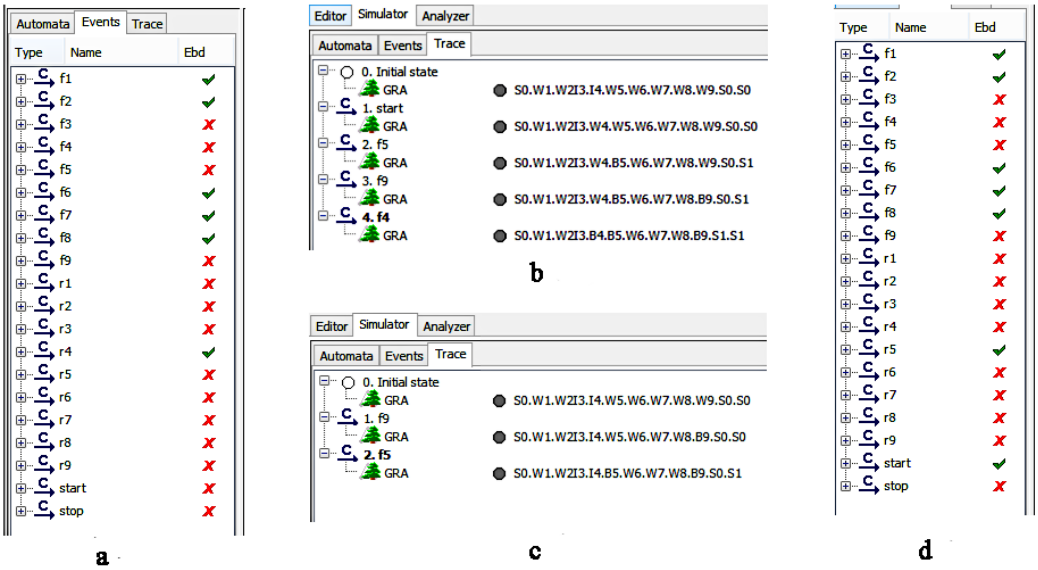


Figure 48 Simulation for PtR4(1)(2) and PtR5(3)(4)

Figure 48 shows the simulation of 'priority to be first repaired' management requirements. In (2), f5, f9 and f4 have already happened. (a) is the possible happening events of (b), and (a) shows the only allowed repair is r4, which means G4 has the priority to be first repaired in the whole system and PtR4 is effective. (d) is the possible happening events of (c), and (d) shows the only allowed repair is r5, which means G5 has the priority to be first repaired in Sys3 and PtR5 is effective.

Figure 49 Simulation for FFFR67

Figure 49 shows the simulation of 'First Failed First Repaired' management requirement for G6 and G7. (1) shows the failure trajectory of 'first f6 and then f7'. (2) is the possible happening events of (1), and (2) shows that G6 is allowed to be repaired (r6 is first to be executed). (3) shows the failure trajectory that first f7 and then f6. (4) is the possible happening events of (3), and (4) shows that G7 is allowed to be repaired (r7 is allowed). Based on this simulation result, it can be known that the management requirement FFFR67 is successfully executed in the system-model.

As the management requirements are set for the consideration of the operational dependability, especially in the domain of the managed repair operations, and the repair sequence (specified by specification automata) is proofed to be realized in the model. So, this simulation result verifies that the system-model generated by this research is able to consider the operational dependability.

3.3.3 Simulation: Effectiveness of Management Requirements

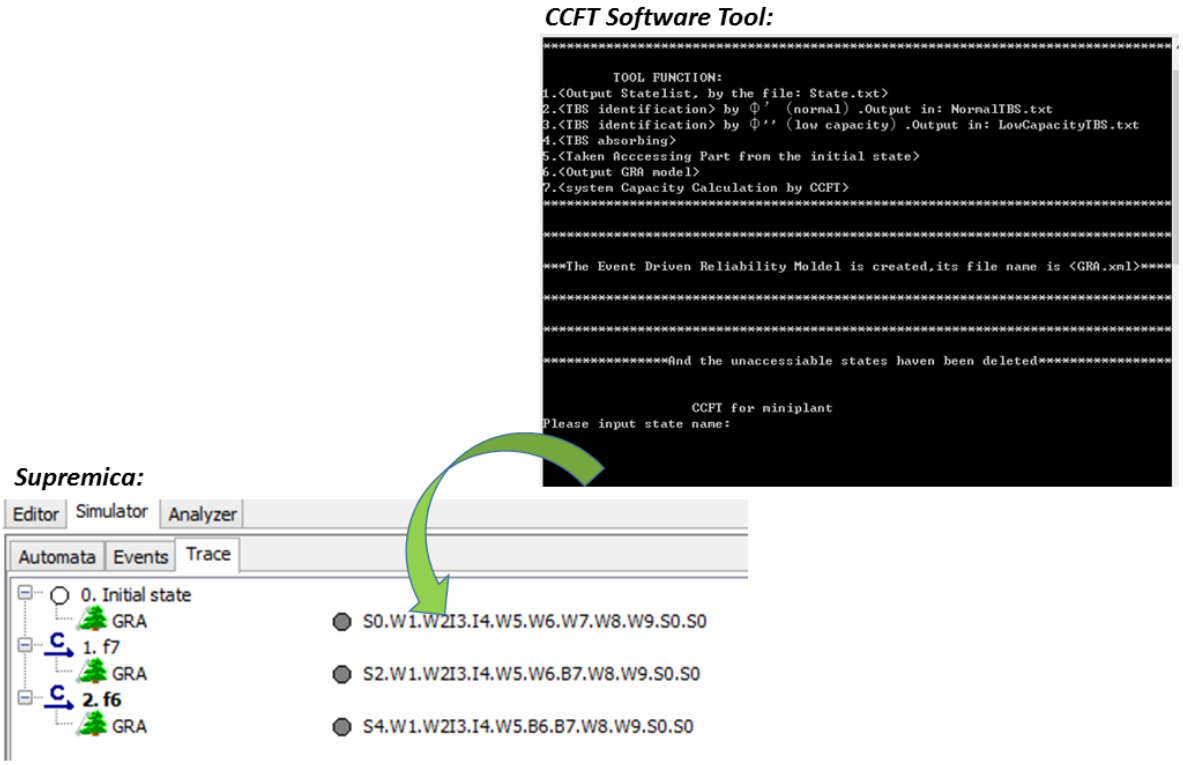


Figure 50 Simulation method of Management requirement effectiveness

Two software are applied for simulating the effectiveness of the management requirements based on the proposal. Figure 50 depicts the simulating solution, Supremica is able to simulate the repair trajectory (specified by the management requirements) by considering the 'event sequence' and the 'reachable states', based on the transition function of the system-model. The CCFT Software Tool is able to calculate the system capacity. As each state is already associated with AP, and the calculation of CCFT is issued by the AP, the state names can be the direct input of the CCFT Software Tool. Thus, according to the reachable states simulated by Supremica, the dynamic system-capacity caused by the repair trajectory is able to be simulated by CCFT. Moreover, the time duration and the money of the events are already given in the introduction part. Thus, the effectiveness of the management requirements can be simulated, by the consideration of 'money-cost', 'time-cost' and 'system capacity'.

The effectiveness of the management requirements for PtR4, PtR5 and FFFR67 is studied in the following part.

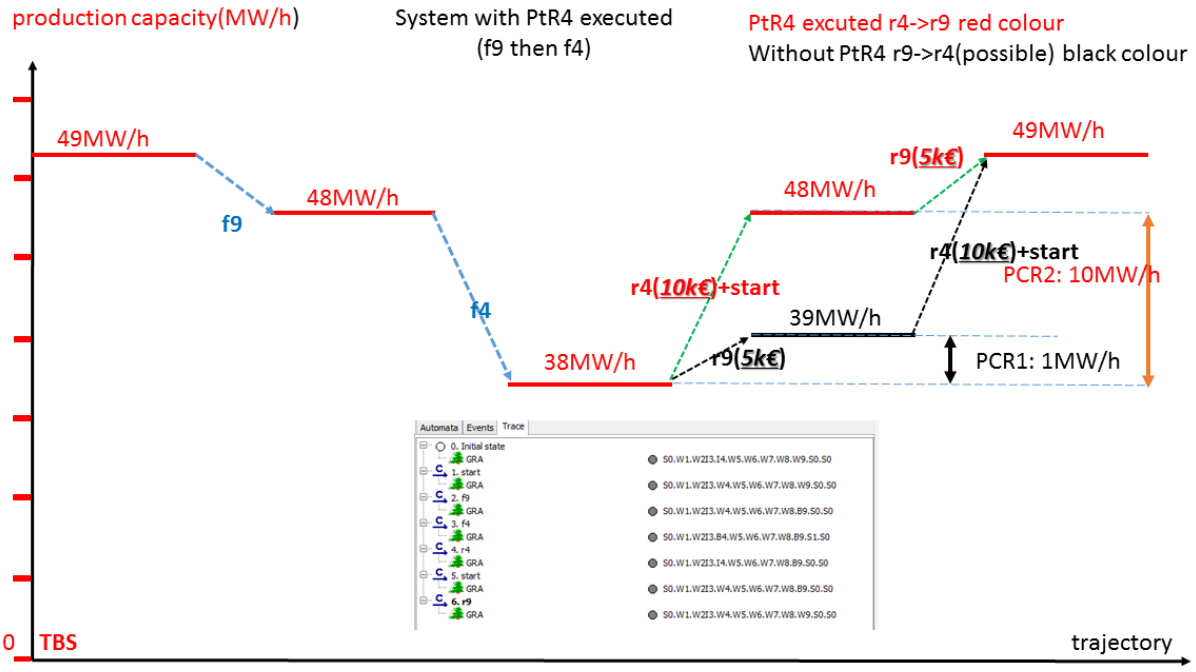


Figure 141 Simulation for PtR4

Figure 141 shows the effectiveness of PtR4 executed (in red colour) and PtR4 unexecuted (in black colour). After component G4 is started, the system capacity is 49MW/h. According to the failure trajectory (shown in the lower, by Supremica) 'firstly G9 failed and then G4 failed', according to CCFT the capacity decreases from 49MW/h to 48MW/h and then to 38MW/h. If PtR4 is executed, the repair trajectory should be 'first r4 and then r9', the capacity recovers to 48MW/h and then to 49MW/h. By the money cost of 10k€, G4 is first repaired, the PCR2 is 10MW/h. If PtR4 is not executed, the repair trajectory is possible to be r9 and then r4, the capacity recovers to 39MW/h and then to 49MW/h. By the money cost of 5k€, G9 is first repaired, the PCR1 is 1MW/h. Though more expensive, the system is able to get a more effective recovery for the production capacity (PCR2>PCR1). Thus, the original intention of management requirement PtR4 (high-production component should be first repaired, in order to get a fast recovery of the system production) is achieved.

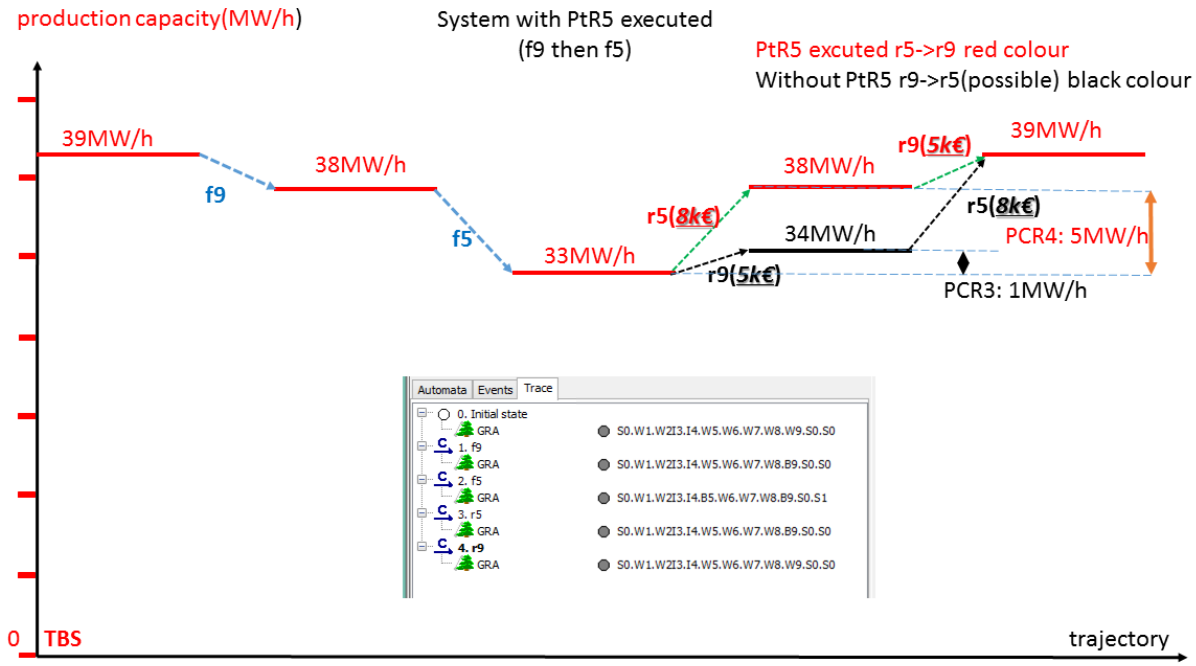


Figure 152 simulation for PtR5

Figure 152 shows the effectiveness of PtR5 executed (in red colour) and PtR5 unexecuted (in black colour). From the initial state, the system capacity is 39MW/h. According to the failure trajectory (shown in the lower, by Supremica) 'first G9 failed and then G5 failed', the capacity decreases from 39MW/h to 38MW/h and then to 33MW/h. If PtR5 is executed, the repair trajectory is r5 and then r9, the capacity recovers to 38MW/h and then to 39MW/h. By the money cost of 8k€, G5 is first repaired, the PCR4 is 5MW/h. If PtR5 is not executed, the repair trajectory is possible to be r9 and then r5, the capacity recovers to 34MW/h and then to 39MW/h. By the money cost of 5k€, G9 is first repaired, the PCR3 is 1MW/h. Though more expensive, the system is able to get a more effective recovery for the production capacity (PCR2>PCR1). Thus, the original intention of the management requirement PtR5 (high-production component should be first repaired, in order to get a fast recovery of the system production) is also achieved.

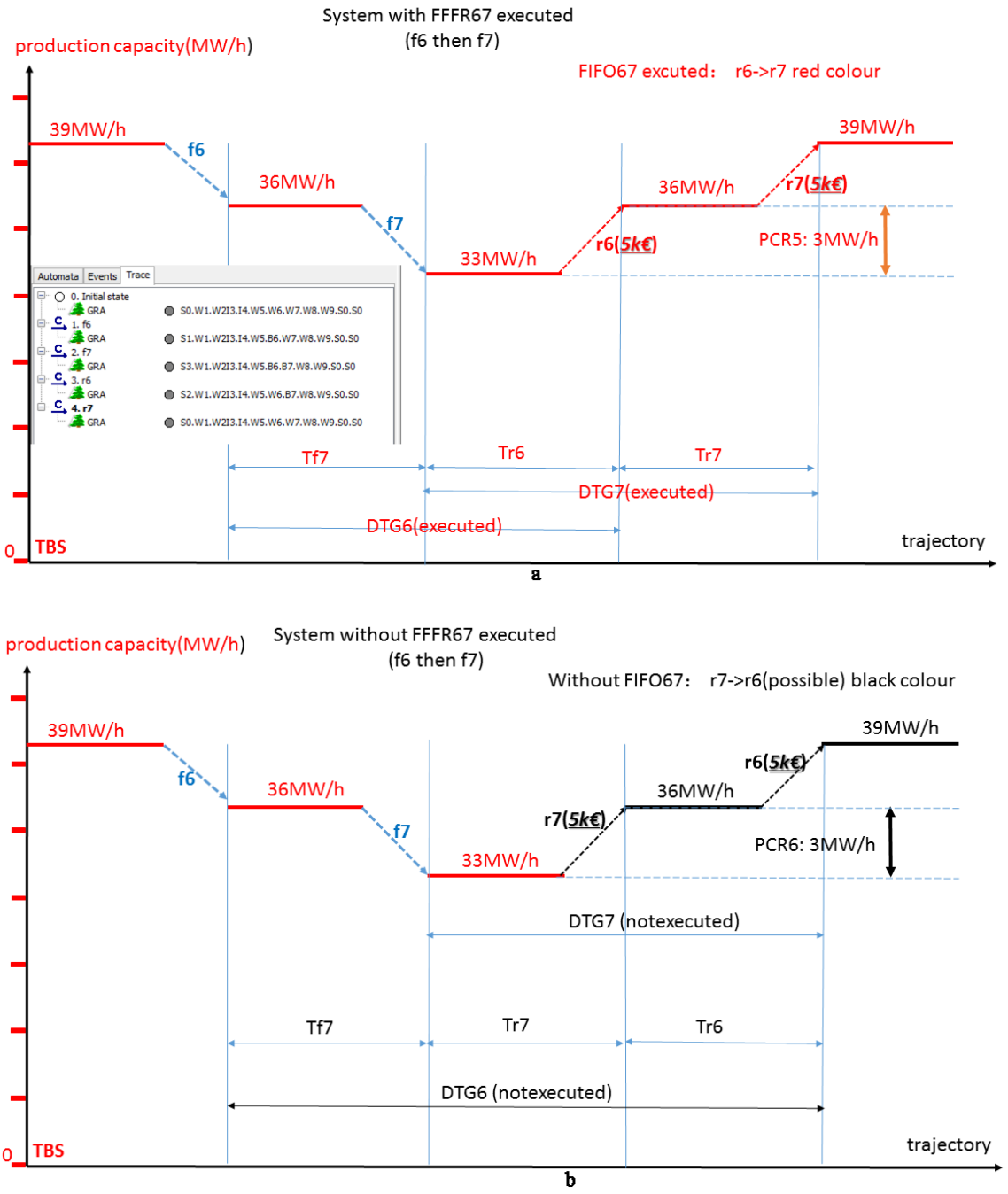


Figure 163 Simulation for FFFR67

The simulation for FFFR67 is shown in Figure 163, the upper side is the effectiveness of FFFR67, and the lower side is the effectiveness without FFFR67. Assume G6 fails and then G7 fails, the system capacity decreases from 39MW/h to 36MW/h and then to 33MW/h. In case that FFFR67 is executed, G6 is first to be repaired and then G7 is repaired. The system production capacity recovers to 36MW/h and then 39MW/h. The downtime of G6 is DTG6=Tf7+Tr6 and the downtime of G7 is DTG7=Tr6+Tr7. Here, 'Tf' means the time duration of failure, and 'Tr' means the time duration of repair. Therefore, the summation of downtime in the system is DTsys=DTG6+DTG7=Tf7+2Tr6+Tr7. In case FFFR67 is not executed in the system (shown in the lower part of the figure), it is possible that

G7 is first repaired and G6 is repaired. The downtime of G6 is DTG6=Tf7+Tr7=Tr6, and DTG7=Tr7+Tr6. Thus, the summation of the downtime in the system is DT_{sys}=DTG6+DTG7=Tf7+2Tr6+2Tr7. It is obvious that the summation of downtime in the system is smaller by applying FFFR67, while the capacity recovered is the same (PCR5=PCR6) and the money payment is the same (money cost is: 5k€+5k€). Therefore, the operational requirement FFFR67 enforced on the system by composition is effective to save the downtime of the system.

3.3.4 Summary

In this chapter, the approach of operational reliability model generation is illustrated by a miniplant benchmark, following the steps of 'composition of the component behaviors to generate the system model', 'management requirements expressing by the specification automata' and 'structure analysis and capacity calculation by CCFT'. The simulation of 'system failure scenario expression' verifies the reliability model is able to consider the system's failure scenario, as the function of considering 'dependability' it should own. The simulation of 'operational dependability modeling' and 'effectiveness of management requirements' verifies the management requirements set onto the miniplant are fully considered by quality (by the event trajectory in the model) and by quantity (by the dynamic capacity caused by the operations), as the function of considering 'operational' it should own.

The resulted operational models in this case of study are of relatively big scales, while GFA has 960 states and GRA has 625 states, so that the exploitation of the evaluating dependability is only possible by the assessment tools, such as the already demonstrated tool GRIF. The association of operational specifications and CCFT is of the advantage of 'operational dependability' in the aspects of both quality and quantity, which highlights the simulating functions by the event trajectory in model associated with the capacity, time cost and economic cost, benefiting the dependability analysts， system designers and system operators.

General conclusions and perspectives

Conclusions

There are many tools for modelling and evaluating safety indicators, each with known advantages and disadvantages. The particularity of the MBSA approach consists in formulating techniques close to those used in engineering. By considering requirements (operating constraints after failure) expressed in the same formalism as the dysfunctional models, this proposal reinforces the MBSA framework. Compared to classical model generation approaches, a change or addition of specification always leads to a new formulation of the model, which is both resource consuming and error prone.

In some cases, as the template has already been specified, adding an additional requirement involves a new template provided that the multiple requirements do not conflict. The resolution of conflicts is made simple in the modular modeling approach proposed here, it consists in using the formal tools issued from the theory of automata. The operational reliability model generation approach used in this thesis meets these principles.

The components are assumed to exhibit stochastic independence in failure / repair occurrences, the switch in the voting structure is ideal, and the dysfunctional behaviors among the components are asynchronous (no common mode). The theory of discrete event systems is exploited here both for the generation of models and the establishment of proofs (equivalence between EDA and CTMC). The automaton formulation validates the accessibility to specific states such as TBS, state or set of critical states. Generation comes from the composition of models (dysfunctional behavior, requirements). This simple composition operation overcomes the horizontal complexity (modular formulation). Thus in the generated model, faulty behaviors, complex configurations and operational requirements are taken into account.

To fully express the failure or recovery scenario, various reasoning starting from failure occurrence and in inactivity or maintenance situations can be taken into account. The overall faulty model of the system is generated by composing the component models, in which the failure scenario of the whole system is explicit.

On the basis of minimum cuts from fault trees, total failure states are identified and located in the dysfunctional model. These states must become absorbing states in the global model in order to define a reliability model. Finally, the proposed computational fault tree model (CCFT) evaluates the efficiency of requirements by simulation. Quantitative issues become useful when the generated model size is not more readable.

The highlights of this work lie in a collaborative approach between engineering and operational safety through the MBSA concept (model based safety assessment). The formal aspects on which the proposal is based (composition of regular automata) are consistent with the rigor of the design on the one hand and the resulting operations on the other hand, particularly in terms of validation properties. As such, this work is as much about methodology as it is modeling.

This work is exploratory in nature, and therefore is limited: no direct interface with tools for calculating reliability indicators (for example GRIF or AltaRica). The global model is derived from models of local components through the composition increasing complexity. This presents a potential risk of explosion of states. Structures can only be analyzed by minimal cuts if the assumptions based on RBD or FtA are correct. The dynamic structures of certain fault trees or the mixed structures (continuous / discrete) are excluded from the study. The CCFT can be used on the sequence of the scenarios.

Perspectives

The work presented opens interesting perspectives:

Enrich conventional states to N-state Automata models for functional and dysfunctional representation: for example addition of the states "inactive", "degraded operation" and "partial failure", "total failure", "maintenance", ... because the more accurate the system is, the more states are needed to represent the failure or recovery scenario in the model.

The generation of the operational reliability model is introduced in this thesis as a guideline. Operational availability and operational maintainability models can also be generated using an identical approach. The initial assumptions are changed and the specifications are particular to the intended objective.

The formalism presented is based on automata models. Still, Petri nets can be of interest in dysfunctional simulation because they are more concise. On the other hand, the composition remains an open problem. However, subsequent to these models, the expressiveness of the formal properties of discrete event systems (liveliness, reachability) remains highly interesting and supports the development stages.

The quantitative aspects remain totally prospective. Beyond the operating limits of fault trees for capacity calculation (CCFT), dynamic gates or event triggers to describe more complex failure scenarii would be of a definite contribution.

Résumé

Génération de modèles de fiabilité opérative

Changyi XU, laboratoire Ampère UMR 5005

Ce résumé se compose de quatre parties principales, I la définition du contexte de la recherche, II les motivations et positionnement, III le cadre propositionnel et ses originalités pour se terminer sur IV un bilan et des perspectives.

Les faits marquants de ce travail se situent dans une approche collaborative de l'ingénierie et de la sûreté de fonctionnement au travers du concept MBSA (model based safety assessment). Les aspects formels sur lesquels reposent la proposition (composition d'automates réguliers) sont cohérents à la rigueur de conception d'une part et aux exploitations qui y sont conséquentes d'autre part, notamment en termes de propriétés de validation. A ce titre, ce travail relève autant de la méthodologie que de la modélisation.

Ⅰ. Contexte de la recherche

Les systèmes industriels sont conçus et pilotés pour fournir un service global délivré à partir de plusieurs composants en interaction et éventuellement distribués. Les interactions entre ces composants reposent sur des échanges d'information au travers d'interdépendances matérielles. Chaque composant est sujet à des défaillances conduisant à des états de panne et certains d'entre eux, moyennant des moyens appropriés peuvent être réparés. L'impact d'une défaillance d'un composant sur la prestation du service global attendu par le système peut varier en fonction des interdépendances relatives à ce composant. Ainsi, l'aboutissement d'une panne de composant n'est pas toujours synonyme d'une panne totale du système et ainsi de perte définitive du service global.

La garantie de continuité de service de système est généralement critique dans tout contexte industriel. Afin de fournir une telle garantie, les défaillances des composants individuels et leurs impacts possibles sur la prestation de services doivent être évalués et des opérations appropriées de circonsision (de propagation de défaillance) doivent être anticipées. Ainsi, l'ingénierie de conception et de pilotage d'un système en vue d'assurer une délivrance de service global malgré l'occurrence de défaillance doit être complétée par un corpus d'action lié à l'analyse de la sûreté de fonctionnement. Les travaux abordés ici dans ce contexte concernent notamment l'assistance à la modélisation et l'évaluation de la fiabilité, la disponibilité et la maintenabilité d'un système structuré par des composants soumis à des défaillances.

Dans un cadre d'ingénierie de la conception, outre les connaissances des processus de défaillance, l'intégration des spécifications de reprise de fonctionnement (pilotage) après panne partielle constituent l'une des originalités de ces travaux.

Dans un tel contexte, le flux de conception de système regroupe à la fois les points de vue fonctionnels et dysfonctionnels et l'évaluation des impacts de défaillance nécessite plusieurs étapes de modélisation :

1. une modélisation dysfonctionnelle locale : l'étape de modélisation des composants sujets aux défaillances fournit des connaissances qualitatives et quantitatives sur les comportements défectueux et leur probabilité d'occurrence ;

2. une modélisation structurelle du système : l'étape de conception du système définit la structuration des composants, support de propagation de défaillance, au travers de leurs interdépendances. Le modèle structurel qui en résulte met en évidence l'impact comportemental de la défaillance d'un composant sur le service global attendu ;

3. une approche d'évaluation : l'étape d'évaluation de la fiabilité basée sur un modèle repose sur les 2 modèles précédents. L'évaluation peut être qualitative ou quantitative et s'appuie sur des outils formels dédiés pour définir l'assurance de la continuité du service global. Cette dernière étape fournit à la fois des résultats quantitatifs (temps moyen de défaillance, de réparation, probabilités d'atteignabilité des états de panne, etc.) et qualitatifs (scénarios de propagation de défaillance et impact du choix des spécifications de reprise sur le service attendu).

La manipulation de ces différents types de modèle s'inscrit dans une démarche systémique classique de la sûreté de fonctionnement encore appelée MBSA (Model Based Safety Analysis).

A. Modélisation dysfonctionnelle

Dans le contexte d'éventuelles occurrences de défaillance sur les composants, les comportements spécifiques liés à l'impact de leur propagation sur la qualité de service global doivent être exprimés en tant que tel. Ainsi, selon les précepts classiques de la sûreté de fonctionnement, la fiabilité est la capacité d'un système à fournir en permanence un service attendu sans atteindre un état de panne total, la disponibilité est la capacité du système à fournir un service malgré des états de panne totale et enfin, la maintenabilité désigne la capacité d'un système à être entretenu pour recouvrir un service. Derrière ces expressions intuitives et qualitatives, il existe un corpus mathématique sur lequel se fondent les principaux outils relevant du MBSA.

Sous certaines hypothèses opérationnelles (unicité, indépendance), il sera considéré qu'un composant soumis à des défaillances peut être réparé. Les défaillances se produisent avec un taux moyen noté 'λ', et les réparations s'opèrent avec un taux moyen noté 'μ'. De manière rudimentaire, un composant peut évoluer entre deux états stables : un état de fonctionnement et un état de panne.

Dans un modèle d'état de fiabilité, un composant est supposé être initialement fonctionnel. L'état initial est l'état de fonctionnement. Il peut subir l'occurrence d'une défaillance avec un taux 'λ' et suivre la transition vers l'état de panne. Dans le modèle d'état de maintenabilité, le composant est supposé être initialement en état de panne et peut être réparé avec un taux moyen de 'μ', revenant à l'état de fonctionnement. Dans le modèle de disponibilité, le système est supposé être initialement fonctionnel, subir une défaillance et être réparé de manière récurrente.

Ces trois modèles d'état sont représentatifs pour le calcul des indicateurs de temps moyen de défaillance / réparation les plus classiques. De telles représentations de base se concentrent essentiellement sur l'aspect dysfonctionnel, néanmoins, les modèles peuvent être enrichis par la considération d'états lié au service (état de veille non productif, état de mise en sécurité, état de mise en réparation partielle, etc…). Ainsi le cadre initial conforme à la normalité d'exploitation du ou des composants peut être précisé. Ceci est illustré en modélisation par des événements de lancement ou d'arrêt de production {'α', 'β'}, de panne / réparation {'λ', 'μ'}, d'état veille ou 'idle', 'fonctionne' et 'a échoué'.

La fiabilité, la maintenabilité et la disponibilité peuvent facilement être évaluées sur ces modèles enrichis en définissant de manière adéquate l'état initial comme étant un état inactif ou fonctionnel.

B. Modélisation structurelle

La gestion de la complexité d'un système industriel destiné à fournir des services nécessite des compétences et des ressources d'ingénierie avancées, à plusieurs niveaux :

1. conception : définition de l'architecture fonctionnelle, logique et physique par l'analyse des besoins, l'analyse fonctionnelle et la décomposition, la cartographie des composants logique et physique ;

2. développement : pour chaque composant définir et valider son comportement, puis en inclusion progressive avec le reste du système ;

3. déploiement : mise en place progressive des interactions ;

4. fonctionnement et gestion du cycle de vie : cela englobe la maintenance prédictive et curative.

Une phase d'analyse dysfonctionnelle menée en parallèle avec la phase de développement du système, induit souvent des itérations coûteuses afin de reconcevoir les composants, du fait de l'identification tardive des aspects critiques des impacts de défaillance. L'intégration précoce et la

prédiction des caractéristiques de fiabilité, de disponibilité et de maintenabilité dans le flux de conception minimise la surcharge de telles itérations.

La complexité est induite par l'interaction des services de ses composants sous-jacents. Ces interactions matérialisent des flux ayant la morphologie des cascades, des dérivations, des cycles ou une combinaison de ces schémas. Ces modèles sont identifiés comme des structures, qui peuvent être soit en série, soit parallèles, soit mixtes. Dans une approche d'analyse RAM (Relaibility pour fiabilité, Availability pour disponibilité et Maintenability), un ensemble de services en interaction doit être appréhendable selon ces structures. Le plus souvent, cette étape préliminaire fait abstraction de la nature du service et se concentre uniquement sur l'interdépendance. Par exemple, dans une structure en série, quelle que soit sa nature, un service global nécessite que tous les services locaux soient opérationnels.

Dans une structure parallèle, le service global nécessite qu'au moins un service local soit opérationnel. On suppose, sans perte de généralité, que les pannes de composants ne sont jamais simultanées.

La spécification de l'interconnexion entre les composants est obligatoire, car elle détermine qualitativement les indicateurs RAM de l'ensemble du système. Ainsi, le service système revient à traiter un flux d'entrée, identifié comme ‘source’, et à produire une sortie identifiée comme la ‘cible’ par le support collaboratif de ses composants.

En outre, la structure du système détermine également la sensibilité aux comportements dysfonctionnels à anticiper. Le modèle comportemental du système global comprend un ensemble de configurations identifiables comme capables de fonctionner et des configurations hors service, nommées états de panne totaux, ainsi que toutes les transitions possibles entre ces configurations. Il est intéressant de noter que ce modèle est exhaustif car il représente toutes les séquences possibles conformes aux comportements de ses composants. Cela permet de raisonner sur des séquences spécifiques exprimant les comportements souhaités (réalisables), tels que les politiques de maintenance, la priorisation de réparation ou de mise en service, etc. Cette séparation entre les spécifications structurelles du système et les exigences d’exploitation en regard des états de panne est d'un grand intérêt.

C. Approche d’évaluation

Les exigences de gestion ou de pilotage sont fixées dans le but d’offrir un fonctionnement optimal du système. Cette optimalité service/coût nécessite dans le cadre de la sûreté de fonctionnement de pouvoir évaluer la capacité du système à produire en dépit des états de panne partielle, et des spécifications de récupération de service qui sont liées.

Par exemple, la redondance structurelle d’un système garantit qu’en cas de panne de plusieurs composants (ensemble d’états de panne partielle), le système est toujours en mesure de fournir un service avec une capacité de production vraisemblablement affaiblie. L’évaluation de la capacité de production du système associée à l’analyse de la propagation des défaillances permet non seulement d'analyser la criticité d’un composant mais aussi de définir la capacité de production dynamique du système en tenant compte des scénarios de récupération.

Ⅱ. Position et Motivations

Les modèles markoviens (et la formulation de leurs hypothèses) demeurent intéressants pour l'analyse des mécanismes de défaillance/réparation et l'évaluation des indicateurs de la sûreté de fonctionnement. Leur intégration (des modèles sous jacents) dans une approche Model Based Safety Assessment, est synonyme de cohérence, de complétude et de capitalisation de la connaissance, donc d’une conception rigoureuse d’ingénierie.

Les défaillances et les réparations peuvent être naturellement ressenties comme des événements se produisant dans un scénario complexe. Par conséquent, du fait de la manipulation d’états et des combinaisons événementielles, les techniques de modélisation relevant de la perception dynamique des

systèmes à événements discrets semble être bien adaptée pour appréhender cette complexité. Les approches matures reposent sur les réseaux de Petri stochastiques ou les processus de Markov, pour le calcul des indicateurs, la vérification et la simulation.

L'approche BDMP (Boolean logic Driven Markov Process) fournit une modélisation basée sur un arbre de défaillances pour générer les comportements dysfonctionnels sous-jacents en tant que processus de Markov. Les séquences d'échec sont explicites. Par exemple, les architectures de redondance froide sont modélisées précisément en considérant que la défaillance du composant redondant ne peut se produire qu'après la défaillance du composant principal, ce qui est exprimé par une séquence de défaillance appropriée.

AltaRica fournit une solution de modélisation discrète puissante sous la forme de systèmes de transition gardés. AltaRica offre une flexibilité de modélisation grâce à une ouverture vers d'autres langages de modélisation, tels que AADL, SysML et Event-B.

A. Positionnement

Pour en revenir à une démarche d'ingénierie sûre, MBSA fournit un cadre de développement à condition que l'ensemble des modèles sous-jacents soit conforme à trois exigences:

1. Il est capable de décrire l'ensemble des comportements dysfonctionnels. Pour un système industriel complexe, impliquant plusieurs composants en interaction, chaque composant doit être modélisé en fonction de son comportement dysfonctionnel. Le comportement global est obtenu à partir de la combinaison de comportements locaux, selon une structure systématique qui est conçue pour fournir le service système ;

2. Il est capable de cibler des configurations spécifiques, et de fournir des mécanismes pour associer des trajectoires dysfonctionnelles à ces configurations. Ces configurations sont généralement modélisées par des états. Par exemple, un modèle de composant peut présenter plusieurs états de fonctionnement nominaux, tels que repos, fonctionnement, panne partielle, panne totale ;

3. Il est capable d'intégrer des exigences supplémentaires, telles que des aspects qualitatifs ou quantitatifs de service. Les exigences qualitatives peuvent être caractérisées par des contraintes structurelles ou de priorité. Par exemple, dans le cas où plusieurs composants sont en attente de réparation, les composants les plus critiques doivent avoir la priorité d'être réparés en premier. Les aspects quantitatifs sont principalement liés aux coûts des scenarios de réparation. Des estimations supplémentaires de la 'santé résultante du système réparé peuvent également être prises en compte.

Afin de garantir l'exactitude de l'évaluation, ces aspects doivent être exprimés et traités de manière formelle. Pourtant, à notre connaissance aucun résultat de recherche n'existe jusqu'à présent, évaluant les indicateurs RAM à la structure et aux exigences opérationnelles de service. Par exemple, les modèles générés à partir de GRIF ou d'AltaRica manque d'extension pour ces considérations opérationnelles. À son tour, BDMP est capable de prendre en compte les séquences d'échec commençant par un événement déclencheur, prend également en compte les comportements complexes des composants (par exemple, le taux de réussite à la mise sous tension et le comportement de 'reconfiguration'), mais ne permet pas d'intégrer des exigences de trajectoire (séquentialité de réparation).

B. Originalité

L'originalité de cette recherche est de fournir une approche unifiée pour l'évaluation de la sûreté de fonctionnement (notamment fiabilité dans un premier temps) des systèmes structurés en tenant compte des exigences de service. L'objectif consiste alors à générer un modèle dysfonctionnel intégrant dysfonctionnement, structure et exigences.

Plusieurs étapes formelles de modélisation et de transformation sont préconisées, traitant successivement du comportement dysfonctionnel et de la structure du système au regard des propagations de défaillance et des exigences de service.

La première étape vise à relier de manière exhaustive les configurations de panne globale du système aux comportements dysfonctionnels des composants locaux. Chaque comportement dysfonctionnel de composant est formellement modélisé comme un système à événements discrets. Le comportement dysfonctionnel global du système est obtenu formellement en combinant les comportements locaux de tous les composants. Le modèle global obtenu par composition représente alors toutes les configurations d'état de fonctionnement ou de panne.

La deuxième étape consiste à identifer formellement les états, notamment à distinguer les états de panne locale et globale au niveau du système. A ce titre, une analyse arborescence de défauts (FtA) est appliquée sur la description structurelle, il en est déduit une expression logique de déclaration des états de panne. Les plus simples combinaisons des défaillances des composants critiques provoquant la panne de l'ensemble du système sont exprimées sous la forme de coupe minimal. L'ensemble des coupes minimales constitue l'expression logique d'échec de service du système. Il s'agira par la suite d'identifer formellement par une procédure d'étiquetage appropriée ces états sur les modèles de comportements dysfonctionnels.

À ce stade, des services spécifiques tels que les exigences de réparation peuvent être intégrés. Ceux-ci sont définis comme des modèles comportementaux, décrivant éventuellement des séquences ou des priorités de réparation souhaitées, ou d'autres comportements opérationnels. Cette troisième étape produit un modèle global qui présente les comportements dysfonctionnels liés aux exigences de service.

La quatrième et dernière étape cible les configurations de panne globale dans le modèle comportemental en vue des objectifs de l'analyse, fiabilité, disponibilité ou maintenabilité. Sans perte de généralité, seul est présenté dans ce travail l'identification de la panne totale, comme état absorbant du modèle à des fins de fiabilité dite opérationnelle *i.e.* tenant compte des spécifications de service.

Face à la fiabilité opérationnelle, trois objectifs peuvent être abordés :

1. Fiabilité opérationnelle, en supposant l'absence de perturbation à l'état initial et l'accessibilité à un état de panne total, ce modèle généré inclue les exigences spécifiques de service ou de reprise ;

2. Maintenabilité opérationnelle, en considérant que le système est initialement en panne totale, le modèle généré définit le retour à un état de fonctionnement sain du système sous les exigences spécifiques de service ou de reprise ;

3. Disponibilité opérationnelle, en considérant que le système fonctionne initialement, le modèle généré définit les cycles panne-récupération sous exigences de service.

Pour rappel, ce travail traite uniquement de la fiabilité opérationnelle. Ainsi, chaque composant peut être raisonnablement supposé « réparable ». Cependant pour respecter l'objectif de fiabilité, l'état de panne totale est identifié comme état absorbant dans le modèle. Afin d'exprimer cette particularité, aucune réparation ne doit être autorisée dans les configurations de panne globale. La transformation supplémentaire du modèle comportemental étiqueté est alors nécessaire.

Le système qui en résulte présente des trajectoires déclenchées par des événements menant à une panne globale définitive. Ces événements sont en général décrits stochastiquement par les taux d'occurrence. Par conséquence, une chaîne de Markov en temps continu est dérivée du modèle comportemental étiqueté et offre les calculs habituels pour l'évaluation de la fiabilité.

Alternativement, au-delà du calcul des probabilités des impacts des occurrences de défaillance, il apparaît intéressant d'évaluer l'efficacité de la politique de gestion des réparations locales. Le cadre

présenté dans ce travail propose cette fonctionnalité, à travers une simulation quantitative des coûts, en fonction de l'événement et / ou des coûts de configuration globale.

En conclusion, ce travail fournit un cadre de modélisation formel, basé sur la génération comportementale modulaire, capable d'évaluer la fiabilité opérationnelle du système, en prenant en compte des spécifications spécifiques de service ou de réparation et en évaluant leur efficacité. Une méthodologie d'analyse quantitative est développée, pour le calcul de la capacité de reprise de service du système, prenant en compte les défaillances locales des composants.

Ⅲ. Proposition et Originalité

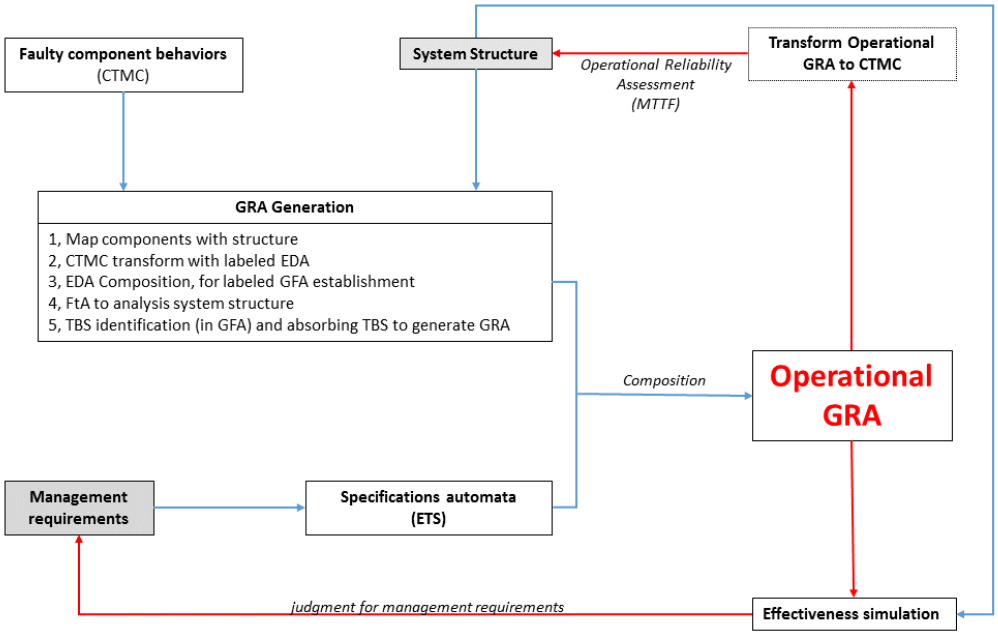


Figure1. cadre de proposition

Au plus proche des exigences de conception et de pilotage de l'ingénierie, la proposition ambitionne de générer des modèles formels du comportement dysfonctionnel intégrant les spécifications de recouvrement de service (attribut opérationnel). La figure 1 illustre l'approche retenue respectant le cadre de modélisation modulaire cohérente aux précepts du MBSA. En spécifiant les exigences de gestion des états de service dans le modèle cible et en procédant à l'évaluation de l'efficacité de service, les indicateurs de la sûreté de fonctionnement sont ici désignés comme indicateurs opérationnels. Ainsi la fiabilité opérationnelle peut être prise en compte.

Le modèle cible est appelé « GRA opérationnel » pour Global Reliability Automaton, il relève de la composition de deux entrées principales, le modèle GRA nominal (modèle SED à base d'automates) et les exigences de gestion telles que planification/priorité de réparation également définies sous forme de modèle SED à base d'automates.

Le modèle GRA nominal (non contraint par les exigences) relève de la connaissance des comportements des composants défectueux au travers de modèles markoviens à temps continu (CTMC), de l'architecture fonctionnelle du système sous forme de structure de causalité de service (structure série, redondante, à vote…) et de l'identification des états de panne totale (panne système).

La génération du modèle GRA nominal est l'aboutissement de cinq étapes :

1 une définition de CTMC locales,

2 une transformation des CTMC en automate étiqueté (Event Driven Automaton),
3 une composition selon la structure de service des EDA générant le GFA (Global Faulty Automaton),
4 l'identification du TBS (Total Broken Down State) sur le GFA à partir des coupes minimales de l'arbre de faute (FtA)
5 le GRA résulte de la modification du GFA en rendant l'ensemble des TBS absorbants.

Le GRA opérationnel peut alors être exploité par divers outils d'évaluation, telle l'efficacité de production présentée ici.

A. Approche de génération de modèle de fiabilité

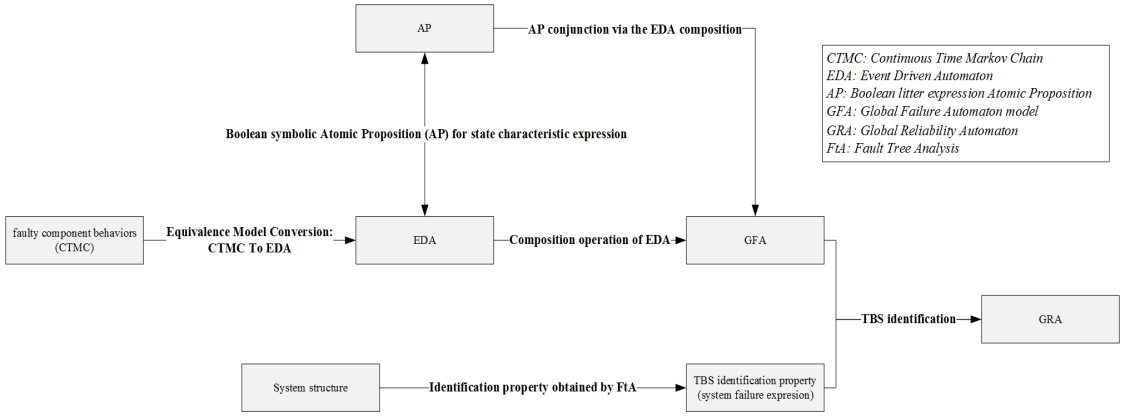


Figure 2. cadre de la génération GRA

La génération du GRA (nominal) est rappelé à la figure 2. Les chaînes de Markov à temps continu (CTMC) constituent les entrées de modélisation, les taux d'occurrence des événements associés aux transitions dans les CTMC sont équivalents aux événements de transition dans un automate régulier EDA (Event Driven Automaton), ceci est conforté par le travail de Ionescu qui établi la transformation du modèle CTMC en EDA. Une fois défini les équivalences de modèle (CTMC-EDA) leur composition est rendue formellement possible. L'automate GFA (Global Faulty Automaton) est ainsi établi. Le modèle de fiabilité résultera de l'identification dans ce modèle des états de panne totale (TBS), ces états de GFA seront rendus absorbants pour générer le GRA.

L'adjonction d'étiquettes permet l'identification des états du modèle EDA, et donc des TBS lorsqu'ils ont été exprimés par les coupes minimales de l'arbre de faute lié à la structure du système étudié.

1. Conversion des modèles d'équivalence : CTMC et EDA

Il s'agit ici de définir les équivalences des modèles stochastiques CTMC et automates réguliers EDA afin de permettre la composition formelle de modèles et d'accéder ainsi du comportement dysfonctionnel du local (composant) au global (système).

Soit une CTMC définie par :

$$CTMC = (Q_{mc}; E_{mc}; \delta_{mc}; q0_{mc})$$

Q_{mc} est l'ensemble d'états, Emc est l'ensemble de transition, δmc est la fonction de transition, q0 est l'état initial. La dynamique du modèle peut être présentée par l'union des séquences d'événements associées aux états, l'ensemble des états associés xi est équivalent à l'état défini dans CTMC:

$$\{x_i\} = Q_{mc}$$

La fonction de transition dans une CTMC décrit l'évolution des états conduite par les événements.

Soit un EDA (Event Driven Automaton) défni par :

$$EDA = (Q_{EDA}; E_{EDA}; \delta_{EDA}; q0_{EDA}; qm_{EDA})$$

Q_{EDA} est l'ensemble d'états, E_{EDA} est l'ensemble de transition, δ_{EDA} est la fonction de transition, $q0_{EDA}$ est l'état initial et qm_{EDA} est l'état marqué. Par le même principe que précédemment, on peut avoir :

$$\{x_i\} = Q_{EDA}$$

Étant donné que CMTC et EDA sont modélisés pour le même système, les transitions et les états sont identiques :

$$Q_{EDA} = \{x_i\} = Q_{mc} \text{ et } \delta_{mc} \leftrightarrow \delta_{EDA}$$

pour le modèle EDA: $\delta_{EDA}(x, e_{xy}) = y$ et pour le modèle CTMC: $\delta_{mc}(x, e_{xy}) = y$, de taux:$R\text{-}e_{xy}$.

2. Opération de composition d'EDA

L'équivalence comportementale des modèles CTMC et EDA ayant été démontrée, la composition des comportements défectueux des composants, décrite par CTMC peut être abordée formellement par composition des EDA équivalents.

3. Proposition atomique booléenne (AP) pour identification les états

La proposition atomique booléenne (AP) appliquée aux modèles EDA permet d'étiqueter les états et ainsi assurer une identification symbolique. Les modèles EDA sont alors étendus de la sorte :

$$EDA = (Q; E; \delta; q0; qm; AP; L)$$

AP est l'ensemble des propositions atomiques exploités dans l'identification des états fautifs ou non. Le symbole booléen F (signifie 'fails ') associé à un état signifie que celui-ci est un état de panne, et sa négation ¬ F indique un état de fonctionnement correct. Il est alors possible d'obtenir par conjonction d'étiquettes (AP), la caractérisation des états combinés résultant d'opération de composition de modèles.

Supposons l'ensemble AP :

$$AP = \{ap_0, ap_1, ap_2 \ldots\ldots ap_n\}$$

L'ensemble de puissance de AP : $2^{AP} = \{\varnothing$, ap0， ap1， ap2， ap0∧ap1, ap0∧ap2, ap1∧ap2, ap0∧ap1∧ap 2…… $\}$

L'opération de composition des AP appliquée à la composition de modèles EDA défini le Global Faulty Automaton GFA comme suit :

$$GFA = EDA1//EDA2 = AC\ (Q1 \times Q2;\ E1 \cup E2;\ \delta;\ q01 \times q02;\ qm1 \times qm2;\ AP1 \cup AP2,;\ L1 \wedge L2)$$

$AP1 \cup AP2$ est l'union de l'ensemble de propositions atomiques. $L1 \wedge L2$ est la fonction d'association d'AP pour EDA1//EDA2 est:

$$L1 \wedge L2 : Q1 \times Q2 \rightarrow 2^{AP1 \cup AP2}$$

L'expression des symboles AP relève de situations réelles du système, et la fonction d'association est motivée par la signification de l'état. Si un état a plus d'une caractéristique, il s'agira d'un élément combiné (par exemple, ap0∧ap1) dans 2^{AP} associé à cet état. En outre, si chaque nom d'état est unique, mais il est possible que plusieurs états partagent le même AP.

4. Propriété d'identification obtenue par FtA

Les états sont ainsi identifiés par leur symbole. Les diverses combinaisons logiques d'AP sont l'identification des états composés. Tout état peut être ainsi identifié, il en va de même pour pour l'identification du TBS. Le TBS est un état redouté dans le système, il résulte de la propagation d'un événement (ou séquence) redouté.

Les arbres de fautes (FtA) expriment cette propagation par la combinaison logique d'événement initiaux. Cette combinaison résulte de la structure collaborative des composants constituant le système. L'entrée des feuilles de FtA portera un symbole AP typé défaillance, et l'expression logique de l'événement redouté est alors la combinaison des AP de composants. Ainsi, le TBS (Total Breakdown State) peut être identifié dans GFA.

5. Génération du GRA (Global Reliability Automaton)

La génération du GRA résulte de l'identification du TBS sur le GFA et de la transformation des états TBS en états absorbants *i.e.* supprimer sur le modèle GFA, les transitions de sortie de l'ensemble des états étiquetés TBS. Comme certaines transitions de sortie ont été supprimées, il est possible qu'il existe plusieurs états inaccessibles à partir de l'état initial. Ces états n'ont aucun sens dans GRA et sont supprimés.

B. Prise en compte de la fiabilité opérationnelle

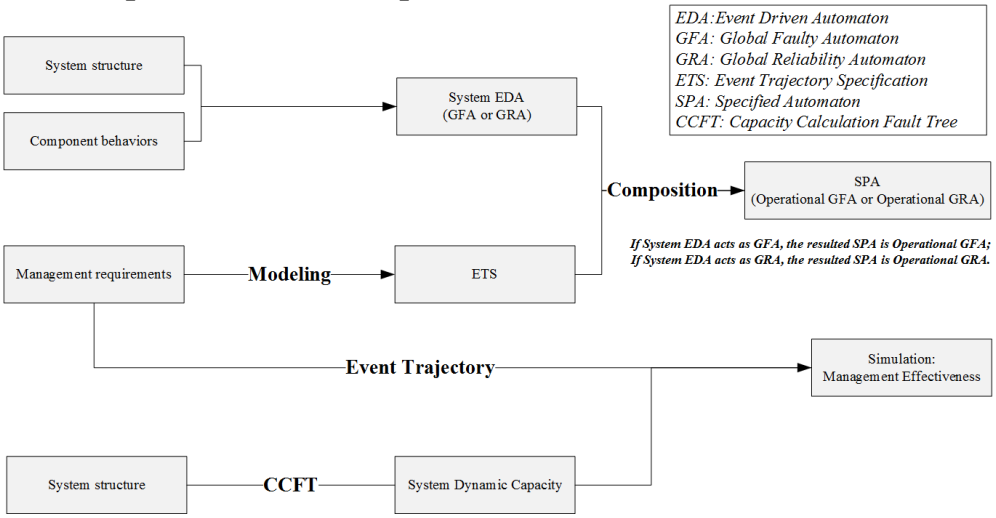


Figure 3. cadre de fiabilité opérationnelle

La figure 3 illustre l'enrichissement du modèle GFA (Global Faulty Automaton) à la notion d'opérationalité. Autrement dit, il s'agit de contraindre le comportement du système par la considération de spécification typée Sûreté de Fonctionnement (ici d'une trajectoire de reprise de fonctionnement après panne, ETS Event Trajectory Specification Automaton). Cette prise en compte dans le modèle GFA, caractérise la fiabilité opérationnelle. L'intégration de ces exigences selon l'approche multimodèle retenue reste alors conforme au cadre MBSA. Par la composition formelle du GRA et de l'ETS, le GRA opérationnel est établi.

1. Prise en compte des exigences de reprise dans le cadre de la fiabilité opérationnelle

Dans le domaine de l'application de la théorie des systèmes d'événements discrets, les spécifications sont modélisées sous la forme d'un automate régulier, à la fois pour manipuler des trajectoires d'exécution et surtout rester cohérent au modèle dysfonctionnel préalablement retenu (automate).

L'automate de spécification ETS est ainsi défini :

$$ETS=(Q_{ETS};E_{ETS};\delta_{ETS};q0_{ETS};qm_{ETS})$$

Q_{ETS} est l'ensemble d'états, E_{ETS} est l'ensemble de transition, δ_{ETS} est la fonction de transition, $q0_{ETS}$ est l'état initial et qm_{ETS} est l'état marqué.

La composition de GRA avec l'automate de spécification détermine le GRA opérationnel.

GRA Operationnel = GRA // ETS

Ainsi, la fonction de transition de GRA et ETS devient :

$$\delta_{GRA}((\mathrm{x.y}),\mathrm{e}) = \begin{cases} (\,\delta_{GRA}(x,e)\ .\delta_{\mathrm{ETS}}(y,e)\,) & Si\ e \in\ E_{GRA} \cap E_{\mathrm{ETS}} \\ (\,\delta_{GRA}(x,e)\ .\ y\,) & Si\ e \in E_{GRA}\backslash E_{GRA} \cap E_{\mathrm{ETS}} \\ (\,x\ .\ \delta_{\mathrm{ETS}}(y,e)\,) & Sie \in E_{ETS}\backslash E_{GRA} \cap E_{\mathrm{ETS}} \\ indéfini & autrement \end{cases}$$

2. Quantification

Le choix d'une séquence de reprise (scenario) influence naturellement la capacité de production du système. Une analyse quantitative basée sur une représentation séquentielle arborescence a été déployée pour évaluer ces impacts (nommée CCFT: Capacity Calculation Fault Tree). L'exécution du scenario est en fait une simulation où le 'coût économique', le 'coût temporel' et la 'capacité du système' sont analysés en fonction des exigences de reprise engagée.

Les exigences de reprise sont fixées dans le but d'offrir le fonctionnement optimal du système, il s'agit de définir cette optimalité par un jeu de simulation/comparaison obtenu à partir de l'analyse CCFT (Capacity Calculation Fault Tree). Le CCFT est en mesure de calculer la capacité du système du système en tenant compte de la panne partielle et de la reprise.

Le modèle CCFT manipule deux portes logiques 'et' et 'ou'. Si un composant tombe en panne, sa capacité de production est nulle et est d'un certain niveau s'il fonctionne correctement. Sur une base de FtA, dans le cas où le système fonctionne correctement, la capacité de production du système est la somme des capacités des composants. En cas de panne du système, la capacité de production du système est nulle.

L'efficacité est caractérisée par trois éléments : 'coût économique', 'coût temporel'et 'capacité du système'. Le 'coût économique' désigne le montant dû à l'opération de réparation, le 'coût temporel' représente le temps d'arrêt des composants. Chaque état étant identifié par sa capacité de production dans le modèle GRA opérationnel, la capacité de production du système peut être calculée. L'état du modèle GRA opérationnel est caractérisé non seulement son label, mais également par la combinaison des étiquettes pour présenter les situations de production du système (composants en panne, composants en fonctionnement, TBS…). L'exploitation du CCFT permet d'évaluer la capacité résultante.

Selon les commutations d'états possibles du modèle GRA opérationnel, les états accessibles du scénario de réparation (le scénario de réparation est exécuté par exigence de gestion) sont connus et les capacités du système dans les états atteignables peuvent être ainsi calculées par le CCFT. Le 'coût économique', le 'coût en temps' et la 'capacité de production du système' correspondante peuvent être simulés, en fonction des exigences de reprise. La simulation quantitative est une contribution complémentaire à la prise en compte de la sûreté de fonctionnement. L'exploitant du système est en mesure de choisir une exigence de reprise appropriée, à l'aide de cette approche de simulation.

C. Etude de cas

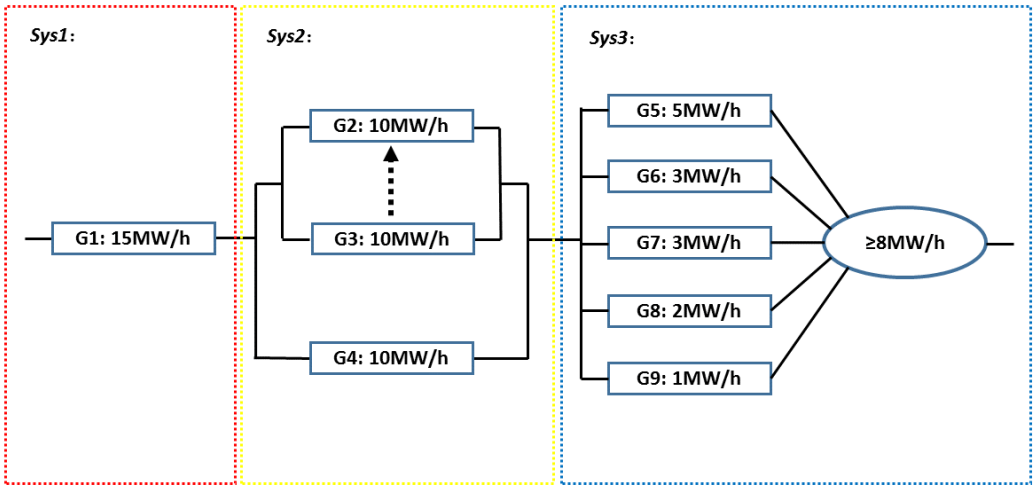


Figure 4. miniplant

1. Modélisation des comportements locaux dysfonctionnels

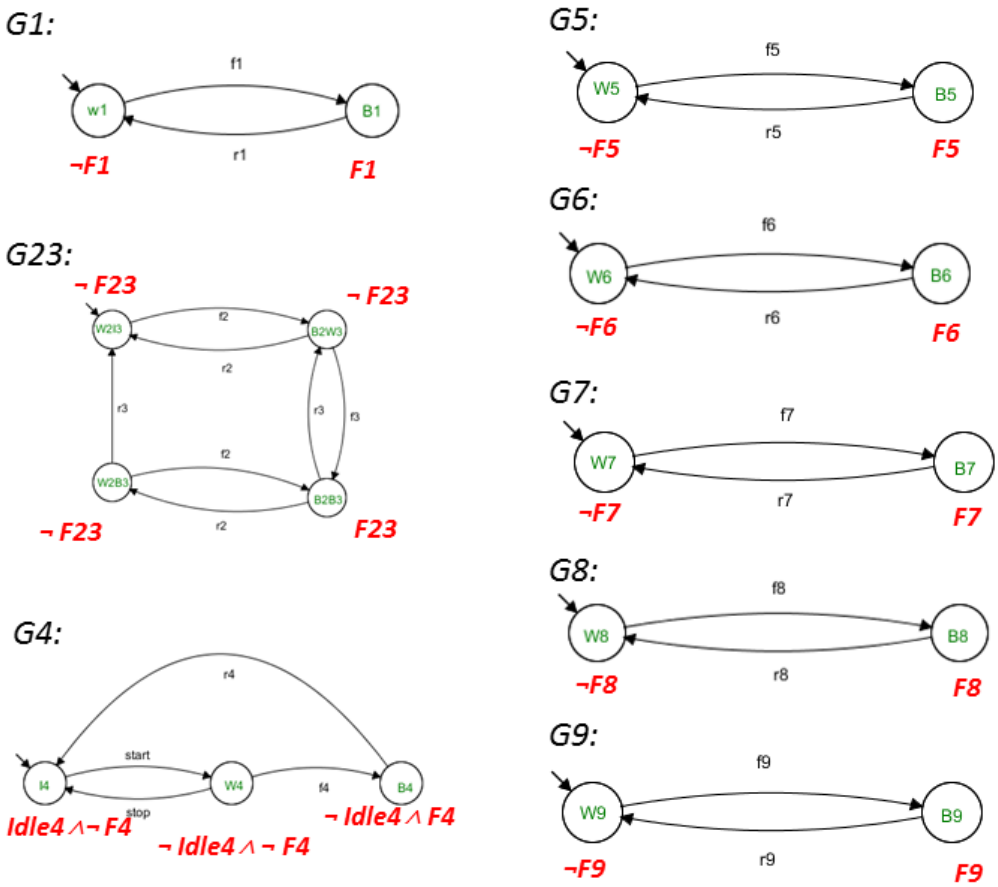


Figure 4 comportements locaux dysfonctionnels

2. Modélisation des exigences de reprise

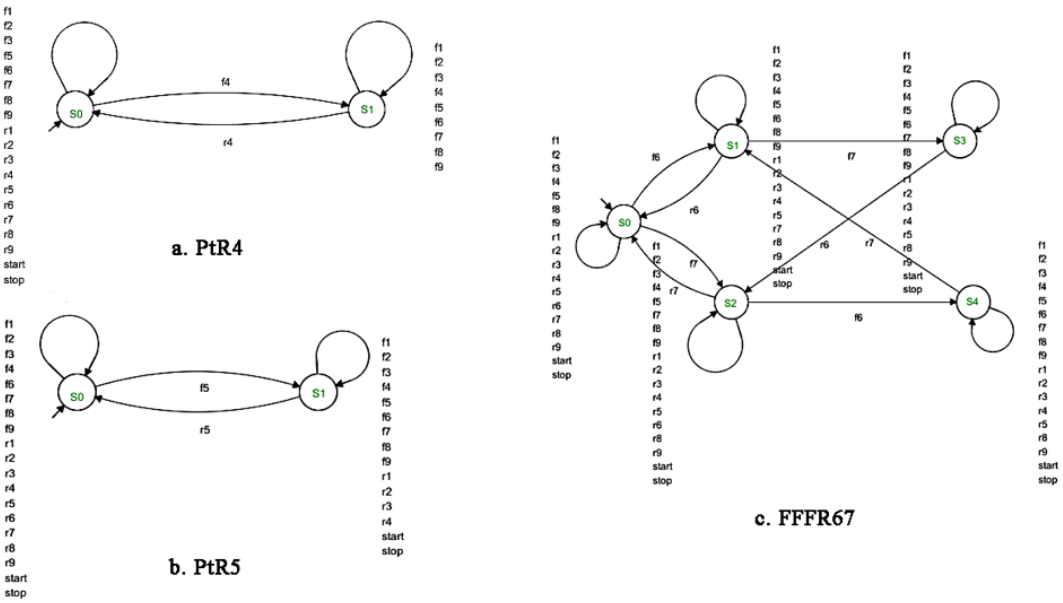


Figure 5. exigences de reprise

Le modèle GRA opérationnel résultant comporte pour ce cas d'étude plus de 3 800 états, de sorte que son exploitation n'est possible qu'au travers d'outils d'évaluation (de type GRIF). L'utilisation du CCFT associé est par contre avantageuse et de fait met en avant la simulation de trajectoire. Les résultats sont les suivants :

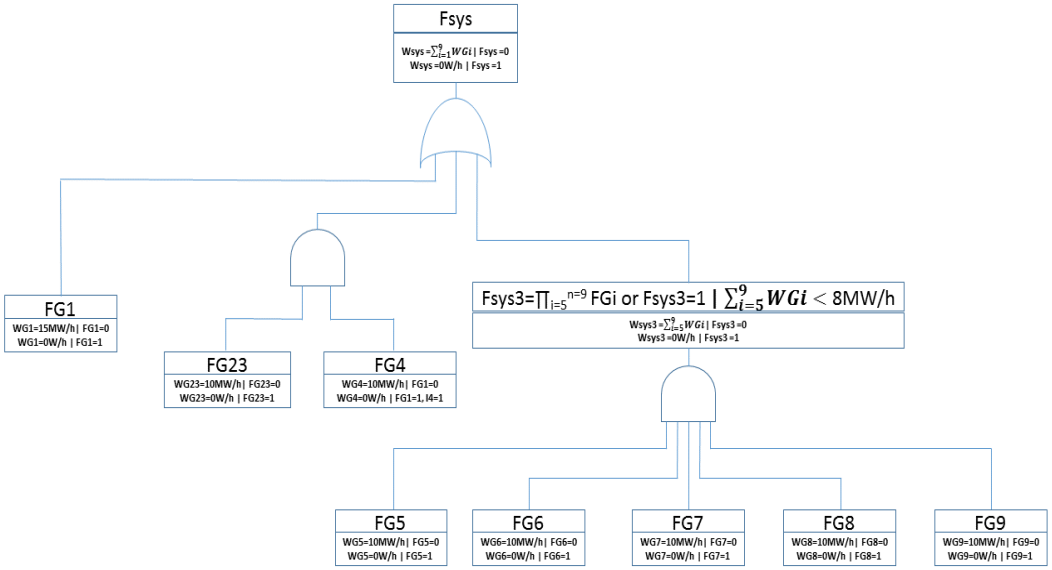
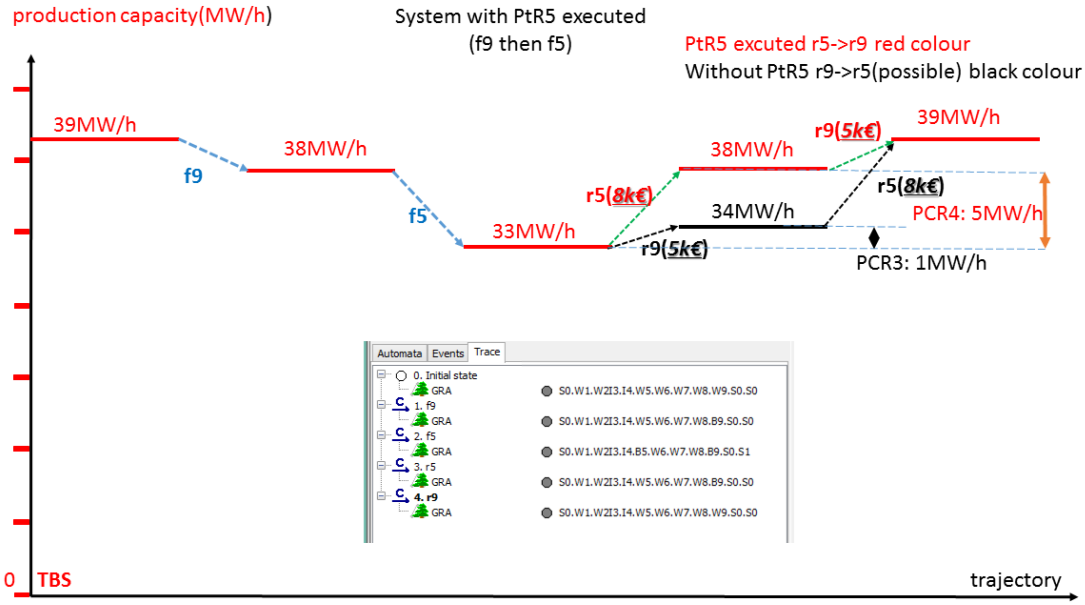


Figure 6. CCFT

3. Résultat de simulation



Ⅳ. Conclusions et Perspectives

A. Conclusions

Les outils de modélisation et d'évaluation des indicateurs de la sûreté de fonctionnement sont nombreux et présentent pour chacun d'entre eux des avantages et inconvénients connus. La particularité de l'approche MBSA consiste à formuler des techniques proches de celles utilisées par l'ingénierie. Par la considération d'exigences (contraintes de fonctionnement après panne) exprimées sous un même formalisme que les modèles dysfonctionnels, cette proposition conforte le cadre MBSA. Par rapport aux approches de génération de modèle classique, un changement ou un ajout de spécification conduit toujours à une nouvelle formulation du modèle, ce qui est à la fois consommateur de ressources et sujet d'erreurs.

Dans certains cas, comme le modèle a déjà été spécifié, l'ajout d'une exigence supplémentaire implique un nouveau modèle à condition que les multiples exigences ne soient pas en conflit. La levée des conflits est rendue simple dans la démarche de modélisation modulaire ici proposée, elle consiste à exploiter les outils formels de la théorie des automates. L'approche de génération du modèle de fiabilité opérationnelle retenu dans cette thèse répond à ces principes.

Les composants sont supposés présenter une indépendance stochastique dans les occurrences de défaillance/réparation, le commutateur dans la structure de vote est idéal et les comportements dysfonctionnels parmi les composants sont asynchrones (pas de mode commun). La théorie des systèmes à événements discrets est exploitée ici à la fois pour la génération de modèles et d'établissement de preuves (équivalence entre EDA et CTMC). La formulation en automate valide l'accessibilité à des états spécifiques tels que TBS, état ou ensemble d'états critiques. La génération est issue de la composition de modèles (comportement dysfonctionnel, exigences). Cette simple opération de composition pallie la complexité horizontale (formulation modulaire). Ainsi dans le modèle généré, les comportements défectueux, les configurations complexes et les exigences opérationnelles sont pris en compte.

Pour exprimer pleinement le scénario de défaillance ou de reprise, divers raisonnements en cas de défaillance et en situations d'inactivité ou de maintenance peuvent être pris en compte. Le modèle global défectueux du système est généré par la composition des modèles de composants, dans lesquels le scénario de défaillance de l'ensemble du système est explicite.

Sur la base de l'expression des coupes minimales d'un arbre de fautes, les états de panne totale sont identifiés et repérés dans le modèle dysfonctionnel. Ces états sont rendant absorbants dans le modèle global pour définir un modèle de fiabilité. Enfin, le modèle d'arbre de défaillances de calcul (CCFT), évalue l'efficacité des exigences par simulation.

Les faits marquants de ce travail se situent dans une approche collaborative entre ingénierie et sûreté de fonctionnement au travers du concept MBSA (model based safety assessment). Les aspects formels sur lesquels reposent la proposition (composition d'automates réguliers) sont cohérents à la rigueur de conception d'une part et aux exploitations qui y sont conséquentes d'autre part, notamment en termes de propriétés de validation. A ce titre, ce travail relève autant de la méthodologie que de la modélisation.

Ce travail présente un caractère exploratoire, et de ce fait est limité : pas d'interface directe avec les outils de calcul d'indicateurs de fiabilité (par exemple GRIF ou AltaRica). Le modèle global est issu des modèles de composants locaux par le biais de la composition augmentant la complexité. Ce qui présente un risque potentiel d'explosion d'états. Les structures ne sont analysables par les coupes minimales que si les hypothèses fondées sur les RBD ou les FtA conviennent. Les structures dynamiques de certains arbres de fautes ou les structures mixtes (continu/discret) sont exclues de l'étude. Le CCFT est exploitable sur la séquentialité des scenario.

B. Perspectives

Les travaux en cours ouvrent des perspectives intéressantes :

Augmentation des états fonctionnels et dysfonctionnels dans les modèles automate à N-états : par exemple adjonction des états « inactif », «fonctionnement dégradé » et «panne partielle », « panne totale », « maintenance »,… car plus le système est complexe et plus d'états sont nécessaires pour présenter le scénario de défaillance ou de reprise dans le modèle.

La génération du modèle de fiabilité opérationnelle est introduite dans cette thèse. Les modèles de disponibilité opérationnelle et de maintenabilité opérationnelle peuvent également être générés selon une approche identique. Les hypothèseses initiales sont modifiées et les spécifications sont particulières à l'objectif visé.

Le formalisme utilisé ici repose sur les modèles à automates, les réseaux de Petri peuvent présenter des intérêts dans la simulation dysfonctionnelle car ces derniers sont plus concis. Par contre, la composition reste un problème ouvert. Cependant, subséquent à ces modèles, l'expressivité des propriétés formelles des systèmes à événements discrets (vivacité, atteignabilité) demeure fortement interessante et supporter les étapes de développement.

Les aspects quantitatifs restent totalement prospectifs. Au-delà des limites d'exploitation des arbres de défaillance pour le calcul de capacité (CCFT), les portes portes dynamiques ou des déclencheurs d'événements pour décrire des scénarios de défaillance plus complexes serait d'un apport certain.

Bibliography：

[1] R. F. Stapelberg, *Handbook of reliability, availability, maintainability and safety in engineering design*, Springer Science & Business Media, 2009.

[2] D. Noyes, and F. Pérès, “Analyse des systèmes-Sûreté de Fonctionnement,” *Techniques de l'ingénieur*, 2007.

[3] M. K. Molloy, “Performance analysis using stochastic Petri nets,” *IEEE Transactions on computers*, no. 9, pp. 913-917, 1982.

[4] I. A. Papazoglou, and E. P. Gyftopoulos, “Markov processes for reliability analyses of large systems,” *IEEE Transactions on Reliability,* vol. 26, no. 3, pp. 232-237, 1977.

[5] L. Pièrre-Cambacédès, and M. Bouissou, "Modeling safety and security interdependencies with BDMP (Boolean logic Driven Markov Processes)."2010 IEEE International Conference on Systems, Man and Cybernetics,IEEE, pp. 2852-2861,2010.

[6] T. Prosvirnova, M. Batteux, P.-A. Brameret, A. Cherfi, T. Friedlhuber, J.-M. Roussel, and A. Rauzy, “The altarica 3.0 project for model-based safety assessment,” *IFAC Proceedings Volumes,* vol. 46, no. 22, pp. 127-132, 2013.

[7] N. Clavé, P.-j. Cacheux, A. Dutertre, and C. Vinuesa, "GRIF-Bool: évaluation des risques à l'aide d'un outil d'analyse Booléenne multi-approches."Congrès Lambda Mu 20 de Maîtrise des Risques et de Sûreté de Fonctionnement,Saint Malo, France,pp. 5E,2016.

[8] S. Sharvia, S. Kabir, M. Walker, and Y. Papadopoulos, "Model-based dependability analysis: State-of-the-art, challenges, and future outlook." In *Software Quality Assurance*, pp. 251-278, : Elsevier, 2016.

[9] J. I. Aizpurua, and E. Muxika, “Model-based design of dependable systems: limitations and evolution of analysis and verification approaches,” *International Journal on Advances in Security,* vol. 6, no. 1, pp. 12-31, 2013.

[10] M. Modarres, M. P. Kaminskiy, and V. Krivtsov, *Reliability engineering and risk analysis: a practical guide*, CRC press, 2016.

[11] J.-C. Laprie, “Sûreté de fonctionnement informatique: concepts, défis, directions,” *Computing,* vol. 1, no. 1, pp. 11-33, 2004.

[12] J.-F. Aubry, and N. Brinzei, *Systems Dependability Assessment*, ISTE Limited and John Wiley & Sons Incorporated, 2015.

[13] S. Malik, “IT infrastructure deployment project plan template and dashboard,” pp. 21, Feburary 11th, 2013.

[14] A. Demri, A. Charki, F. Guerin, and H. Christofol, "Functional and dysfunctional analysis of a mechatronic system."2008 Annual Reliability and Maintainability Symposium,IEEE, pp. 114-119,2008.

[15] M. Chibane, F. Mhenni, and J.-Y. Choley, "Product-Process MBSE and Dysfunctional Analysis Approach Applied to a Production Line."2019 20th International Conference on Research and Education in Mechatronics (REM),IEEE, pp. 1-7,2019.

[16] S. Jimeno, A. Riaz, M. Guenov, and A. Molina-Cristobal, "Enabling Interactive Safety and Performance Trade-offs in Early Airframe Systems Design."AIAA Scitech 2020 Forum,pp. 0550,2020.

[17] S. Sinha, N. Kumar Goyal, and R. Mall, “Early Prediction of Reliability and Availability of Combined Hardware-Software Systems based on Functional Failures,” *Journal of Systems Architecture*, pp. 23-38, 2018.

[18] D. A. Marca, and C. L. McGowan, *SADT: structured analysis and design technique*, McGraw-Hill,Inc 1987.

[19] P. Micouin, *Model Based Systems Engineering: Fundamentals and Methods*, John Wiley & Sons, 2014.
[20] Q. Li, and Y.-L. Chen, "Idef0 function modeling." In *Modeling and Analysis of Enterprise and Information Systems*, pp. 98-122, : Springer, 2009.
[21] J. M. Dorador, and R. I. M. Young, "Application of IDEF0, IDEF3 and UML methodologies in the creation of information models," *International Journal of Computer Integrated Manufacturing,* vol. 13, no. 5, pp. 430-445, 2000.
[22] L. P. Khoo, "An IDEF0 model-based intelligent fault diagnosis system for manufacturing systems," *International Journal of Production Research,* vol. 37, no. 1, pp. 35-48, 1999.
[23] L. Peters, and J. Peters, "Using IDEF0 for dynamic process analysis."Proceedings of International Conference on Robotics and Automation,IEEE, pp. 3203-3208,1997.
[24] J.-W. Kim, J.-S. Yon, and B. Cho, "Modeling, control, and design of input-series-output-parallel-connected converter for high-speed-train power system," *IEEE Transactions on industrial electronics,* vol. 48, no. 3, pp. 536-544, 2001.
[25] V. M. Bier, A. Nagaraj, and V. Abhichandani, "Protection of simple series and parallel systems with components of different values," *Reliability Engineering & System Safety,* vol. 87, no. 3, pp. 315-323, 2005.
[26] G. Levitin, "Incorporating common-cause failures into nonrepairable multistate series-parallel system analysis," *IEEE Transactions on Reliability,* vol. 50, no. 4, pp. 380-388, 2001.
[27] M. Sakawa, "Optimal reliability-design of a series-parallel system by a large-scale multiobjective optimization method," *IEEE Transactions on Reliability,* vol. 30, no. 2, pp. 173-174, 1981.
[28] F. A. Tillman, C.-L. Hwang, and W. Kuo, "Determining component reliability and redundancy for optimum system reliability," *IEEE Transactions on Reliability,* vol. 26, no. 3, pp. 162-165, 1977.
[29] P. Clemens, "Fault tree analysis," *JE Jacobs Severdurup*, pp. 13, 2002.
[30] R. Serfozo, *Basics of applied stochastic processes*, Springer Science & Business Media, 2009.
[31] F. L. Spitzer, *Random fields and interacting particle systems*, Mathematical Association of America Oberlin-Ohio, 1971.
[32] P. A. Gagniuc, *Markov Chains: From Theory to Implementation and Experimentation*, John Wiley & Sons, 2017.
[33] R. F. Serfozo, "An equivalence between continuous and discrete time Markov decision processes," *Operations Research,* vol. 27, no. 3, pp. 616-620, 1979.
[34] A. J. P. Tixier, M. R. Hallowell, and B. Rajagopalan, "Construction safety risk modeling and simulation," *Risk analysis,* vol. 37, no. 10, pp. 1917-1935, 2017.
[35] A. Platis, N. Limnios, and M. Le Du, "Dependability analysis of systems modeled by non-homogeneous Markov chains," *Reliability Engineering System Safety,* vol. 61, no. 3, pp. 235-249, 1998.
[36] W. R. Gilks, S. Richardson, and D. Spiegelhalter, *Markov chain Monte Carlo in practice*, Chapman and Hall/CRC, 1995.
[37] W. J. Anderson, *Continuous-time Markov chains: An applications-oriented approach*, Springer Science & Business Media, 2012.
[38] G. A. Fenton, and D. V. Griffiths, "Risk assessment in geotechnical engineering," *John wiley&Sons,* vol. 461, pp. 381-400, 2008.
[39] H. Hermanns, "Interactive markov chains." In *Interactive Markov Chains*, pp. 57-88, : Springer, 2002.
[40] H. Boudali, P. Crouzen, and M. Stoelinga, "Dynamic fault tree analysis using input/output interactive markov chains."37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07),IEEE, pp. 708-717,2007.
[41] J. Conlisk, "Interactive markov chains," *Journal of Mathematical Sociology,* vol. 4, no. 2, pp. 157-185, 1976.

[42] Z. Ren, and B. H. Krogh, "State aggregation in Markov decision processes."Proceedings of the 41st IEEE Conference on Decision and Control, 2002.,IEEE, pp. 3819-3824,2002.
[43] C. Baier, B. Haverkort, H. Hermanns, and J.-P. Katoen, “Model-checking algorithms for continuous-time Markov chains,” *IEEE Transactions on software engineering,* vol. 29, no. 6, pp. 524-541, 2003.
[44] G. Tauchen, “Finite state markov-chain approximations to univariate and vector autoregressions,” *Economics letters,* vol. 20, no. 2, pp. 177-181, 1986.
[45] Z. Ghahramani, and M. I. Jordan, "Factorial hidden Markov models."Advances in Neural Information Processing Systems,pp. 472-478,1996.
[46] H. Hemi, J. Ghouili, and A. Cheriti, “Combination of Markov chain and optimal control solved by Pontryagin’s Minimum Principle for a fuel cell/supercapacitor vehicle,” *Energy Conversion and Management,* vol. 91, pp. 387-393, 2015.
[47] P. J. Haas, *Stochastic petri nets: Modelling, stability, simulation*, Springer Science & Business Media, 2006.
[48] V. Kumar, and K. Aggarwal, “Petri net modelling and reliability evaluation of distributed processing systems,” *Reliability Engineering & System Safety,* vol. 41, no. 2, pp. 167-176, 1993.
[49] N. G. Leveson, and J. L. Stolzy, “Safety analysis using Petri nets,” *IEEE Transactions on software engineering*, no. 3, pp. 386-397, 1987.
[50] J. B. Dugan, K. S. Trivedi, R. M. Geist, and V. F. Nicola, *Extended Stochastic Petri Nets: Applications and Analysis*, WISCONSIN UNIV-MADISON MOTOR BEHAVIOR LAB, 1984.
[51] A. Adamyan, and D. He, “Analysis of sequential failures for assessment of reliability and safety of manufacturing systems,” *Reliability Engineering & System Safety,* vol. 76, no. 3, pp. 227-236, 2002.
[52] W. G. Schneeweiss, “Tutorial: Petri nets as a graphical description medium for many reliability scenarios,” *IEEE Transactions on Reliability,* vol. 50, no. 2, pp. 159-164, 2001.
[53] T. Liu, and S. Chiou, “The application of Petri nets to failure analysis,” *Reliability Engineering & System Safety,* vol. 57, no. 2, pp. 129-142, 1997.
[54] V. Volovoi, “Modeling of system reliability Petri nets with aging tokens,” *Reliability Engineering & System Safety,* vol. 84, no. 2, pp. 149-161, 2004.
[55] M. H. Everdij, M. B. Klompstra, H. A. Blom, and B. K. Obbink, "Compositional specification of a multi-agent system by stochastically and dynamically coloured Petri nets." In *Stochastic Hybrid Systems*, pp. 325-350, : Springer, 2006.
[56] N. Fota, M. Kaaniche, and K. Kanoun, "A modular and incremental approach for building complex stochastic Petri net models."Proc First Int Conf on Mathematical Methods in Reliability,1997.
[57] P. Huber, K. Jensen, and R. M. Shapiro, "Hierarchies in coloured Petri nets."International Conference on Application and Theory of Petri Nets,Springer, pp. 313-341,1989.
[58] E. Kindler, "A compositional partial order semantics for Petri net components."International Conference on Application and Theory of Petri Nets,Springer, pp. 235-252,1997.
[59] W. Vogler, J. Hartmanis, and G. Goos, *Modular construction and partial order semantics of Petri nets*, Springer, 1992.
[60] M. Malhotra, and K. S. Trivedi, “Dependability modeling using Petri-nets,” *IEEE Transactions on Reliability,* vol. 44, no. 3, pp. 428-440, 1995.
[61] A. Ferscha, “A petri net approach for performance oriented parallel program design,” *Journal of Parallel Distributed Computing,* vol. 15, no. 3, pp. 188-206, 1992.
[62] K. Lodaya, and P. Weil, "Series-parallel posets: algebra, automata and languages."Annual Symposium on Theoretical Aspects of Computer Science,Springer, pp. 555-565,1998.
[63] R. Guillerm, H. Demmou, and N. Sadou, "ESA PetriNet: Petri net based tool for reliability analysis."2009 IEEE International Conference on Systems, Man and Cybernetics,IEEE, pp. 4740-4745,2009.
[64] R. Gaeta, A. Bobbio, G. Franceschinis, and L. Portinale, "Dependability assessment of an industrial programmable logic controller via parametric fault-tree and high level Petri

net."Proceedings 9th International Workshop on Petri Nets and Performance Models,IEEE, pp. 29-38,2001.

[65] J. Knezevic, and E. Odoom, “Reliability modelling of repairable systems using Petri nets and fuzzy Lambda–Tau methodology,” *Reliability Engineering & System Safety,* vol. 73, no. 1, pp. 1-17, 2001.

[66] F. Ahmad, H. Huang, and X. Wang, “A technique for reachability graph generation for the Petri net models of parallel processes,” *International Journal of Electrical Electronics Engineering,* vol. 3, no. 1, pp. 57-61, 2009.

[67] G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad, “A symbolic reachability graph for coloured Petri nets,” *Theoretical Computer Science,* vol. 176, no. 1-2, pp. 39-65, 1997.

[68] M. H. Hwang, and B. P. Zeigler, “Reachability graph of finite and deterministic devs networks,” *IEEE Transactions on Automation Science Engineering,* vol. 6, no. 3, pp. 468-478, 2009.

[69] B. Berthomieu*, P.-O. Ribet, and F. Vernadat, “The tool TINA–construction of abstract state spaces for Petri nets and time Petri nets,” *International Journal of Production Research,* vol. 42, no. 14, pp. 2741-2756, 2004.

[70] B. Berthomieu, and F. Vernadat, "Time Petri Nets Analysis with TINA."QEST,pp. 123-124,2006.

[71] M. Bouissou, "BDMP (Boolean logic Driven Markov Processes)® as an alternative to Event Trees."EDF R&D,Clamart, France,2008.

[72] M. Bouissou, and J.-L. Bon, “A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes,” *Reliability Engineering & System Safety,* vol. 82, no. 2, pp. 149-163, 2003.

[73] M. Bouissou, "A generalization of dynamic fault trees through Boolean logic driven Markov processes (BDMP)®."Proceedings of the16th European Safety and Reliability Conference (ESREL’07),2007.

[74] P.-Y. Piriou, J.-M. Faure, and J.-J. Lesage, “Generalized Boolean logic Driven Markov Processes: A powerful modeling framework for Model-Based Safety Analysis of dynamic repairable and reconfigurable systems,” *Reliability Engineering & System Safety,* vol. 163, pp. 57-68, 2017.

[75] M. Bouissou, "Automated dependability analysis of complex systems with the KB3 workbench: the experience of EDF R&D."Proceedings of the International Conference on ENERGY and ENVIRONMENT CIEM 2005,Citeseer,2005.

[76] M. Bouissou, S. Humbert, S. Muffat, and N. Villatte, "KB3 tool: feedback on knowledge bases."Proceedings of the11th European Safety and Reliability Conference (ESREL’02),2002.

[77] A. Arnold, G. Point, A. Griffault, and A. Rauzy, “The AltaRica formalism for describing concurrent systems,” *Fundamenta Informaticae,* vol. 40, no. 2, 3, pp. 109-124, 1999.

[78] T. Prosvirnova, “AltaRica 3.0: a model-based approach for safety analyses,” Computational Engineering, Finance, and Science[cs.CE], Ecole Polytechnique, 2014.

[79] M. Batteux, T. Prosvirnova, A. Rauzy, and L. Kloul, "The AltaRica 3.0 project for model-based safety assessment."2013 11th IEEE International Conference on Industrial Informatics (INDIN),IEEE, pp. 741-746,2013.

[80] M. Batteux, and A. Rauzy, "Stochastic simulation of AltaRica 3.0 models."Proceedings of the European Safety and Reliability Conference ESREL,pp. 1093-1100,2013.

[81] A. Griffault, and A. Vincent, "The Mec 5 model-checker."International Conference on Computer Aided Verification,Springer, pp. 488-491,2004.

[82] A. Griffault, and A. Vincent, "Vérification de modèles AltaRica."MAJECSTIC: Manifestation des jeunes chercheurs STIC,Marseille,France,2003.

[83] M. Boiteau, Y. Dutuit, A. Rauzy, and J.-P. Signoret, “The AltaRica data-flow language in use: modeling of production availability of a multi-state system,” *Reliability Engineering & System Safety,* vol. 91, no. 7, pp. 747-755, 2006.

[84] R. Adeline, P. Darfeuil, S. Humbert, J. Cardoso, and C. Seguin, "Toward a methodology for the AltaRica modelling of multi-physical systems."ESREL 2010,Rhodes, Greece,2010.

[85] F. Cassez, C. Pagetti, and O. Roux, “A timed extension for AltaRica,” *Fundamenta Informaticae,* vol. 62, no. 3-4, pp. 291-332, 2004.

[86] C. Pagetti, “Extension temps réel d'AltaRica,” Génie logiciel [cs.SE], Ecole Centrale de Nantes (ECN); Université de Nantes, 2004.
[87] P. David, V. Idasiak, and F. Kratz, "Automating the synthesis of AltaRica Data-Flow models from SysML."In Reliability, risk and safety: theory and applications: ESREL 2009,Taylor & Francis Group, pp. 105-112,2009.
[88] T. Ruin, E. Levrat, and B. J. I. P. V. Iung, “Modeling framework based on SysML and AltaRica data flow languages for developing models to support complex maintenance program quantification,” vol. 45, no. 31, pp. 169-174, 2012.
[89] M. Bozzano, A. Cimatti, O. Lisagor, C. Mattarei, S. Mover, M. Roveri, and S. Tonetta, “Symbolic model checking and safety assessment of altarica models,” *Electronic Communications of the EASST,* vol. 46, 2012.
[90] M. Bozzano, A. Cimatti, O. Lisagor, C. Mattarei, S. Mover, M. Roveri, and S. Tonetta, “Safety assessment of AltaRica models via symbolic model checking,” *Science of Computer Programming,* vol. 98, pp. 464-483, 2015.
[91] J.-C. Chaudemar, E. Bensana, C. Castel, and C. Seguin, "Altarica and event-b models for operational safety analysis: Unmanned aerial vehicle case study."Workshop on Integration of Model-based Formal Methods and Tools,2009.
[92] J. Brunel, P. Feiler, J. Hugues, B. Lewis, T. Prosvirnova, C. Seguin, and L. Wrage, "Performing safety analyses with AADL and AltaRica."International Symposium on Model-Based Safety and Assessment,Springer, pp. 67-81,2017.
[93] H. Mortada, T. Prosvirnova, and A. Rauzy, "Safety assessment of an electrical system with AltaRica 3.0."International Symposium on Model-Based Safety and Assessmemt,Springer, pp. 181-194,2014.
[94] Y. Zhu, J. Zhang, Q. Gong, Y. Fan, P. Wang, and C. Wang, "Reliability and safety assessment with AltaRica for complex aircraft systems."The Proceedings of 2011 9th International Conference on Reliability, Maintainability and Safety,IEEE, pp. 588-593,2011.
[95] S. Duo, and S. Li, “A practicable safety modeling methodology for aircraft systems using Altarica,” *Procedia Engineering,* vol. 80, pp. 127-139, 2014.
[96] T. Prosvirnova, and A. Rauzy, "AltaRica 3.0 project: compile Guarded Transition Systems into Fault Trees."European Safety and Reliability Conference, ESREL 2013,2013.
[97] S. Li, and X. Li, “Study on generation of fault trees from Altarica models,” *Procedia Engineering,* vol. 80, pp. 140-152, 2014.
[98] A. Rauzy, “Mode automata and their compilation into fault trees,” *Reliability Engineering & System Safety,* vol. 78, no. 1, pp. 1-12, 2002.
[99] M. Batteux, T. Prosvirnova, and A. Rauzy, "Advances in the simplification of Fault Trees automatically generated from AltaRica 3.0 models."2018.
[100] S. Friedenthal, A. Moore, and R. Steiner, *A practical guide to SysML: the systems modeling language*, Morgan Kaufmann, 2014.
[101] M. Hecht, E. Dimpfl, and J. Pinchak, "Automated generation of failure modes and effects analysis from SysML models."2014 IEEE International Symposium on Software Reliability Engineering Workshops,IEEE, pp. 62-65,2014.
[102] M. C. Kim, “Reliability block diagram with general gates and its application to system reliability analysis,” *Annals of Nuclear Energy,* vol. 38, no. 11, pp. 2456-2461, 2011.
[103] M. Alharby, and A. van Moorsel, “BlockSim: A Simulation Framework for Blockchain Systems,” *ACM SIGMETRICS Performance Evaluation Review,* vol. 46, no. 3, pp. 135-138, 2019.
[104] A. Lehman, N. O'Rourke, L. Hatcher, and E. Stepanski, *JMP for basic univariate and multivariate statistics: methods for researchers and social scientists*, Sas Institute, 2013.
[105] Y. Papadopoulos, and J. A. McDermid, "Hierarchically performed hazard origin and propagation studies."International Conference on Computer Safety, Reliability, and Security,Springer, pp. 139-152,1999.

[106] M. Gudemann, and F. Ortmeier, "A framework for qualitative and quantitative formal model-based safety analysis."2010 IEEE 12th International Symposium on High Assurance Systems Engineering,IEEE, pp. 132-141,2010.
[107] M. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0: Verification of probabilistic real-time systems."International conference on computer aided verification,Springer, pp. 585-591,2011.
[108] V. K. Garg, R. Kumar, and S. I. Marcus, “A probabilistic language formalism for stochastic discrete-event systems,” *IEEE Transactions on Automatic Control,* vol. 44, no. 2, pp. 280-293, 1999.
[109] V. K. Garg, "An algebraic approach to modeling probabilistic discrete event systems."Decision and Control, 1992., Proceedings of the 31st IEEE Conference on,IEEE, pp. 2348-2353,1992.
[110] D.-R. Ionescu, “Evaluation quantitative de séquences d’événements en sûreté de fonctionnement à l’aide de la théorie des langages probabilistes,” Automatique / Robotique, Université de Lorraine, 2016.
[111] C. G. Cassandras, and S. Lafortune, *Introduction to discrete event systems*, Springer Science & Business Media, 2009.
[112] D. Lo, and S.-C. Khoo, "QUARK: Empirical assessment of automaton-based specification miners."13th Working Conference on Reverse Engineering,IEEE, pp. 51-60,2006.
[113] K. Akesson, M. Fabian, H. Flordal, and A. Vahidi, "Supremica—a tool for verification and synthesis of discrete event supervisors."11th mediterranean conference on control and automation,2003.
[114] D. J. Robinson, *An introduction to abstract algebra*, Walter de Gruyter, 2008.
[115] M. Bouissou, H. Chraibi, and S. Muffat, "Utilisation de la simulation de Monte-Carlo pour la résolution d’un benchmark (MINIPLANT)."Actes du 14e congres de fiabilité et maintenabilité de l’IMdR (λμ14),2004.