REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE IBN KHALDOUN - TIARET

MEMOIRE

Présenté à :

FACULTÉ MATHEMATIQUES ET INFORMATIQUE
DÉPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER

Spécialité : Réseau et télécommunication

Par :

Lamrous Noureddine

Neki Abdelhak

Sur le thème

Diagnostic des pannes dans les systèmes dynamiques complexes (application aux réseaux de télécommunication)

Soutenu publiquement le 19 /11 / 2020 à Tiaret devant le jury composé de :

Mr. Mostefaoui Kadda	Grade	M.A.A	Univ Ibn Khaldoun	Encadreur
Mr. Laid Lahcen	Grade	M.C.B	Univ Ibn Khaldoun	Président
Mr .Mostefaoui Sid Ahmed	Grade	M.C.B	Univ Ibn Khaldoun	Examinateur

Année Universitaire : 2019 /2020

REMERCIEMENTS

Nous remercions d'abord la grâce du dieu, pour nous avoir guidés et éclairer sur la bonne voie du savoir pour continuer ce travail et atteindre les objectifs traces.

Nous exprimons nos remerciements particulièrement et les plus sincères à notre encadreur Mr MOSTEFAOUI KADDA de nous avoir encadré pour réaliser ce travail par ses précieux conseils et de nous avoir donné le meilleur de son savoir et aide. Nous remercions vivement Mr LAID LAHCEN, et Mr MOSTEFAOUI SID AHMED d'avoir accepté de faire partie de nos jurys du mémoire.

Nous remercions profondément tous ceux qui à faire de son mieux de près ou de loin dans l'élaboration de ce travail. Nous ne pouvons pas sans remercier nos parents qui sont la lumière de notre chemin et notre famille et amis.

Dédicace

Je dédie ce travail

A mes très chers parents, nulle dédicace n'est susceptible de vous exprimer ma profonde affection, mon immense gratitude pour tous les sacrifices que vous avez consacrés pour moi

A mes frères et sœurs, beaux-frères, nièces et neveu

A tous mes collègues et amies

A toute ma famille

A tous ceux qui m'aiment

NOUREDDINE

Dédicace

Je dédie ce modeste travail aux personnes les plus chères à mes yeux mes

Parents.

A ces deux grands cœurs qui m'entourent toujours par leur tendresse et leur affection.

A ceux qui m'ont toujours encouragée et soutenue dans mes études et m'ont éclairée et

Ouvert la vie de l'avenir.

Je vous dédie le fruit de mes efforts, comme un symbole de gratitude. Que Dieu vous me garde

Et que vous soyez toujours fières de moi.

A mes très chers frères

Dédiez ce travail au nouveau bébé

MARAM

Qui occupent une place particulière dans mon cœur

A tous mes amis, pour leur amitié, leur soutien moral, et leur conseil

En particulier

En souvenir de nos éclats de rire et des bons moments

Abdelhak

Résumé

La tâche de diagnostic consiste à détecter les anomalies intervenant sur un système puis à expliquer ces erreurs en indiquant les composants pouvant être défectueux. Pour cela, le diagnostic à base de modèles utilise une description du fonctionnement du système .Le défaut est détectée lorsque les données du modèle et les données du systèmes sont incohérentes.

Le but de ce mémoire est d'utiliser le formalisme DEVS pour diagnostiquer les systèmes d'événements discrets (DES). Le diagnostic basé sur le formalisme DEVS permet de détecter et localiser toute anomalie pouvant survenir dans le système. Ce système peut être simple ou complexe selon les besoins. Le formalisme DEVS nous a permis de modéliser certaines applications industrielles. Cette approche est basée sur une nouvelle structure qui sépare explicitement les états normaux des états défectueux dans le diagnostiqueur. Une telle distinction permet de suivre séparément l'évolution des traces normales et défectueuses dans le diagnostiqueur. L'évaluation de cette approche est illustrée par La modélisation et simulation d'un système complexe de réseau de télécommunication sous le simulateur DEVS SUITE.

Mots clés : DES, DEVS, diagnostiqueur, états normaux, états défectueux.

Abstract

The diagnostic task consists of detecting anomalies occurring on a system and then there errors by indicating the components that may be defective. For this, the Model-based diagnostics use a description of how the system works.is detected when model data and system data are inconsistent. The purpose of this thesis is to use the DEVS formalism to diagnose discrete event systems (DES). Diagnosis based on the DEVS formalism makes it possible to detect and locate any anomaly that may arise in the system. This system can be simple or complex as required. The DEVS formalism allowed us to model certain industrial applications. This approach is based on a new structure that explicitly separates normal states from faulty states in the diagnostic tool. Such a distinction makes it possible to follow the evolution of normal and defective traces in the diagnostician. The evaluation of this approach is illustrated on a complex telecommunication network system. We will see the modeling of this system and their implementation in the DEVS SUITE simulator.

Keywords: DES, DEVS, diagnoser, normal states, faulty states.

ملخص

تمكن مهمة التشخيص من اكتشاف حالات الأخطاء، مع الإشارة إلى المكونات التي قد تكون معيبة. لهذا، فإنها تستخدم التشخيصات القائمة على النموذج وصفًا لكيفية عمل النظام.يتم اكتشافه عندما تكون بيانات النموذج وبيانات النظام غير متسقة.*الغرض من هذه الأطروحة هو استخدام* شكلية DEVS لتشخيص أنظمة الأحداث المنفصلة .(DES) يتيح التشخيص المستند إلى شكلية DEVS اكتشاف وتحديد أي شذوذ قد ينشأ فيالنظام. يمكن أن يكون هذا النظام بسيطًا أو معقدًا حسب الحاجة. سمحت لنا شكليات DEVS بنمذجة الشاذة التي تحدث في النظام بعض التطبيقات الصناعية. يعتمد هذا النهج على بنية جديدة تفصل صراحةً الحالات الطبيعية عن الحالات السيئة في أداة التشخيص. مثل هذا التمييز يجعل من الممكن متابعة تطور الآثار الطبيعية والمعيبة في أداة التشخيص بشكل منفصل. يتم توضيح تقييم هذا النهج في نظام شبكة اتصالات معقد. سنرى نمذجة هذا النظام وتنفيذها في محاكي DEVS SUITE .

الكلمات المفتاحية: DES، DEVS ، التشخيص ، الحالات الطبيعية ، الحالات المعيبة.

Tale des matières

Chapitre 2 : Le formalisme DEVS

Chapitre 3 : Modélisation et simulation

Liste des figures

Introduction générale

Introduction générale

De manière générale, le diagnostic est le raisonnement menant à l'identification de la cause (l'origine) d'une défaillance, d'un défaut ou d'une maladie , à partir des caractères ou symptômes relevés par des observations, des contrôles ou des tests.

L'intérêt pour le diagnostic des défaillances s'explique par la complexité croissante des systèmes en particulier les systèmes à évènement discrets tel que le réseau de télécommunication qui sont de plus en plus exigeants en terme de contraintes de sécurité, de fiabilité, de disponibilité et de performances.

Dans ce cadre, le diagnostic est un processus intéressant de plus en plus pour les industriels. En effet, un processus de diagnostic peut permettre d'aider le réparateur en lui indiquant le symptôme (le conflit) et en lui fournissant une liste de composants pouvant expliquer le symptôme. Ainsi, cela résulte en un gain de temps qui se répercute par une diminution du coût de la réparation.

La diversité des approches qui ont été développées pour le diagnostic des systèmes à ED dynamiques complexes semblent être le résultat de contextes différents associés à la nature des applications visées et aux caractéristiques propres du cahier des charges qui en résulte. Ainsi, la nature des informations disponibles sur le système ou le type de défauts à détecter conduisent à la mise en œuvre de stratégies spécifiques.

Pour un réseau de télécommunication Les opérateurs de télécommunication font de plus en plus d'efforts pour fournir des services de très bonne qualité à leurs abonnés. En outre, les réseaux de télécommunication doivent être fiables et robustes pour garantir également la haute disponibilité de ces services. La gestion des réseaux est de ce fait devenue un problème central pour les opérateurs de télécommunication qui développent des travaux de recherches afin d'automatise rautant que possible les opérations complexes de gestion et d'administration des réseaux, telles que la gestion de pannes. Le diagnostic de pannes est un aspect central de la gestion et de l'administration d'un réseau. Une panne se définit comme la cause racine d'une ou de plusieurs anomalies survenues sur un réseau et observées sous forme d'alarmes ou de paramètres hors du standard de fonctionnement nominal de ce réseau.

Les alarmes ou symptômes sont des manifestations externes des pannes JAKOBSON et WEISSMANI [1993]. L'objectif principal du diagnostic est de retrouver aussi rapidement que possible les causes racines des anomalies observées qui interrompent ou dégradent la qualité de service fournie aux abonnés. Dans ce mémoire on utilise une approche basée sur le formalisme DEVS pour faire le diagnostic d'un système de réseau de télécommunication.

Organisation de mémoire

Ce document est organisé en trois chapitres :

Le premier chapitre présente un état de l'art sur le diagnostic des pannes dans les systèmes informatiques et les approches utilisées pour faire le diagnostic.

Le deuxième chapitre est dédié au formalisme DEVS et leurs fonctionnements.

Dans Le troisième chapitre nous donnerons une modélisation par le formalisme DEVS du diagnostic d'un système de réseau de télécommunication et leur simulation sous l'environnement DEVS SUITE.

Chapitre 1 : Etat de l’art

1. Introduction

Les réseaux de télécommunications ont pris un essor important dans le monde. Aujourd'hui, ce sont des systèmes que nous utilisons quotidiennement (utilisation de l'Internet par exemple). Les besoins se multipliant, la complexité de ces systèmes augmente. On demande aux réseaux de télécommunications d'être efficaces, robustes, sûrs et toujours disponibles. Leur gestion devient une activité qui est elle-même de plus en plus complexe. De nouveaux besoins informatiques sont nécessaires afin d'améliorer la gestion par son automatisation. En particulier, l'un des points cruciaux de la gestion de réseau, est le diagnostic des pannes. L'objectif est de détecter les problèmes au plus tôt afin de pouvoir assurer la qualité de service demandée par les utilisateurs (sureté, efficacité, disponibilité). Dans ce chapitre nous présentons un état de l'art sur le diagnostic des pannes dans les systèmes complexes (En particulier les systèmes à évènement discrets) et les modèles de simulation et leurs types, ensuite nous parlons sur la notion de diagnostic dans ces systèmes et leurs objectifs et principes, nous expliquons quelques définitions des notion importantes dans la tâches de diagnostic, ensuite nous présentons une classification des méthode de diagnostic. Enfin nous donnons les formalismes et les approches utilisés pour le diagnostic des pannes dans les systèmes à ED.

2. Concepts de bases sur les systèmes informatiques

2.1. Un système :

Un système est un groupe d'objets qui sont reliés (interconnectées) entre eux avec des interactions régulières ou des interdépendances afin d'accomplir des objectifs prédéfinis.[1]

✓ Exemple: Usine de production de véhicule (Machines, pièces, et les travailleurs fonctionnent conjointement pour produire un véhicule).

✓ Exemple: réseaux informatique (L'utilisateur, les hôtes, les routeurs, câbles,...).

2.2. Classification des systèmes :

2.2.1. Système discret :

Les variables d'état changent à un ensemble discret de points (**exemple** : banque). Il y a Plusieurs modèles mathématiques qui ont été proposés pour l'analyse et la commande de ces systèmes, en particulier les **machines d'états finis** et les **Réseaux de Pétri** (**RdPs**).

2.2.2. Système continu :

Les variables d'état changent continuellement dans le temps ; **Exemple** : Température de l'eau dans un barrage. [1]

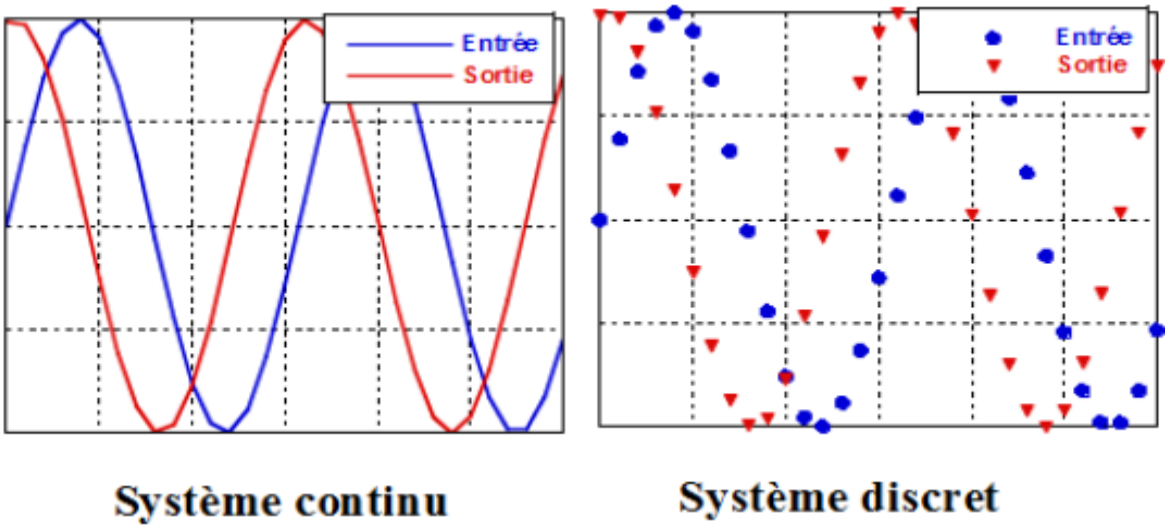

Figure 1: système discret et système continu

2.2.3. Les Systèmes Dynamiques Hybrides :

Les Systèmes Dynamiques Hybrides (**SDH**) couvrent simultanément les deux aspects continu et discret. Ces systèmes évoluent dans le temps et combinent des variables continues et des variables discrètes.

3. La modélisation et la Simulation :

3.1. Un modèle :

Un modèle est une description de la structure d'un système et la représentation graphique comportementale ou fonctionnelle de chacun de ses composants.[1]

3.2. Les types de modèles :

- ✓ **Statique**: Représente un système à un moment donné.
- ✓ **Dynamique**: Représente un système sur un intervalle de temps.
- ✓ **Déterministe**: Les modèles de simulation sans variables aléatoires.
- ✓ **Stochastique**: Les modèles de simulation avec des variables aléatoires.
- ✓ **Discret**: les changements d'état du système se produisent seulement à des points de temps discrets.
- ✓ **Continu**: les changements d'état du système se produisent en permanence.

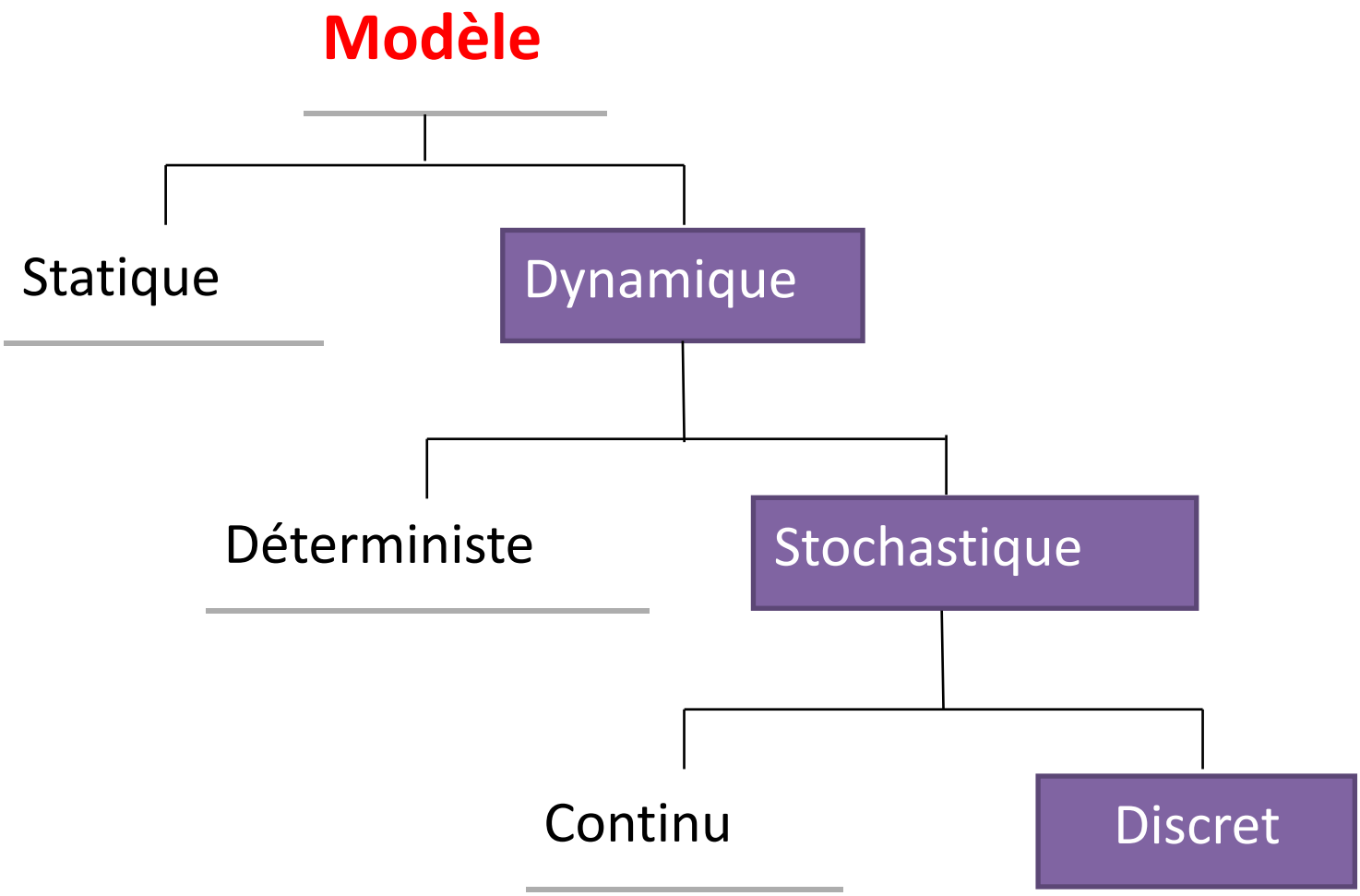

Figure 2: Les types de modèles

4. Diagnostic des pannes dans les systèmes à ED

4.1. Notions importante

4.1.1. Diagnostic:

Le mot « diagnostic » vient du mot grec «**diágnosi** » construit à partir de « **dia** », signifiant « à travers » et de « **gnosi** » signifiant connaissance, discernement .Il s'agit donc d'acquérir la connaissance et de produire une décision à travers les signes observables. Formellement, La fonction diagnostic permet de déterminer les causes et de localiser les éléments défaillants, qui ont entraîné la dégradation du système. En effet, le diagnostic établit

un lien de cause à effet entre un symptôme observé et la défaillance constatée. Cette fonction suit la fonction de détection et inclut les fonctions de localisation et d'identification. [2]

4.1.2. Surveillance :

Un système de surveillance a comme première vocation d'émettre à partir des informations générées par les capteurs, des alarmes dont l'objectif est d'attirer l'attention de l'opérateur de supervision sur l'apparition d'un ou plusieurs événements susceptibles d'affecter le bon fonctionnement de l'installation, comme le dépassement d'un seuil de sécurité au niveau du remplissage d'un réservoir. [2]

Dans le cadre de notre travail, nous considérons la surveillance comme un dispositif passif, dans le sens où ce dispositif n'influence pas le comportement du système à diagnostiquer. Un système de surveillance permet de détecter les défaillances en observant l'évolution du système, puis à les diagnostiquer en localisant les éléments défaillants et enfin identifier les causes premières.

4.1.3. Défaut :

C'est une déviation du système par rapport à son comportement normal, qui ne l'empêche pas de remplir sa fonction. Un défaut est donc une anomalie qui concerne une ou plusieurs propriétés du système, pouvant aboutir à une défaillance et parfois même à une panne.[2]

4.1.4. Dégradation :

Tout état qui se caractérise par une évolution irréversible des caractéristiques d'un système est une dégradation. La dégradation peut être liée à des facteurs directs, tels que l'usage, le temps…,ou à des facteurs indirects, tels que l'humidité, la température… La dégradation peut aboutir à une défaillance, quand les performances du système sont en dessous d'un seuil d'arrêt défini par les spécifications fonctionnelles. [2]

4.1.5. Défaillance :

Une défaillance est une anomalie altérant ou empêchant l'aptitude d'une unité fonctionnelle à accomplir la fonction souhaitée. Une défaillance correspond à un passage d'un état à un autre, par opposition à une panne qui est un état. Par abus de langage, cet état de panne on pourra l'appeler mode de défaillance.

Une défaillance implique l'existence d'un défaut, puisqu'elle aboutit à un écart entre la caractéristique mesurée et la caractéristique de référence. Inversement, un défaut ne conduit pas nécessairement à une défaillance. En effet, le système peut très bien conserver son aptitude à assurer une fonction requise, si les défauts qui l'affectent n'ont pas d'impacts

significatifs sur la mission. Si une défaillance peut conduire à une cessation de l'exécution de la mission principale du système, ce dernier est déclaré en état de panne. Ainsi, la panne est toujours le résultat d'une défaillance. [2]

4.1.6. Panne :

Une panne est un état de non-fonctionnement ou de dysfonctionnement, matériel ou logiciel au sens où une entité est incapable d'accomplir une fonction requise à la suite d'une défaillance.

Deux types de pannes peuvent être distingués :

- **Les pannes permanentes :** une fois la panne est produite, elle nécessite une action de réparation.
- **Les pannes intermittentes :** le système peut retrouver son fonctionnement nominal après l'occurrence de la panne. Une panne intermittente est généralement le résultat d'une dégradation partielle et progressive d'un composant du système, pouvant aboutir à une panne permanente. [2]

4.1.7. Symptôme :

Un symptôme correspond à une ou plusieurs observations qui révèlent d'un dysfonctionnement. Il s'agit d'un effet qui est la conséquence d'un comportement anormal. [2]

4.1.8. Observation :

Est une information obtenue à partir du comportement ou du fonctionnementréel du système. [2]

4.2. Objectif et principe de diagnostic

L'objectif du diagnostic consiste à déterminer à chaque instant le mode de fonctionnement du système par ses manifestations extérieures. Il s'appuie sur une connaissance à priori des modes de fonctionnement et sur une connaissance instantanée matérialisée par une nouvelle observation de l'état du système. Son principe général consiste à confronter les données relevées au cours du fonctionnement réel du système avec la connaissance que l'on a de son fonctionnement normal ou défaillant. Si le mode de fonctionnement identifié est un mode défaillant, le système de diagnostic pourra localiser sa cause. La fonction du diagnostic est donc de chercher une causalité liant le symptôme, la défaillance et son origine.

Un système de diagnostic réalise les tâches suivantes :

4.2.1. Détection:

La détection permet de déterminer si le système physique fonctionne normalement et a pour objectif de signaler la présence d'un défaut en comparant le comportement courant du système avec celui donné pour référence. Malheureusement, il est impossible en pratique d'obtenir un comportement de référence scrupuleusement identique à celui du système en fonctionnement normal. À cause de cette différence, les outils de détection sont généralement de nature statistique (tests d'hypothèses). Un risque d'erreur subsiste donc. Détecter un défaut inexistant peut provoquer un arrêt inutile et générer une perte de confiance des opérateurs (probabilité de fausse détection). A l'opposé, omettre un défaut, qui peut entraîner ultérieurement une panne, est préjudiciable (probabilité de non détection).Il subsiste donc nécessairement un compromis induisant la recherche d'un réglage permettant de minimiser ces probabilités. A ce stade, l'objectif est de déterminer si un événement affectant le système est le signe d'une anomalie et par conséquent de distinguer les évènements qualifiés de normaux (réaction de la commande pour rejeter une perturbation, action d'un opérateur) de ceux qui ne le sont pas (défaut).

4.2.2. Localisation :

La localisation a pour objet la localisation d'un défaut détecté survenant sur les actionneurs, les capteurs d'instrumentation, la commande ou le système commandé en indiquant quel organe ou composant est affecté par celui-ci...

4.2.3. Identification :

L'identification a pour but de caractériser le défaut en durée et en amplitude afin de le classifier par types et degrés de sévérité. Ainsi, il peut servir à assurer le suivi de son évolution, ce qui est fort utile dans le cas d'un changement de comportement lent dû au vieillissement et à l'usure. De plus, ce module peut comprendre une procédure visant à déterminer la cause du défaut, c'est-à-dire son origine. Les tâches de localisation et d'identification constituent le diagnostic de défaut. La plupart des systèmes pratiques comportent seulement ces deux étapes. Aussi, dans beaucoup de cas, au terme de "diagnostic", on associe simplement celui de "**localisation**".

4.2.4. Isolation :

Cette tache permet d'isoler et de réparer le défaut détecté précédemment et retourner le système à l'origine. Par autre définition, **L'isolabilité** est la capacité du système de diagnostic à remonter directement à l'origine du défaut. [2]

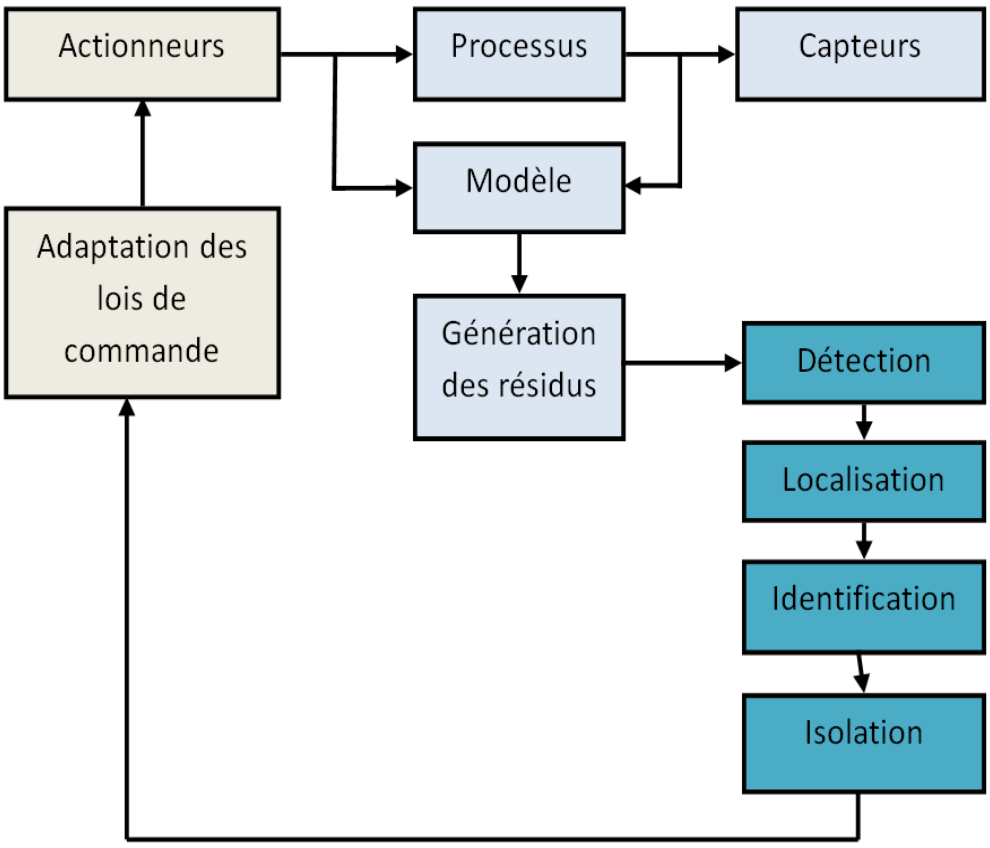

Figure 3:principe de diagnostic

4.3. Classification des méthodes de diagnostic

Les méthodes de diagnostic des défauts utilisées dans le milieu industriel sont très variées. Leur principe général repose sur une comparaison entre les données observées au cours du fonctionnement du système et les connaissances acquises sur son comportement normal et ses comportements de défaillance. Dans cette section, nous présentons une classification des principales méthodes de diagnostic rencontrées dans la littérature. [2]

Cette classification, représentée dans la figure 4, peut être réalisée selon plusieurs critères tels que la nature de l'information disponible (quantitative ou qualitative), la dynamique du système (continu, discret ou hybride), la structure de prise de décision (centralisée, décentralisée ou distribuée). Dans la suite, nous proposons une classification non exhaustive des méthodes de diagnostic selon deux axes : **les approches sans modèles et les approches à base de modèles.**

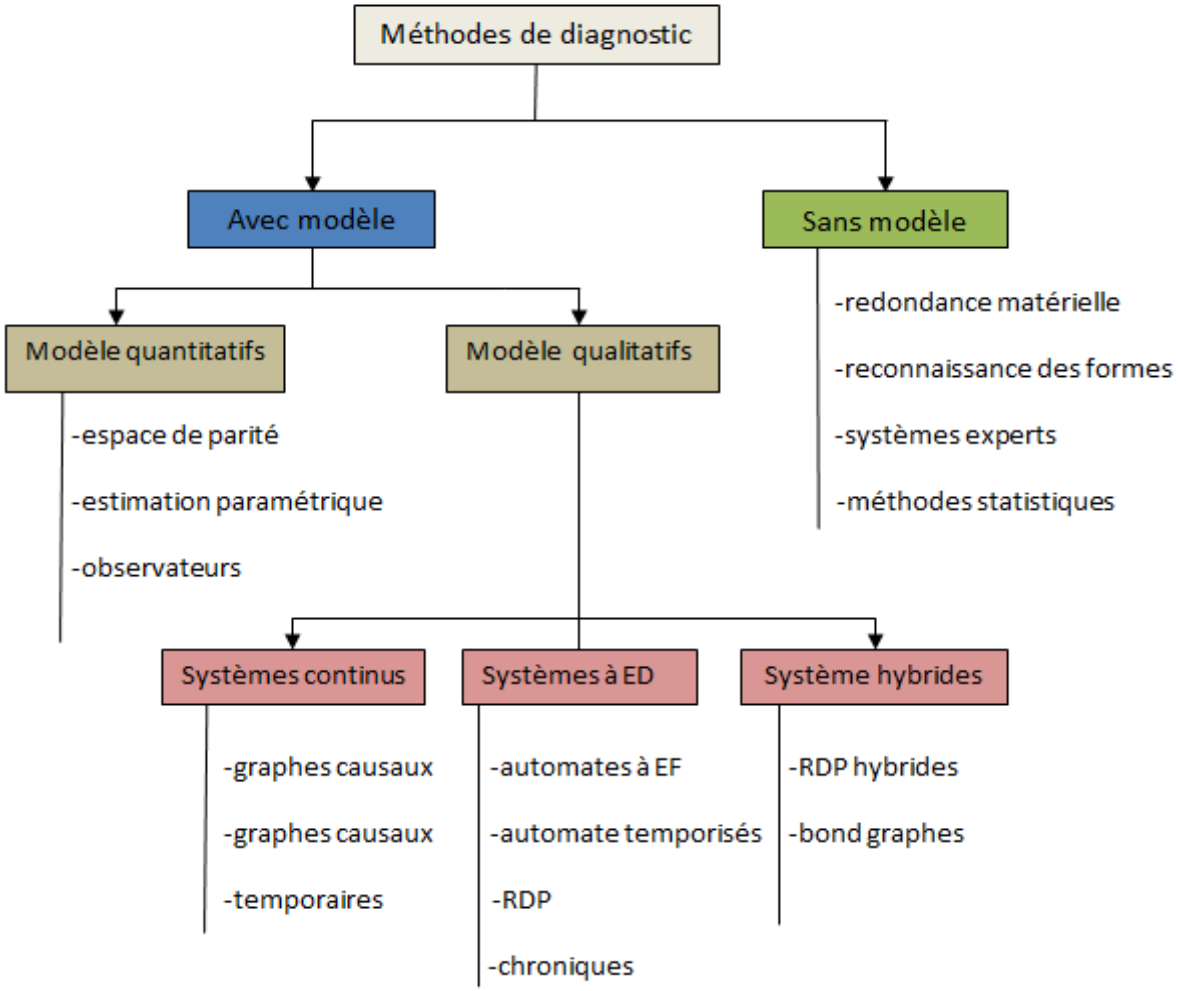

Figure 4:classification des méthodes de diagnostic

4.3.1. Méthodes sans modèles

Dans certaines applications industrielles, il est difficile, voire impossible, d'obtenir le modèle du système. Cette difficulté est justifiée par la complexité accrue ou à de nombreuses reconfigurations intervenants durant le processus de production. En effet, seules les méthodes de surveillance sans modèles sont opérationnelles pour ce type d'applications industrielles. Ces méthodes de diagnostic se basent sur des informations issues d'une expérience préalable, sur des règles heuristiques ou encore sur des exemples de résolution. Parmi ces méthodes, on trouve :

- **La méthode du Seuillage :** les signaux fournis par les capteurs sont comparées avec des valeurs limites constantes ou adaptatives (évoluant en fonction du point de fonctionnement). Un premier niveau indique la présence probable d'un défaut alors qu'un second niveau peut en caractériser la gravité. Le franchissement d'un seuil révèle la présence d'une anomalie.

- **Les méthodes statistiques :** les méthodes statistiques ne supposent que les signaux fournis par les capteurs possèdent certaines propriétés statistiques, sur lesquelles des tests de seuil sont établis. En effet, l'étude de l'évolution de la moyenne ou de la variance d'un signal peut favoriser la mise en évidence d'une anomalie.
- **La reconnaissance des formes :** ces méthodes reposent sur l'utilisation des algorithmes de classification des formes et des mesures (continues ou discrètes). Le fonctionnement d'un système de diagnostic par reconnaissance des formes se déroule en trois phases :

✓ **une phase d'analyse :** qui consiste à déterminer et à réduire l'espace de représentation des données et à définir l'espace de décision permettant de spécifier l'ensemble des classes possibles.

✓ **Une phase de choix d'une méthode de décision :** permettant de définir une règle de décision qui a pour fonction de classer les nouvelles observations dans les différentes classes de l'ensemble d'apprentissage.

✓ **Une phase d'exploitation :** qui détermine, en appliquant la règle de décision, le mode de fonctionnement du système en fonction de chaque nouvelle observation recueillie sur le processus.

- **Les systèmes experts :** les systèmes experts utilisent une information heuristique pour lier les symptômes aux défauts. Ce sont des systèmes à base de règles qui établissent des associations empiriques entre effets et causes. Ces associations sont généralement fondées sur l'expérience de l'expert plutôt que sur une connaissance de la structure et/ou du comportement du système. Leur fonctionnalité est de trouver la cause de ce qui a été observé en parcourant les règles par un raisonnement inductif par chaînage avant ou arrière.

4.3.2. Méthodes à base de modèles

Les approches à base de modèles s'appuient sur des modèles comportementaux explicites du système soumis au diagnostic. Un grand avantage de ces approches par rapport aux approches relationnelles et de traitement de données, réside sur le fait que seule l'information du comportement normal du procédé est prise en compte par l'intermédiaire d'un modèle de référence. La précision du modèle, liée aux besoins de la surveillance et aux critères de performance du diagnostic, définit le choix de l'utilisation de **modèles quantitatifs**, **qualitatifs** ou semi qualitatifs. Selon, les méthodes de diagnostic à base de modèles présentent les avantages suivants :

- La connaissance sur le système est découplée de la connaissance de diagnostic.

- Il s'agit de connaissance de conception plutôt que d'exploitation.
- Les fautes et les symptômes ne doivent pas être anticipés.
- Le coût de développement et de maintenance est moindre.
- Les modèles fournissent un support adéquat pour l'explication (structure du système explicitement représentée). Le principe du diagnostic à base de modèle est de comparer le comportement observé du système et son comportement prédit par un modèle de référence. La figure 5 montre le problème du diagnostic avec une vision comportementale.

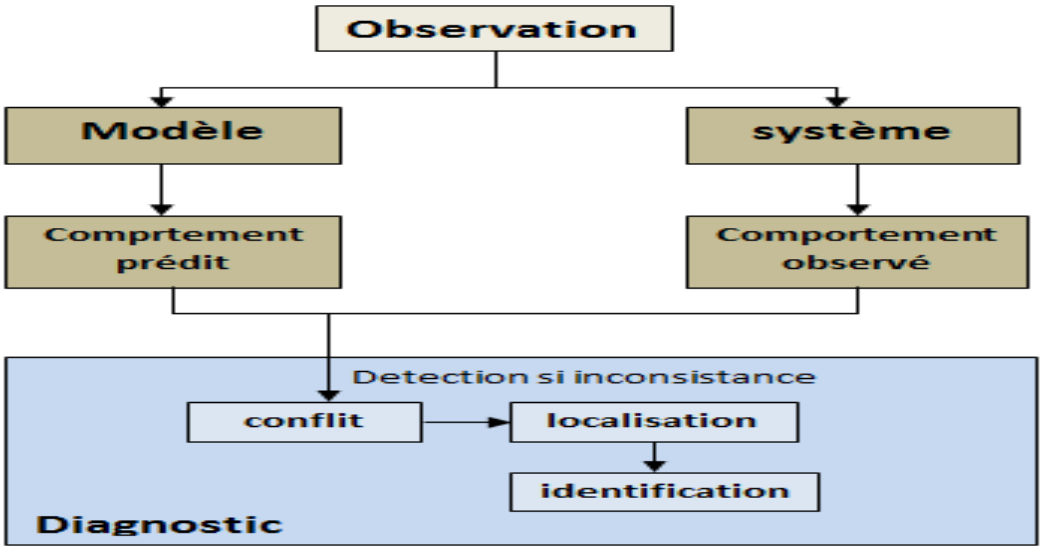

Figure 5: Principe de diagnostic à base de modèle

Le diagnostic à base de modèle est largement présent dans la littérature et a été développé dans les années soixante-dix.

Cette approche est connue sous le nom de **FDI (Fault Detection and Isolation)**, qui, fait intervenir les techniques de génération de résidus, de détection et de localisation. Cette approche est schématisée sur la figure6.

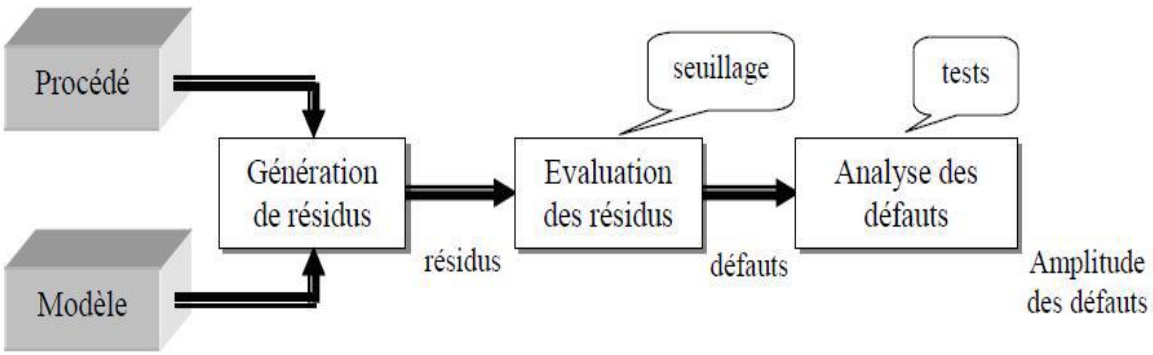

Figure 6: La démarche FDI

4.3.2.1. Méthodes de diagnostic à base de modèles quantitatifs :

Ces méthodes reposent sur l'estimation de l'état, des paramètres ou de l'espace de parité en utilisant des modèles mathématiques du système décrivant le comportement du système. Si l'écart entre ces modèles et les variables du système dépasse un certain seuil, une défaillance est alors détectée. A ce moment, un résidu sera généré et comparé avec toutes les signatures des défauts connues, afin d'isoler et d'identifier la défaillance. Parmi les différentes méthodes de détection et de diagnostic utilisant des modèles mathématiques, nous trouvons principalement l'espace de parité, les observateurs et l'estimation paramétrique.

- **La méthode d'espace de parité :** La méthode de l'espace de parité a été l'une des premières méthodes employées à des fins de FDI. Cette méthode repose sur la vérification de la cohérence (parité) des modèles du procédé avec les mesures issues de capteurs et des entrées connues (consignes,signal de commande, etc...).
- **La méthode à base d'observateurs :** Cette méthode se base sur la reconstruction de la sortie du processus à l'aide d'observateurs, de la comparer avec la sortie mesurée, puis à utiliser l'écart entre ces deux fonctions est utilisé comme résidu.
- **La méthode d'estimation paramétrique :** cette méthode suppose l'existence d'un modèle paramétrique décrivant le comportement du système et que les valeurs de ces paramètres en fonctionnement nominal soient connues. Elle consiste alors à identifier les paramètres caractérisant le fonctionnement réel, à partir de mesures des entrées et des sorties du système. Pour détecter l'apparition de défaillances dans le système, il faut effectuer la comparaison entre les paramètres estimés et les paramètres théoriques.

4.3.2.2. Méthodes de diagnostic à base de modèles qualitatifs :

Les modèles qualitatifs permettent d'abstraire le comportement du procédé avec un certain degré d'abstraction à travers des modèles non plus mathématiques mais des modèles de type symbolique. Ces modèles décrivent d'une manière qualitative l'espace d'état continu du système. Contrairement aux modèles de type numérique, les modèles qualitatifs ne représentent pas la physique du système, mais ils le décrivent en terme de mode de fonctionnement. Les méthodes à base de modèles qualitatifs peuvent être classifiées selon le niveau d'abstraction considéré du système à diagnostiquer, à savoir, les systèmes continus, les systèmes à événements discrets ou les systèmes hybrides dynamiques. Pour les systèmes continus, les approches ont été développées à base de graphes causaux et de graphes causaux

temporels. Le diagnostic des SED est basé sur l'utilisation de modèles discrets tels que les réseaux de Pétri et les automates d'états finis (machine d'états finis),

Les automates temporisées,… Pour les systèmes hybrides, on trouve des méthodes reposant sur des modèles hybrides tels que les automates hybrides à temps discret les bonds graphs, les RdP hybrides…

4.3.3. Critères des méthodes de diagnostic à base de modèles

Les méthodes de diagnostic doivent tenir compte de certains critères qui varient en fonction des besoins en termes de sûreté, des ressources humaines et matérielles disponibles, de l'aspect critique du système surveillé, on a plusieurs critères communs aux méthodes de diagnostic quiont été dégagés. En effet, une méthode de diagnostic doit :

- ✓ fournir un diagnostic fiable (pas de fausses alarmes ni d'alarmes manquantes).
- ✓ être algorithmiquement concevable.
- ✓ être réalisable en temps réel.
- ✓ avoir un temps de réponse raisonnable (faible complexité).
- ✓ permettre un diagnostic rapide des défauts.
- ✓ être facile à mettre en œuvre.
- ✓ nécessite un nombre réduit de capteur.

4.4. Notion de diagnostic centralisé, décentralisé et distribuée

La distribution des composants d'un système, des informations qu'il génère (commandes et compte-rendu des capteurs) peuvent parfois imposer le choix de la structure de prise de décision des méthodes de diagnostic. Ce choix peut être entre une structure centralisée, décentralisée ou distribuée.

➢ **La structure centralisée** consiste à associer un modèle global du procédé avec un seul module de diagnostic. En conséquence, le module de diagnostic collecte les différentes informations du système avant de prendre sa décision finale sur son état de fonctionnement. Cette structure se montre performante en terme de diagnostic. Cependant, elle est exposée au problème de l'explosion combinatoire des modèles utilisés surtout lorsqu'il s'agit de systèmes complexes. En effet, plusieurs approches de diagnostic, reposant sur des structures décentralisées et distribuées, ont été proposées dans la littérature.

➢ **La structure décentralisée** se base sur un modèle global du système à qui sont associés plusieurs modules de diagnostic locaux. Chacun reçoit les informations observables qui lui sont spécifiques et prend une décision locale en se basant sur ses observations locales. Afin de lever le problème d'indécision et permettre aux modules de diagnostic locaux de

diagnostiquer l'ensemble de défauts, un coordinateur doit être utilisé. Il traite les différentes décisions locales, communiquées par les modules locaux, afin de prendre une décision finale.

- **La structure distribuée**, le système est modélisé à travers ses composants par plusieurs modèles locaux. Chacun étant associé à un module de diagnostic local responsable de son composant. Un protocole de communication permet la communication directement entre les différents modules afin de gérer les conflits décisionnels. Chaque module de diagnostic prend sa décision en se basant sur sa propre observation locale et celle communiquée par les autres modules. [3] …

4.5. Diagnostic hors-ligne/en-ligne

- **Le diagnostic hors-ligne** consiste à utiliser un ensemble de comportements et d'observations du système connus à l'avance, pour élaborer son diagnostic. Cette approche,n'impose pas des critères de réactivité de la réponse de diagnostic. En effet, aucune limitation sur le temps de réponse n'est exigée.
- **Le diagnostic en-ligne** consiste à calculer le diagnostic du système pendant qu'il fonctionne,en s'appuyant sur les observations générées. Les approches de diagnostic en-ligne présentent des exigences en terme de réactivité de la réponse. Cela implique un temps de réponse rapide de la part du module de diagnostic. Le diagnostic en-ligne conduit généralement à une difficulté qui concerne la complexité du calcul. En effet, la réponse de diagnostic doit être livrée en temps-réel, le plus rapidement possible. Si le module de diagnostic ne parvient pas à gérer le flux d'observations et inférer le diagnostic suffisamment rapidement, alors le diagnostic en-ligne n'est plus possible. Vu que le nombre d'observations augmente régulièrement, il est nécessaire de disposer d'un diagnostic incrémental, où seul le diagnostic de la date t_{i-1} est pris en compte pour le calcul du diagnostic de la date ti. En effet, la complexité de traitement ne doit pas dépendre du nombre d'observations reçus avant la date t_{i-1}, puisque ce nombre augmente d'une manière non bornée. [3]

4.6. Les approches de diagnostic existantes

Dans les systèmes dynamiques on distingue trois types : les systèmes continus, les systèmes à évènement discrets et les systèmes dynamiques hybrides. Pour les systèmes à événements discrets (SEDs), le 1er travail réalisé pour la notion de **diagnosticabilité** fut introduite par Sampath dans l'année 1995 [4] dans l'approche du "**diagnostiqueur**". Le diagnostiqueur de Sampath est un automate à états finis construit à priori et permettant de faire l'analyse hors ligne de la diagnosticabilité. Le diagnostiqueur est également exécuté en ligne pour diagnostiquer les fautes du système. Afin de réduire la complexité de l'analyse de

la diagnosticabilité pour l'approche diagnostiqueur, les approches **twin-plant et vérificateur** [5,6] ont été proposées par la suite. La complexité est polynomiale avec ces nouvelles structures. Par contre, elles ne résolvent pas le problème d'explosion combinatoire. En particulier, en pratique, beaucoup de temps et de ressources mémoire sont consommés pour analyser la diagnosticabilité de systèmes complexes. Ensuite, des approches basées sur les réseaux de Petri (**RdP**) ont été développées. Les **RdPs** modélisent plus simplement les comportements parallèles et les synchronisations dans un système. Dans [7], une condition suffisante de la diagnosticabilité a été proposée en vérifiant les **T-semiflots** (***T-invariants*** **enanglais**) pour analyser la diagnosticabilité d'un **RdP**vivant. Dans [8], l'analyse de **MBRG** (**Modified Basis Reachability Graph**, en français **graphe d'accessibilité debase**)/**BRD** (**Basis Reachability Diagnoser**, en français **diagnostiqueur d'accessibilité debase**) a été élaborée. **MBRG/BRD** fournissent une manière compacte pour la construction de l'espace d'états afin de mieux combattre l'explosion combinatoire. Dans [9], une approche de **vérification de la *k*-diagnosticabilité** a été proposée, utilisant la programmation linéaire. Dans [10], une structure appelée le **VN** (**Verifier Net**, en français **Vérificateur de RdP**) a été développée pour analyser la diagnosticabilité pour les **RdPs bornés** et non **bornés**, et aussi la ***k*-diagnosticabilité** des **RdPs** non bornés. Cette approche réduit la complexité de l'analyse de **la diagnosticabilité des RdPs bornés**. Dans [11], **l'analyse à la volée** a été proposée, pour l'analyse de la diagnosticabilité et de la ***k*-diagnosticabilité de RdPsvivants et bornés** ; seulement une partie de l'espace d'états est construite à la volée (au lieu de développer la totalité de l'espace d'états à priori), afin de combattre l'explosion combinatoire. [12]

5. Les formalismes utilisées dans le diagnostic à base de modèles qualitatifs pour les systèmes à ED

5.1. Les automates à états fini (machine d'états) :

Un automate à états finis est une machine à états qui permet de décrire les évolutions possibles d'un système à événements discrets. Ainsi, le comportement d'un SED est représenté à travers un ensemble d'événements associé à un ensemble d'états.Un automate est composé d'un ensemble de cercles (qui représentent les états) et un ensemble d'arcs (qui représentent les transitions). Formellement, un automate à états finis est défini par un cuinquplets :

A=(Σ, X, δ, x_0, Fm) avec :

- ✓ Σun ensemble fini de symboles appelé *alphabet* décrivant les événements
- ✓ Xun ensemble fini d'états
- ✓ $\delta: X \times \Sigma \rightarrow X$la fonction de transition entre les états
- ✓ $x_0 \in X$l'état initial
- ✓ Fm ⊆ F est l'ensemble d'états finaux.

Cette définition est relative à un automate *déterministe* car l'état initial est considéré unique (x_0) et la relation de transition à partir de chaque état, et suite à l'occurrence d'un événement donné, définit toujours un état unique (la fonction δ étant définie sur $X \times \Sigma$à valeurs dans X). Un *mot* (ou *séquence*) noté $s$est une suite ordonnée d'événements. Le nombre d'événements qui le forment est noté $|s|$. Il correspond à la taille du mot. On note par εle *mot vide* qui peut représenter l'absence d'événements. Dans la théorie des langages, l'ensemble des mots ou chaines pouvant être formés suivant un alphabet Σet incluant l'élément vide εest noté par Σ^*où « $*$» représente **l'*Etoile de Kleene*.** Le langage (A) engendré par un automate est un sous ensemble de Σ^*. Il correspond à l'ensemble des mots réalisables par cet automate (c.-à-d. des mots auxquels correspondent des chemins dans A). [13]

Exemple : La Figure 7 représente un exemple d'automate à états finis déterministe. Il est composé de l'ensemble d'événements $\Sigma = \{a, b, c, d, e, f\}$ et de l'ensemble des états $X = \{x_0, x_1, x_2, x_3\}$. L'état initial est donné par x_0(désigné par une flèche entrante sur le graphe). Les arcs sont une représentation graphique de la fonction de transition. Si on considère la séquence d'événements $abcb$, la notation (x_0, $abcb$) = $x_2$signifie que le franchissement de cette séquence à partir de l'état initial $x_0$fait que le modèle évolue de l'état $x_0$vers l'état x_2. Le langage généré par l'automate Aest donné par (A)= $\{\varepsilon, a, ab, abc, abd, abe, abcb, \dots\}$.

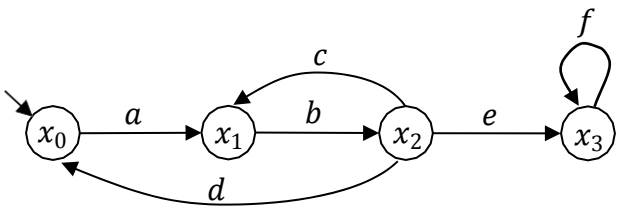

Figure 7: Exemple d'un automate à états finis déterministe.

5.2. Les automates temporisées :

Les automates temporisés ont été mis au point pour l'analyse des systèmes temporisés. Ils sont généralement adaptés pour l'analyse et la vérification des systèmes temps réels et à

fortes contraintes temporelles. Ces modèles peuvent être décrits comme des automates à états finis avec des contraintes temporelles. Celles-ci sont représentées par des inéquations mettant en jeu plusieurs horloges à valeurs réelles. L'automate temporisé de la Figure 8 est caractérisé par deux horloges h et g. Chaque horloge peut être indépendamment réinitialisée après un événement donné (h :=0 sur l'exemple de la Figure 8 signifie que l'horloge h est remise à zéro lorsque l'événement a se produit). Une horloge mesure donc le temps écoulé depuis sa dernière mise à zéro. Les contraintes sont de deux types : celles liées aux transitions et appelées *gardes* (reportées près des arcs sur l'exemple de la Figure 8) et celles associées aux états et appelées *invariants* (entre crochets sur l'exemple de la Figure 8). Par exemple la garde h=3 signifie que la transition peut avoir lieu seulement lorsque l'horloge h est égale à 3. L'invariant h≤5 signifie que l'horloge h peut avoir une valeur inférieure ou égale à 5 lorsque le système est dans l'état associé à cet invariant et le modèle ne peut plus rester dans cet état lorsque h>5. Formellement, un automate temporisé AT est défini comme suit :

AT=<Σ,,$\mathcal{H}$,Inv,δT,x0> avec :

- ✓ Σ est l'ensemble des événements (alphabet).
- ✓ X un ensemble fini d'états.
- ✓ $\mathcal{H}$ est l'ensemble des horloges à valeurs réelles positives.
- ✓ δT:X×$\mathcal{C}(\mathcal{H})$ ×Σ×2X×X est un ensemble de transitions.(xi,c,α,γ,xj) correspond à une transition entre l'état xi et xj, gardée par la contrainte c∈$\mathcal{C}(\mathcal{H})$ ($\mathcal{C}(\mathcal{H})$ étant l'ensemble des contraintes) suite à l'occurrence de l'événement α avec une remise à zéro des variables spécifiées par γ.
- ✓ Inv:X→$\mathcal{C}(\mathcal{H})$ associe un invariant à chaque état.
- ✓ x0: est l'état initial.

À l'état initial, l'automate temporisé de la Figure 8 se trouve dans x1 et les deux horloges sont à zéro. Quand l'événement a se produit à h=3, l'évolution de x1 vers x2 réinitialise l'horloge h. À l'état x2, la valeur de l'horloge g est donnée par h+3 car g n'a pas été réinitialisée. Pour que l'automate transite de x2 vers x1, l'événement b doit se produire lorsque la valeur des deux horloges dépasse la valeur 4 et que g≤8. [13]

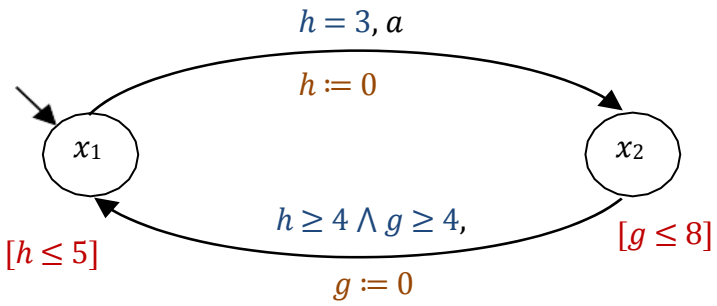

Figure 8: Exemple d'un automate temporisé

5.3. Les automates probabilistes :

Les automates probabilistes sont une extension des automates finis où une probabilité d'occurrence est associée à chaque événement. Ce formalisme a été introduit pour analyser les SED probabilistes dans lesquels les fréquences d'exécution des actions sont connues. Dans le cas d'un automate probabiliste, une *matrice stochastique* peut être utilisée pour représenter les probabilités des transitions. Le langage engendré par un automate probabiliste est appelé **langage stochastique**.

Un automate probabiliste est formellement défini comme suit :

Un automate à états finis probabiliste *AP* est un quadruplet $AP=<\Sigma, X, \delta, 0>$ avec :

✓ Σ est l'ensemble des événements (alphabet).

✓ X un ensemble fini d'états.

✓ $\delta:\times\Sigma\times X\rightarrow[0,1]$ est la fonction des probabilités de transition telle que $\delta P(xi,\alpha,xj)$ détermine la probabilité de transiter de l'état xi vers l'état xj suite à l'occurrence de l'événement α.

✓ $\pi 0:X\rightarrow[0,1]$ est la distribution initiale des probabilités ($\pi 0(xi)$ est la probabilité que xi soit l'état initial).

La probabilité des transitions vérifie $\forall xi\in:\Sigma\Sigma\ (xi,,)xj\in X\alpha\in\Sigma=1$. Si l'on associe des probabilités de transition à l'automate de la Figure 9, on obtient un automate probabiliste représenté sur la Figure 9. Si l'on considère que l'état initial est donné par $x0$, alors la distribution initiale de probabilité sera donnée par $\pi 0(x0)=1$ et $\pi 0(xi)=0$ pour $i=1,\ldots,3$. $(x2,,0)=0.5$ signifie que, lorsque le système se trouve dans l'état $x2$, sa probabilité de transiter vers l'état $x0$ suite à l'occurrence de l'événement d vaut 0.5. [13]

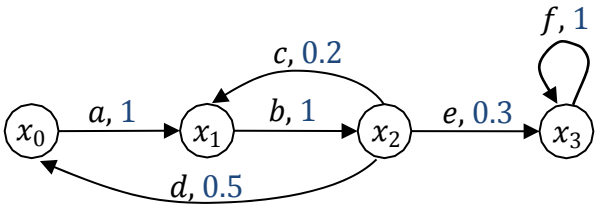

Figure 9:Exemple d'un automate probabiliste.

La modélisation des systèmes complexes en utilisant les automates à états finis et leurs extensions est souvent confrontée au problème de l'explosion combinatoire. C'est un problème récurrent dû à la granularité de la modélisation et qui résulte du nombre d'états qui augmente de façon vertigineuse en fonction de la taille et de la complexité du système.

5.4. Les réseaux de Pétri :

(aussi connu comme un réseau de **Place/Transition** ou réseau de **P/T**) est un modèle mathématique servant à représenter divers systèmes (informatiques, industriels…) travaillant sur des variables discrètes.

Les réseaux de Pétri sont apparus en 1962, dans la thèse de doctorat de **Carl Adam Petri**. **Les Rdps** sont des outils graphiques et mathématiques permettant de modéliser et de vérifier le comportement dynamique des systèmes à événements discrets comme les systèmes manufacturiers, les systèmes de télécommunications, les réseaux de transport. [13]

Définition : Un **RdP** noté Gest donné par G=($\mathcal{P}$, $\mathcal{T}$, W_{PR}, W_{PO}), avec :

- ✓ $\mathcal{P}$ = {P_1, …, P_n} un ensemble de $n$places.
- ✓ $\mathcal{T}$ = {T_1, …, T_q} un ensemble de $q$transitions.
- ✓ W_{PR}∈$(\mathbb{N})^{\times q}$($\mathbb{N}$l ensemble des entiers naturels) la matrice d'incidence avant.
- ✓ W_{PO}∈$(\mathbb{N})^{\times q}$la matrice d'incidence arrière.

La matrice d'incidence W∈$(\mathbb{N})^{\times q}$est donnée par l'équation suivante :

$$W = W_{PO} - W_{PR}$$

Exemple : La figure 10 représente un réseau de pétri simple qui est composée de 4 place P=(p1,p2,p3,p4) et de 5 transition T=(t1,t2,t3,t4,t5).

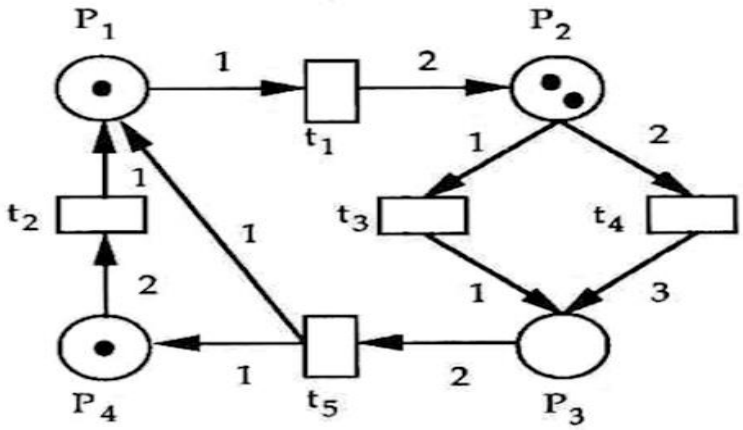

Figure 10: un réseau de Pétri simple

6. Conclusion

Dans ce chapitre, nous avons présenté un état de l'art sur le diagnostic des pannes dans les systèmes et en particulier les systèmes à événements discrets, ainsi nous avons expliqué des concepts de base sur les systèmes et les modèles informatique, leurs définitions et types. Ensuite,Nous avons donné une aperçu sur la notion de diagnostic, des définitions des notions importantes, ainsi l'objectif et le principe de diagnostic .Par la suite nous avons donné une classification des méthodes de diagnostic selon deux axes : les approches avec modèle et les approches sans modèle. Enfin, nous avons donné les formalismes largement utilisés pour l'analyse et le diagnostic des SED, à savoir les automates à états finis, les automates temporisée, les automates probabiliste et les réseaux de Pétri. Dans le prochain chapitre, nous nous donnerons un aperçu général sur le formalisme **DEVS** et son fonctionnement de manière détaillé.

Chapitre 2 : Le formalisme DEVS

1. Introduction

Le formalisme DEVS (Discrete Event Specification) proposé en 1976 par BP Zeigler, est un formalisme expressif, abstrait, universel et indépendant de la mise en œuvre. Il intègre intrinsèquement la dimension temporelle et manipule conjointement les deux concepts d'état et événement. Enfin, il introduit la possibilité d'inclure l'évolution autonome du modèle à travers la durée de vie des états.

DEVS a été utilisé pour modéliser le comportement de systèmes complexes dans divers domaines, y compris l'industrie, médical, défense, etc. Sur la base de ce formalisme, la recherche a abouti à des extensions, des approches, etc.

Dans ce chapitre nous présenterons le formalisme DEVS ainsi que la modélisation et simulation des systèmes complexes par ce dernier, les types des modèles définis par ce formalisme, ainsi que les différents environnements de modélisation et simulation DEVS et enfin les avantages de ce formalisme.

2. La modélisation DEVS

2.1. Définition

DEVS est un formalisme modulaire, abstrait et indépendant de toute implémentation. Il permet la modélisation des systèmes causaux et déterministes.

Un modèle atomique DEVS est basé sur un temps continu, des entrées, des sorties, des états et des fonctions (sortie, transition, durée de vie des états). Des modèles plus complexes sont construits en connectant plusieurs modèles atomiques de façon hiérarchique. Les interactions sont assurées à travers les ports d'entrée et de sortie des modèles, ce qui favorise la modularité.[14]

Le formalisme DEVS définit deux catégories de modèles : les modèles atomiques et les modèles couplés. Les modèles atomiques sont chargés de représenter le(s) comportement(s) du système.

Les modèles couplés sont définis par un ensemble de sous-modèles (atomiques et/ou couplés) et traduisent la structure interne des sous-parties du système grâce à la définition de couplages entre les sous-modèles.

2.2. Le modèle atomique

Le formalisme de base DEVS dit “classique” considère un modèle atomique comme un concept mathématique basé sur le temps, un ensemble de valeur caractérisant tous les stimuli possibles en entrée et en sortie du système ainsi que deux fonctions permettant de déterminer la réponse comportementale à ces stimuli,

Dans le formalisme DEVS dit “classique avec ports” la notion de couple (port, valeur) est introduite pour chaque port (entrée ou sortie) d’un modèle atomique. Les spécifications classiques d’un modèle atomique MA avec ports sont les suivantes :

$MA = (X , Y , S , \delta int , \delta ext , \lambda , ta)$

$X = \{(p,v) | p \in Ports\_entree, v \in Xp\}$ est la liste des ports et des valeurs d’entrées,

$Y = \{(p,v) | p \in Ports\_sortie, v \in Xp\}$ est la liste des ports et des valeurs de sorties,

S est l’ensemble des variables d’états,

$\delta int : S \rightarrow S$ est la fonction de transition interne,

$\delta ext: Q \times X \rightarrow S$ est la fonction de transition externe où,

$Q = \{(s,e) | s \in S, 0 \leq e \leq ta(s)\}$ est l’ensemble des états,

e est le temps écoulé depuis la dernière transition.

$\lambda : S \rightarrow Y$ est la fonction de sortie,

$ta : S \rightarrow R+$ est le temps de vie de l’état S (réels positifs de 0 à ∞).

Les modèles réagissent à deux types d’événements : les événements externes et les événements internes. Les événements externes proviennent d’un autre modèle, déclenchent la fonction de transition externe et mettent à jour le temps de vie du composant. Les événements internes correspondent à des changements d’états du modèle, déclenchent les fonctions de transitions internes et de sorties, le modèle calcul ensuite grâce à la fonction ta la date du prochain événement interne.

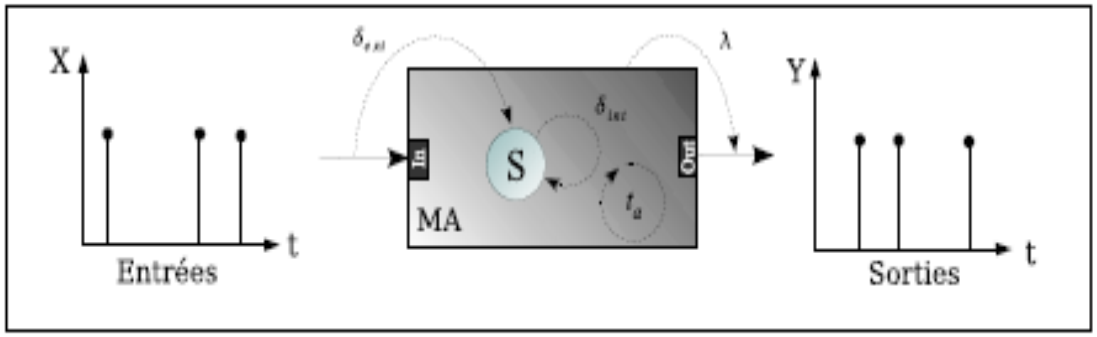

Figure 11:Modèle atomique en action

L'interprétation de ces éléments est illustrée sur la figure11 . A chaque instant le système (modèle atomique) est dans un état s. Si aucun événement externe n'intervient, le système restera dans cet état pendant un temps donné par la fonction ta(s). Lorsque le temps de vie du modèle est expiré, i.e. lorsqu'il s'est écoulé e = ta(s) le système active sa fonction de sortie λ(s) et change d'état grâce à l'exécution de la fonction de transition interne δint (s). Si un événement externe x ∈ X intervient avant que le temps ne soit expiré, i.e. quand le système est dans l'état total (s,e) avec e≤ta(s), le système change d'état grâce à l'exécution de la fonction de transition externe δext (s,e,x). Dans les deux cas, le système est alors dans un nouvel état s'avec un nouveau temps restant ta(s') et ainsi de suite.

Le temps de vie du composant peut être égal à zéro ou à l'infini. Dans le premier cas, la durée de l'état s est tellement courte qu'aucun événement externe ne peut intervenir avant l'arrivée du prochain changement d'état. Nous disons de s qu'il est dans un état transitoire. Dans le second cas, le système restera dans l'état s indéfiniment si aucun événement externe ne vient l'interrompre. Nous disons dans ce cas que s est un état passif.

Notons que dans DEVS classique avec ports, un port p reçoit une seule valeur v d'un événement externe. Avec le formalisme DEVS parallèle, un modèle DEVS peut recevoir sur un port d'entrée donné plusieurs valeurs simultanées. La gestion des événements simultanés est prise en compte au sein d'une fonction de conflit δcon dictant un ordre de priorité entre ces événements. [15]

2.3. Le modèle couplé

Le formalisme DEVS utilise la notion de hiérarchie de description qui permet la construction de modèles dits "couplés" à partir d'un ensemble de sous-modèles et de trois relations de couplage avec ces sous-modèles. Un modèle couplé est constitué de sous-modèles qui peuvent être atomiques ou couplés et il possède les trois relations de couplage suivantes :

- ✓ **IC (Internal Coupling):** une relation de couplage interne pour le couplage entre les ports des sous-modèles qui composent le modèle couplé.
- ✓ **EIC (External Input Coupling):** une relation de couplage des entrées externes pour le couplage entre les ports d'entrées du modèle couplé et les ports d'entrées des sous-modèles.
- ✓ **EOC (External Output Coupling):** une relation de couplage des sorties externes pour le couplage entre les ports de sorties du modèle couplé et les ports de sorties des sous-modèles.

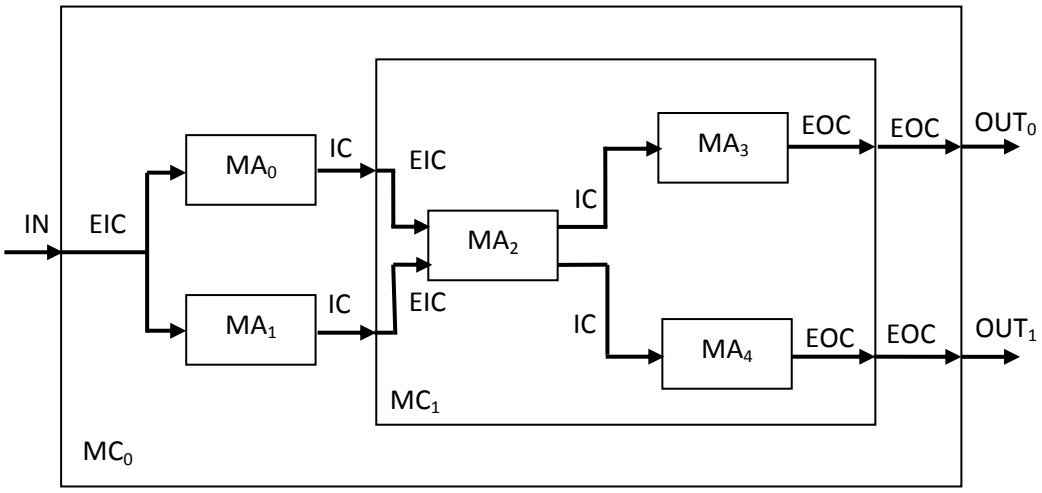

Figure 12: Hiérarchie de modélisation DEVS

La figure 12 montre un exemple de hiérarchie entre les sous-modèles d'un système. Le modèle couplé MC0 modélise au plus haut niveau le système étudié. Il possède deux ports de sortie OUT0 et OUT1, un port d'entrée IN et il contient les deux modèles atomiques MA0 et MA1 ainsi qu'un modèle couplé supplémentaire MC1. Les modèles atomiques MA0 et MA1 sont reliés par une relation de couplage d'entrées externe au modèle couplé MC0, et par une relation de couplage interne au modèle couplé MC1.

Dans le cas du formalisme DEVS classique avec port les spécifications d'un modèle couplé sont les suivantes :

MC =< X,Y,D,{Mi},{Ii},{Zi, j} ,Select >

- ✓ X = {(p,v) | p ∈ ports_entrées, v ∈ Xp} est la liste des ports et des valeurs d'entrées.
- ✓ Y = {(p,v| p ∈ ports_sorties, v ∈ Yp)} est la liste des ports et des valeurs de sorties.
- ✓ D est la liste des composants constituant le modèle couplé.
- ✓ Mi =< Xi,Yi,Si, δext,i, δint,i, λi, ta,i > est un modèle atomique,
- ✓ Pour chaque modèle i ∈ D∪{MC}, Ii est l'ensemble des modèles qui influencent i,
- ✓ Zi, j est la fonction de translation des sorties de i vers j telle que :
- ✓ ZMC, j : XMC → Xj est la fonction de couplage des entrées externes,
- ✓ Zi,MC : Yi → YMC est la fonction de couplage des sorties externes,
- ✓ Zi, j : Yi → Xj est la fonction de couplage interne.
- ✓ Select est la fonction de sélection qui donne la priorité entre les éléments deux à deux, afin d'éviter tout conflit.

La structure d'un modèle couplé doit répondre à des contraintes telles que, ∀ i ∈ D:

1. Mi =< Xi,Yi,Si, δext,i, δint,i, λi, ta,i > est un modèle atomique,

2. Une seule fonction Zi, j contient l'ensemble des informations sur les couplages du modèle couplé,

3. Ii est un sous-ensemble de D∪{MC} et i ∉ Ii.

La cohérence et la conservation des comportements du système entre ces niveaux hiérarchiques est résumée par la propriété dite" fermeture sous couplage". Cette propriété assure qu'un modèle couplé, représenté par le couplage d'un ensemble de sous-modèles plus détaillé, est équivalent à un modèle atomique.

3. La simulation DEVS

L'une des propriétés importantes du formalisme DEVS est qu'il fournit automatiquement un simulateur pour chacun des modèles. DEVS établit une distinction entre la modélisation et la simulation d'un système de manière à ce que n'importe quel modèle DEVS puisse être simulé sans qu'il ne soit nécessaire d'implémenter un simulateur spécifique. Chaque modèle atomique est associé à un simulateur chargé de gérer le comportement du composant et chaque modèle couplé est associé à un coordinateur chargé de la synchronisation temporelle des composants sous-jacents. L'ensemble de ces modèles est géré par un coordinateur spécifique appelé "Root".

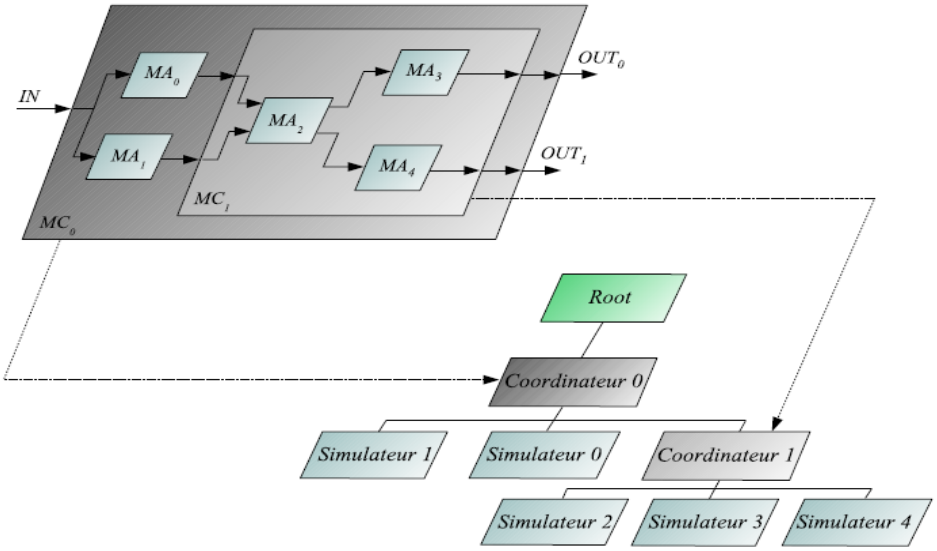

Figure 13: Arbre hiérarchique de simulation DEVS

La figure 13 montre la structure d'un simulateur associé à un modèle DEVS quelconque. La hiérarchie du simulateur DEVS est construite sur une arborescence constituée de simulateurs et de coordinateurs. Les deux coordinateurs "Coordinateur 0" et "Coordinateur 1" sont associés aux modèles couplés MC0 et MC1. Le "Coordinateur 0" est chargé de gérer le "Coordinateur 1" ainsi que les simulateurs "simulateur 0" et "Simulateur 1" associés aux deux modèles atomiques MA0 et MA1. De la même manière, comme le modèle couplé MC1 encapsule les trois modèles atomiques MA2, MA3 et MA4, le coordinateur "Coordinateur 1" associé gère les trois simulateurs "Simulateur 2", "Simulateur 3" et "Simulateur 4". Le

coordinateur ”Root” est chargé de coordonner la totalité des composants de l’arbre de simulation.

L’ordre d’exécution des fonctions comportementales (δint ,δext ,λ ,ta) d’un modèle atomique DEVS de la figure II.4 sous-entend la présence d’un mécanisme de simulation. Ce mécanisme est géré par le composant simulateur qui, comme le coordinateur, peut recevoir ou émettre 4 types de messages :

Le message d’initialisation (i,t) permet à tous les acteurs d’effectuer une synchronisation temporelle initiale.

Le message de transition interne (*,t) permet la gestion d’un événement interne avec l’exécution de la fonction δint .

Le message de sortie (y,t) permet le transport des sorties (données par l) aux éléments parents et résulte de la réception d’un message (*,t).

Le message de transition externe (x,t) permet la gestion d’un événement externe avec l’exécution de la fonction δext.

4. Exemple d'un modèle DEVS

Sur la Figure 14, M1 est un modèle DEVS atomique. Sur ce modèle, les deux types de transitions existantes entre états sont présentés.

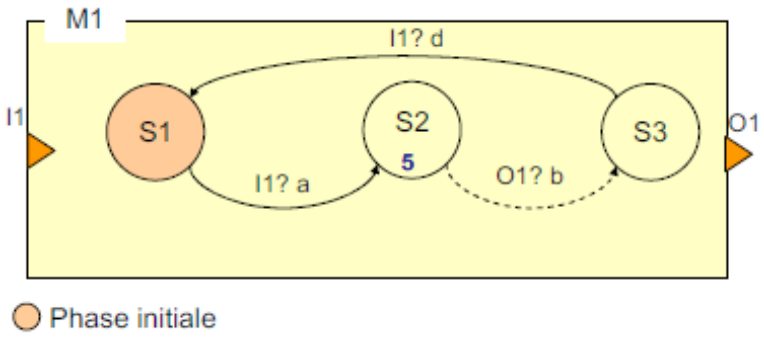

Figure 14: Exemple d'un DEVS atomique

- M1= <X, S, Y, , δext,δint ,λ, ta>
- Où :
- X = {a,d},

- S = {S1, S2, S3},
- Y = {b},
- δint (S2) = S3
- δext : (S1, e, a) = S2

  (S3, e, d) = S1

- γ(S2) = b
- ta (S2) = 5

La dynamique du modèle peut être décrite de la manière suivante. Le modèle est dans sa phase initiale S1. Lorsque le modèle reçoit l'événement "a" sur son port "I1", il passe dans la phase "S2". Au bout de 5 unités de temps, le modèle évolue dans la phase "S3", en émettant l'événement "b". Le modèle de la Figure 15 est un modèle DEVS couplé.

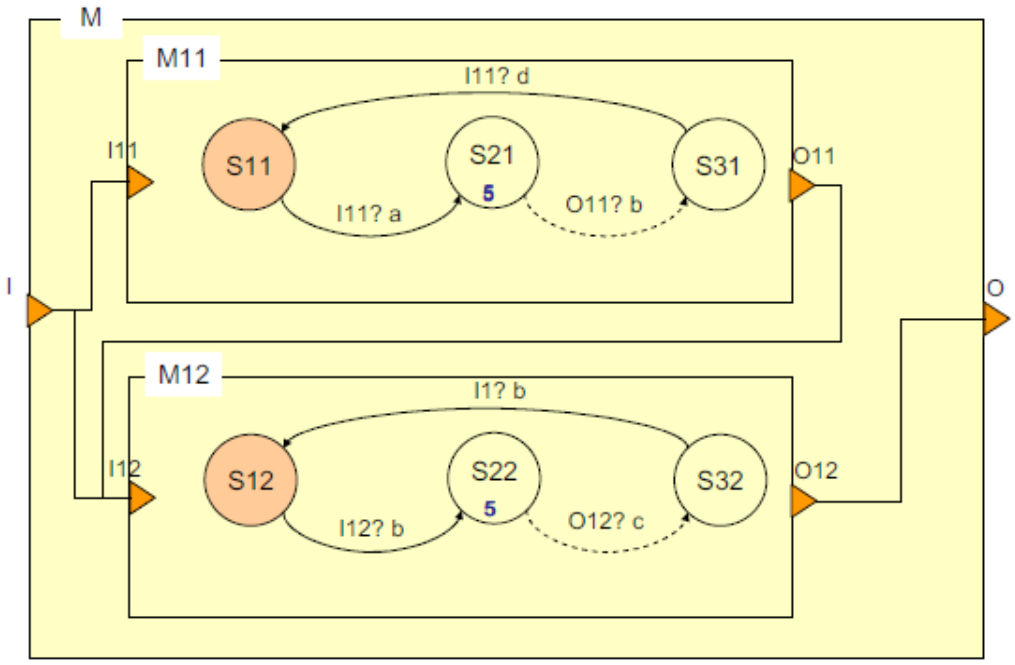

Figure 15: Exemple d'un DEVS couplé

M= <X, Y, D, {Md|dD}, EIC, EOC, IC, Select>

Où :

- X= {(I, a), (I, b), (I, d)},
- Y= {(O, c)},
- D = {M11, M12}, où M11 et M12 sont deux modèles DEVS atomiques,

- EIC= {((M, I), (M11, I11)), ((M, I), (M12, I12))},
- EOC= {((M12, O12), (M, O))},
- IC= {((M11, O11), (M12, O12))},
- Select : (M11, M12) -> M11.

Les deux modèles M11 et M12 sont semblables au modèle atomique décrit ci-dessus.

M11 et M12 sont initialement dans leur phase respective S11 et S12. Lorsque le modèle couplé M reçoit un événement sur son port "I", il l'envoie à M11 et à M12, sur les ports respectifs I11 et I12.

Si l'événement reçu par "I" est "a", M11 évolue dans la phase S21, et M12 ne change pas de phase. Au bout de 5 unités de temps, M11 émet l'événement "b" par son port de sortie O11, qui le transmet à M12 sur le port I12. Si M12 est encore dans sa phase initiale S12, il évolue dans S22, et émet l'événement "c" par son port de sortie "O12", sur le port de sortie "O" du modèle M.

Si l'événement reçu par "I" est "b", M12 évolue dans la phase S22, et M11 ne change pas de phase. Au bout de 5 unités de temps, M12 émet l'événement "c" par son port de sortie O12, qui le transmet à M sur le port O.

5. Environnements de modélisation et simulation DEVS

Le formalisme DEVS est fréquemment adapté ou spécialisé lorsqu'il est placé dans le contexte spécifique d'un domaine d'application. Le formalisme étant souvent trop abstrait, il est nécessaire d'enrichir sa syntaxe pour permettre de cadrer et simplifier la spécification de types de modèles particuliers. Pour cela un grand nombre de techniques dérivées sont l'objet de recherche. Parmi ces techniques, certaines se révèlent utiles pour le domaine d'étude des systèmes complexes.

Il existe plusieurs simulateurs DEVS. Chaque simulateur DEVS à une certaine particularité qui le rend plus performant dans certains domaines, tout comme les spécialisations de DEVS (Parallèle DEVS, Dynamique DEVS, Cell-DEVS, etc.). L'existence d'un simulateur DEVS abstrait donne plus de liberté aux chercheurs pour l'amélioration des performances des environnements de modélisation et simulation DEVS. Un simulateur ne met pas en œuvre le formalisme si les résultats de simulation ne sont pas conformes aux résultats

du simulateur abstrait. Dans la suite de cette partie, nous détaillerons les spécificités de quelques simulateurs DEVS.

✓ **ADEVS (A Discrete Event Simulation)** est un simulateur principalement centré sur la performance et la légèreté. Le noyau de simulation est codé en C ++ et réutilise intensivement les modèles. Cet outil a été développé par l'équipe de Zeigler à l'université d'Arizona. Il offre un ensemble de librairies extensibles pour l'implémentation des modèles DEVS spécifiés.

✓ **CD ++** est un outil développé en C ++, qui permet la définition de modèles DEVS atomiques, Cell-DEVS permet également la définition de modèles DEVS et Cell-DEVS couplés en utilisant un langage de spécification de haut niveau. Différentes versions sont proposées incluant des simulateurs à temps réel, parallèles ou centralisés. Cet outil a été développé par Gabriel Wainer et son équipe, en partenariat avec les universités de Carleton (Canada), et Buenos Aires (Argentine).

✓ **DEVS-Suite** est développé en Java. Il est le successeur du DEVSJava. DEVS-Suite contient un Viewer de simulation supplémentaire et un environnement de suivi. Comme il est codé en Java, il présente une portabilité beaucoup plus élevée à la fois pour le simulateur et les modèles générés.

✓ MS4 Me est basé sur la plate-forme Eclipse. Cet outil est commercial et ses sources sont fermées. Ce simulateur est codé en Java et contient de nombreuses fonctionnalités, comme une interface graphique, IDE, Viewer de simulation, etc.

✓ **JDEVS** permet d'utiliser des modèles à événements discrets, dans un but général. Il peut être utilisé pour des représentations objet, la modélisation basée sur les composants, le développement de modèles de simulation connectés avec des systèmes d'informations géographiques (SIG), ou des systèmes collaboratifs. Il intègre un outil de modélisation et d'exécution visuel. Il a été développé par l'équipe de Jean-Baptiste Filippi de l'Université Pascal Paoli à Corte.

✓ **Python DEVS** est un simulateur codé en Python tout comme les modèles. Il a une petite base de code et il est extrêmement léger. Cet outil a été développé au sein du **Modelling, Simulation and Design Lab (MSDL)** de l'Université McGill à Montréal, Canada

✓ **L'environnement VLE (Virtual Laboratory Environnement),** développé au sein du Laboratoire d'Informatique du Littoral (LIL) de l'Université du Littoral, offre un gain majeur en temps pour le développement des modèles à moindre erreur de codage. D'autres fonctionnalités font partie de cet environnement comme la définition des schémas d'expérimentation pour le chargement des données de la simulation, la sauvegarde des

modèles en **XML**, etc. Il permet de spécifier des systèmes complexes en termes d'objets et d'agents réactifs, de simuler la dynamique du système et d'analyser les résultats des simulations. Les librairies fournies permettent également le développement de programmes personnalisés.[16,17]

6. Avantages de l'utilisation de DEVS

✓ DEVS permet la spécification de modèles à événements discrets. Ses principales caractéristiques sont les suivantes :

✓ DEVS est un formalisme abstrait indépendant de l'implémentation et par conséquent des environnements de simulation.

✓ Il offre une vision modulaire et hiérarchique des systèmes dynamiques.

✓ La fonction de transition interne, formalise et permet le comportement autonome du modèle. Les modèles peuvent ainsi évoluer et émettre des événements lorsqu'aucun événement externe n'est programmé pendant une certaine durée (qui correspond à la durée de vie de l'état dans lequel se trouve le modèle).

✓ DEVS est fermé sous composition. Cela signifie que toute composition obtenue par couplage de composants spécifiés par le formalisme est elle-même spécifiée par le formalisme. La construction hiérarchique des modèles par l'application récursive des procédures de couplage est alors facilitée.

✓ L'interprétation, dans le monde réel, des concepts d'abstraction et de modélisation est explicite.

✓ Il garantit la cohérence de la représentation construite, et par conséquent il offre un modèle implémentable pour simuler des modèles élaborés.

✓ Il est possible de procéder à des vérifications formelles de modèles DEVS, ce qui est une aide précieuse lors de la conception du modèle.

7. Conclusion

Nous avons présenté dans ce chapitre le formalisme **DEVS** et leur modèles : atomique, couplée. Par la suite nous avons présenté les différents environnements de modélisation et de simulation du formalisme DEVS existants à savoir **DEVS-Suite, JDEVS, Python DEVS** etc… et enfin nous avons cité les avantages de ce formalisme.

Chapitre 3 : Modélisation et simulation

1. Introduction

Dans les deux chapitres précédents nous avons donné un aperçu général sur le diagnostic des pannes dans les systèmes à évènement discrets ainsi que le formalisme DEVS et son fonctionnement.

Dans ce chapitre nous allons présenter la modélisation et simulation du diagnostic des pannes dans les systèmes à évènement discrets des réseaux de télécommunication par le formalisme DEVS ainsi que l'implémentation du modèle proposé sur environnement de modélisation et simulation **DEVS SUITE.**

2. Présentation du système de réseau de télécommunication

Les opérateurs de télécommunication font de plus en plus d'efforts pour fournir des services de très bonne qualité à leurs abonnés. Les réseaux de télécommunication doivent être fiables et robustes pour garantir également la haute disponibilité de ces services. La gestion des réseaux est de ce fait devenue un problème central pour les opérateurs de télécommunication qui développent des travaux de recherches afin d'automatiser autant que possible la gestion de pannes. Le diagnostic de pannes est un aspect central de la gestion et de l'administration d'un réseau. Le but principal du diagnostic est de retrouver aussi rapidement que possible les causes racines des anomalies observées qui interrompent ou dégradent la qualité de service fournie aux abonnés.

Dans cette section nous exposons un système appelé Toynet. Ce système est composé de trois commutateurs de données (SW1, SW2, SW3). Ces commutateurs sont chargés d'émettre et de recevoir des données dans un réseau en anneau. Deux commutateurs SWi et SWj communiquent entre eux à l'aide de la connexion cnij. Chaque interrupteur SWi est géré par un poste de commande CSi. La description du comportement du système est la suivante : Un commutateur transmet des données via deux connexions: une connexion ouest (pour SW1, c'est cn12) et une connexion est (pour SW1, c'est cn31). Une connexion entre deux commutateurs est considérée comme un canal de communication bidirectionnel

Le canal n'est pas fiable et peut être affecté par l'échec de la coupe. Si la connexion est coupée (cut), Swi émet un événement observable (par exemple, si cn12 est coupé (**cut**), Sw1 émet l'événement observable cut1 et si cn31 est coupé (**cut**), Sw1 émet l'événement observable cut 3) . Ensuite, le commutateur passe en mode d'attente. Si la connexion est rétablie (**reparationCnij**), le commutateur revient à son mode normal. Un interrupteur peut tomber en panne (**panne**), dans ce cas, un mécanisme d'événement observable informe le système de supervision par l'émission de l'événement observable **down**. De plus, la station de

contrôle CSi détecte ce problème et tente de réinitialiser le commutateur SWi (Swireboot). Après une réinitialisation (Swiendreboot), le commutateur est à nouveau opérationnel et émet un événement observable Swiok. Deux types de pannes peuvent survenir sur une station de contrôle.

Tout d'abord, la station peut raccrocher (CSioff) .Lorsque la station retrouve un mode normal, un événement observable CSiok est émis. Cet événement observable est transmis via le commutateur Swi, de sorte que l'événement observable est masqué si le commutateur n'est pas dans son mode normal. Une station peut également redémarrer (CSireboot) et à la fin de la réinitialisation (CSiendreboot), un événement observable CSiok est émis. Le canal de communication entre une station de contrôle et son commutateur est considéré comme fiable et instantané.

En ce qui concerne le système de supervision, chaque interrupteur y est connecté via un canal d'observation. Dans cet exemple, par souci de simplicité, ces canaux d'observation sont des files d'attente instantanées, c'est-à-dire que l'émission d'un message observable par un commutateur (un événement observable) correspond exactement à la réception de ce message par le système de supervision (l'observation).

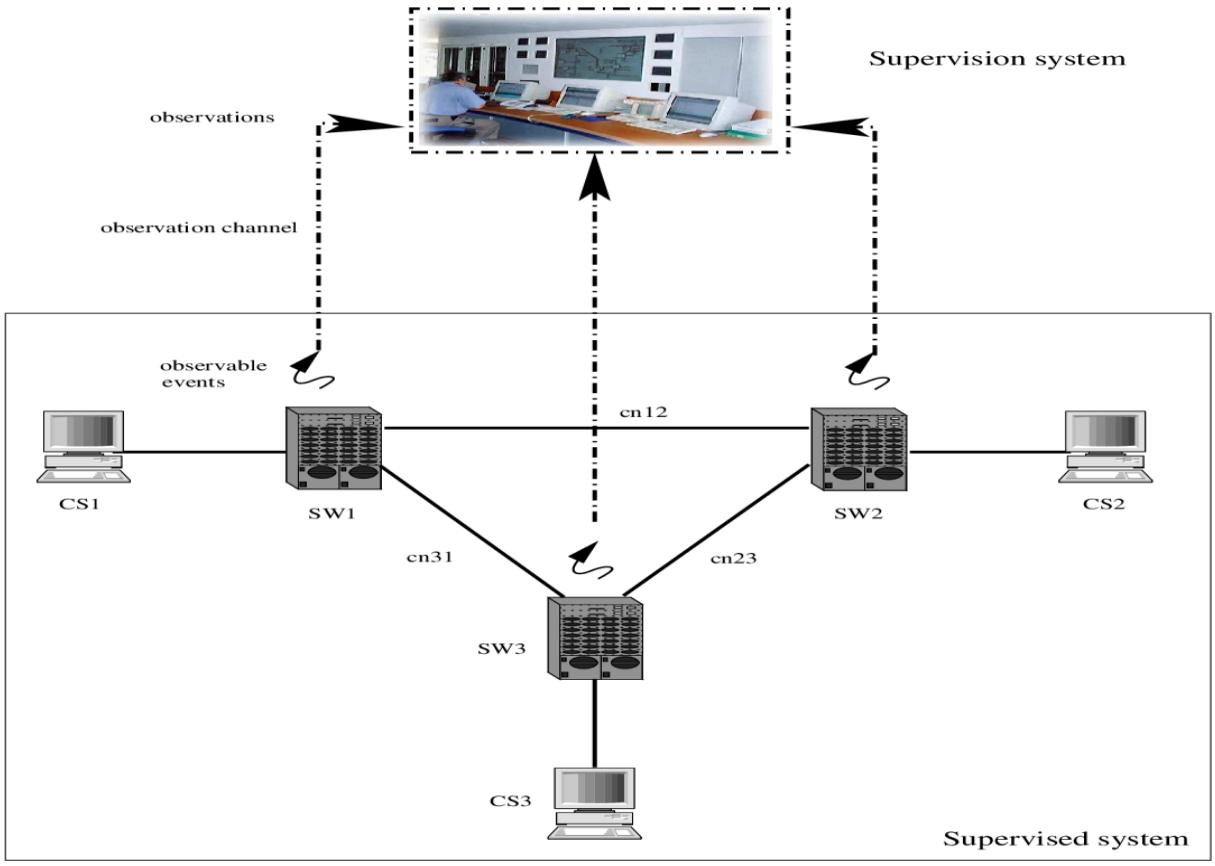

Figure 16: un réseau de télécommunication et leur système de supervision

3. Formalisation du système en formalisme DEVS

Le développement d'une méthode de diagnostic d'un réseau de télécommunication à base d'un modèle de ce réseau se fait en deux étapes. La première étape construit une représentation explicite de la topologie et du comportement du réseau. La seconde étape consiste à développer un modèle DEVS de la topologie et de la propagation de pannes sur le réseau de télécommunication.

Le réseau ToyNet est constitué de trois blocs fonctionnels : canal de connexion(cn), Le Switch(SW) et la station de contrôle (CS).

Le modèle DEVS proposé divise le système en quatre sous-systèmes: Système de Supervision, Switch, Canal et Station de contrôle.

4. Le Modèle DEVS couplée du système

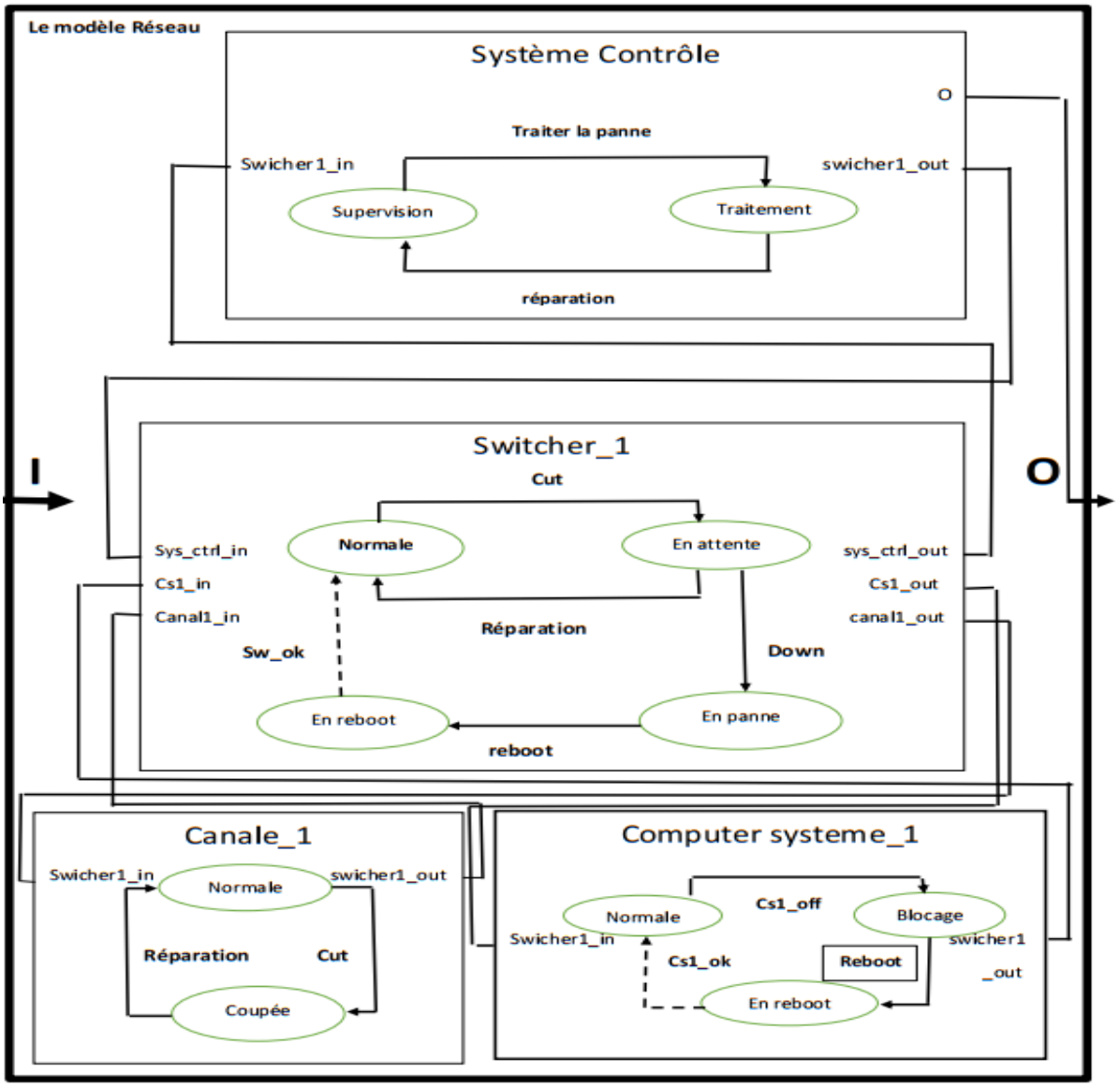

Figure 17 : le modèle couplée du système

5. Les Spécifications de notre système

5.1. La Spécification du sous-système Canal de connexion en DEVS :

➢ Le canal 1

MA Canal 1= (X , Y ,S ,δ_{int} ,δ_{ext} ,λ , ta)

X= {coupure , reparation}

Y={cut}

S= { normale , coupée}

δext:(normale,e,cut)=coupée

δext:(coupée,e,reparation1)=normale

λ=(envoi_message_cut) ta=∞

➢ Le canal 2

MA Canal2 = (X , Y ,S ,δ_{int} ,δ_{ext} ,λ , ta)

X={coupure,reparation}

Y={cut}

S={normale,coupée}

δext : (normale,e,cut)=coupée

δext : (coupée,e,reparation2)=normale

λ =(envoi_message_cut) ta=∞

➢ Le canal 3

MA Canal3= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={coupure,reparation}

Y={cut}

S={normale,coupée}

δext:(normale,e,cut)=coupée

δext:(coupée,e,reparation3)=normale

λ=(envoi_message_cut)

ta=∞

5.2. La Spécification du sous-système Switch en DEVS :

➢ Le switch Sw1

MA SW1= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={cut,reparation1, reparation3,reboot}

Y={cut1,cut3,down,Swok}

S={normale,enattent, panne, reboot}

δext:(normale,e,down)= panne

δext:(normale,e,cut)=enattent

δext:(enattent,e,reparation1)=normale

δext:(enattent,e,reparation3)=normale

δext:(panne ,e,reboot)=reboot

δint:(reboot,e,Swok)=normale ta=0

λ=(envoi_message_Swok) ta=∞

➢ Le switch Sw2

MA SW1= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={cut,reparation1, reparation2,reboot}

Y={cut1,cut2,down,Swok}

S={normale,enattent, panne, reboot}

δext:(normale,e,down)= panne

δext:(normale,e,cut)=enattent

δext:(enattent,e,reparation1)=normale

δext:(enattent,e,reparation2)=normale

δext:(panne ,e,reboot)=reboot

δint:(reboot,e,Swok)=normale ta=0

λ=(envoi_message_Swok) ta=∞

➢ Le switch Sw3

MA SW1= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={cut,reparation2, reparation3,reboot}

Y={cut2,cut3,down,Swok}

S={normale,enattent, panne, reboot}

δext:(normale,e,down)= panne

δext:(normale,e,cut)=enattent

δext:(enattent,e,reparation2)=normale

δext:(enattent,e,reparation3)=normale

δext:(panne ,e,reboot)=reboot

δint:(reboot,e,Swok)=normale ta=0

λ=(envoi_message_Swok) ta=∞

5.3. La Spécification du sous-système station de contrôle en DEVS :

➢ Le station Cs1

MA Cs1= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={ reboot}

Y={Cs1ok ,Cs1off}

S={normale,blocage, reboot}

δext:(normal,e,Cs1off)=blocage

δext:(blocage,e,reboot)=reboot

δint: (reboot,e,Cs1ok)=normal ta=0

λ=(envoi_message_Cs1ok) ta=10

➢ Le station Cs2

MA Cs2= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={ reboot}

Y={Cs2ok ,Cs2off}

S={normale,blocage, reboot}

δext:(normal,e,Cs2off)=blocage

δext:(blocage,e,reboot)=reboot

δint: (reboot,e,Cs2ok)=normal ta=0

λ=(envoi_message_Cs2ok) ta=10

➢ Le station Cs3

MA Cs3= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={ reboot}

Y={Cs3ok ,Cs3off}

S={normale,blocage, reboot}

δint: (reboot,e,Cs3ok)=normal ta=0

δext:(normal,e,Cs3off)=blocage

δext:(blocage,e,reboot)=reboot

δext:(blocage,e,reboot)=reboot

λ=(envoi_message_Cs2ok) ta=10

5.4. La Spécification du sous-système Système de supervision en DEVS :

- **système de supervision**

MA System controle= (X , Y , S ,δ_{int} ,δ_{ext} ,λ , ta)

X={cut1,cut2,cut3,swok,cs1off, cs2off,cs3off ,cs1ok, cs2ok, cs3ok,}

Y={reparation1, reparation2 , reparation3,cs1reboot, cs2reboot , cs3reboot,reboot }

S={supervision, traitement,}

δext:(supervision,e,cs3off)= traitement

δext:(supervision,e,cut1)= traitement

δint:(traitement,e,reparation1)=supervision ta=0

δext:(supervision,e,cut2)= traitement

δint:(traitement,e,reparation2)=supervision ta=0

δext:(supervision,e,cut3)= traitement

δint:(traitement,e,reparation3)=supervision ta=0

δext:(supervision,e,down)= traitement

λ=(envoi_message_reparation1) ta=0

δext:(traitement,e,swok)= supervision

λ=(envoi_message_reparation2) ta=0

δext:(supervision,e,cs1off)= traitement

λ=(envoi_message_reparation3) ta=0

δext:(supervision,e,cs2off)= traitement

5.5. La Spécification du Système global ToyNet en DEVS :

Le modèle couplé de Modèle réseau est défini comme ci-dessous :

CM :< XM, YM, D, EIC, EOC, IC, Select > Avec:

XM = { I , swicher1-in , swicher2-in, swicher3-in, inject2, injet3 , inject4, inject5 ,inject6 ,inject7,inject8,inject9,canal1-in,canal2-in,canal3-in,cs1-in,cs2-in,cs3-in,sys_ctrl-in}

YM = {O, swicher1-out, swicher2-out, swicher3-out, canal1-out, canal2-out, canal3-out, cs1-out, cs2-out,cs3,sys_ctrl-out}

D={system-controle,computer-system1,computer-system2,computer-system3,canal1,canal2,canal3,swicher1,swicher2,swicher3}

EIC =(Reseau,I) { canal1, inject}

(Reseau,I2) { canal2, inject3}

(Reseau,I3) { canal13, inject3}

(Reseau,I4){computer-system1,inject4}

(Reseau,I5){computer-system2, inject5}

(Reseau,I6){computer-system3,inject6},

(Reseau,I7){swicher2, inject7},

(Reseau,I8) { swicher2, inject8},

(Reseau,I9) { swicher3, inject9}

EOC = {sys_ctrl,O}{reseau,O}

IC = {canal1,swicher1-out}{swicher1,canal1-in}
{canal1,swicher2-out}{swicher2,canal1-in}
{canal2,swicher2-out}{swicher2,canal2-in}
{canal2,swicher3-out}{swicher3,canal2-in}
{canal3,swicher1-out}{swicher1,canal3-in}
{canal3,swicher3-out}{swicher3,canal3-in}
{cs1,swicher1-out}{swicher1,cs1-in}
{cs2,swicher2-out}{swicher2,cs2-in}
{cs3,swicher3-out}{swicher3,cs3-in}
{swicher1,canal1-out}{canal1,swicher1-in}
{swicher1,canal3-out}{canal3,swicher1-in}
{swicher1,cs1-out}{cs1,swicher1-in}
{swichr1,sys_ctrl-out}{ sys_ctrl,swicher1-in}
{swichr2,canal1-out}{canal1,swicher2-in}
{swicher2,canal2-out}{canal2,swicher2-in
{swicher2,cs2-out}{cs2,swicher2-in}
{swicher2,sys_ctrl-out}{sys_ctrl,swicher2-in}
{swicher3,canal2-out}{canal2,swicher3-in}
{swicher3,canal3-out}{canal3,swicher3-in}
{swicher3,cs3-out}{canal3,swicher3-in}
{swicher3,sys_ctrl-out}{sys_ctrl,swicher3-in}
{ sys_ctrl,swicher1-out}{swicher1,sys_ctrl-in}
{sys_ctrl,swicher2-out}{swicher2,sys_ctrl-in}
{sys_ctrl,swicher3-out}{swicher3,sys_ctrl-in}
{I7,swicher1-in}{swicher1,inject7}
{I8,swicher2-in}{swicher2,inject8}
{I9,swicher3-in}{swicher3,inject9}
{I,canal1-in}{canal1,inject}
{I2,canal2-in}{canal2,inject2 }
{I3,canal3-in}{canal3,inject
{I4,cs1-in}{cs1,inject4}
{I5,cs2-in}{cs2,inject5}
{I6,cs3-in}{cs3,inject6}
{sys_ctrl,O}{O,reseau}

6. Validation du modèle par la simulation

Nous présentons dans cette section, la méthodologie de validation du modèle DEVS du diagnostic des sur le réseau de télécommunication. La validation consiste à vérifier que les résultats de diagnostic effectués avec ce modèle DEVS sont conformes avec les connaissances métiers relatives au fonctionnement du réseau ToyNet.

7. Validation sur un environnement de simulation

Il existe de nombreux outils et environnements de simulation qui offrent des fonctionnalités de modélisation et de simulation basé sur le formalisme **DEVS** et permettent une présentation des différents modèles **DEVS** atomiques ou couplées. Parmi lesquelles on peut citer (DEVS-SUITE, JDEVS, …etc.), nous présentons ci-dessous l'environnement d'implémentation et les outils utilisés pour modélisation et la simulation modèle proposé.

8. L'environnement de modélisation et de Simulation

Afin d'accomplir notre travail nous avons choisi l'environnement **DEVS-SUITE**, pour modéliser et simuler du diagnostic des pannes dans les systèmes à évènement discrets des réseaux de télécommunication.

8.1. DEVS-Suite

DEVS-Suite est un simulateur d'automates cellulaires et basé sur des composants DEVS parallèles : **CARACTÉRISTIQUES :**

- ✓ Visualisation de composants hiérarchiques box-in-box,
- ✓ Automatisation pour la conception d'expériences,
- ✓ Animation de modèles de simulation avec état et I / Messagerie O.
- ✓ Visualisation d'animation d'exécution synchronisée avec E / S et trajectoires d'état,
- ✓ Trajectoires de données de temps superdense à l'exécution,
- ✓ Visualisation CA,
- ✓ Lecture CA,
- ✓ Bibliothèques de modèles hiérarchiques,
- ✓ Tests I / O Black-Box,
- ✓ PostgreSQL pour stocker les sorties des simulations.

9. Représentation de Modèle proposé à l'aide de l'outil de modélisation DEVS-Suite

9.1.System de supervision

La Figure 17 est une vue en DEVS-SUITE du modèle proposé du diagnostic des pannes dans les systèmes à évènement discrets des réseaux de télécommunication.

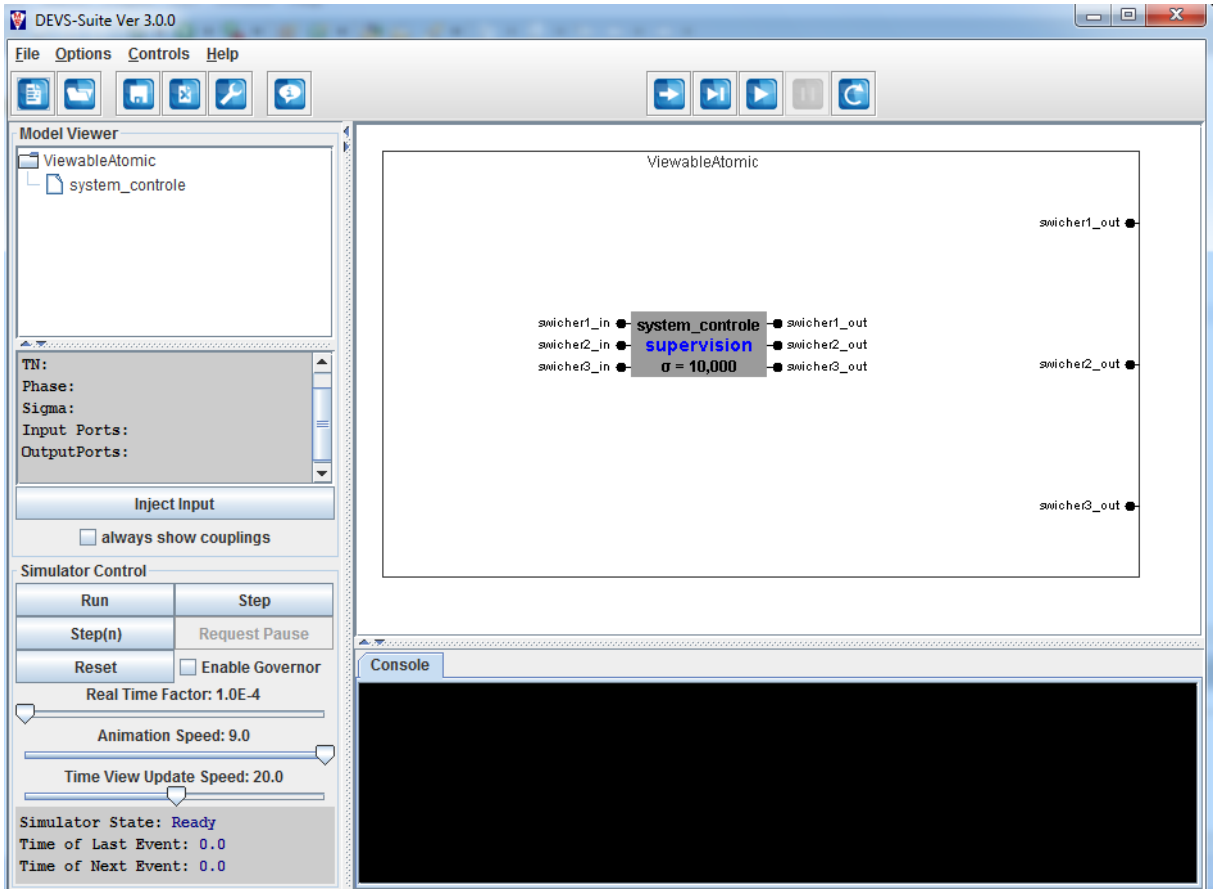

Figure 17: Présentation du Système de supervision

Les figures 18, 19 sont des vues en DEVS-SUITE Représentants les changements d'états et les valeurs dans les ports du Système de supervision.

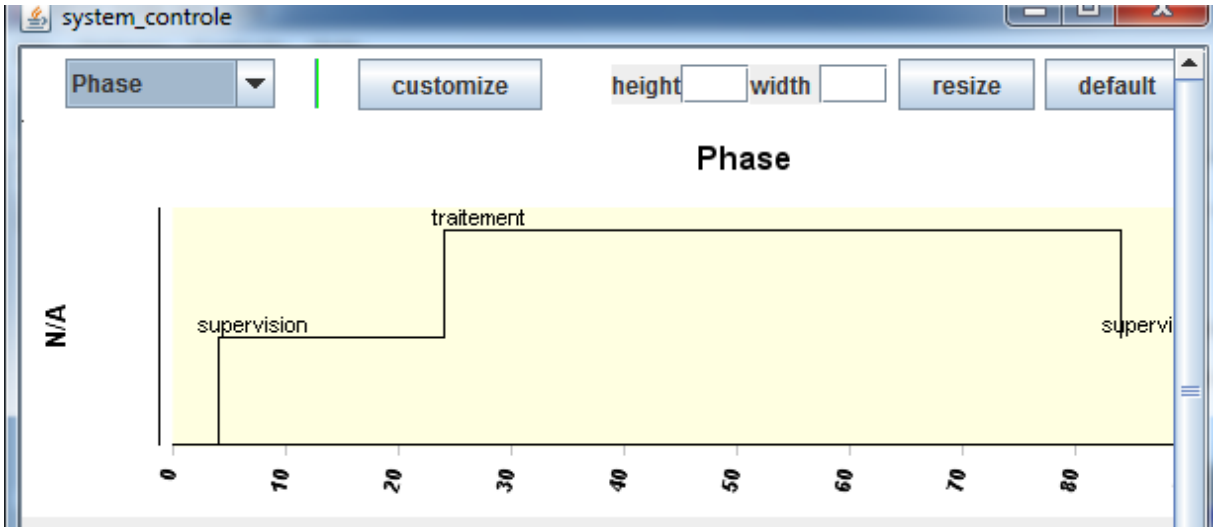

Figure 19:Changement d'état du système de supervision

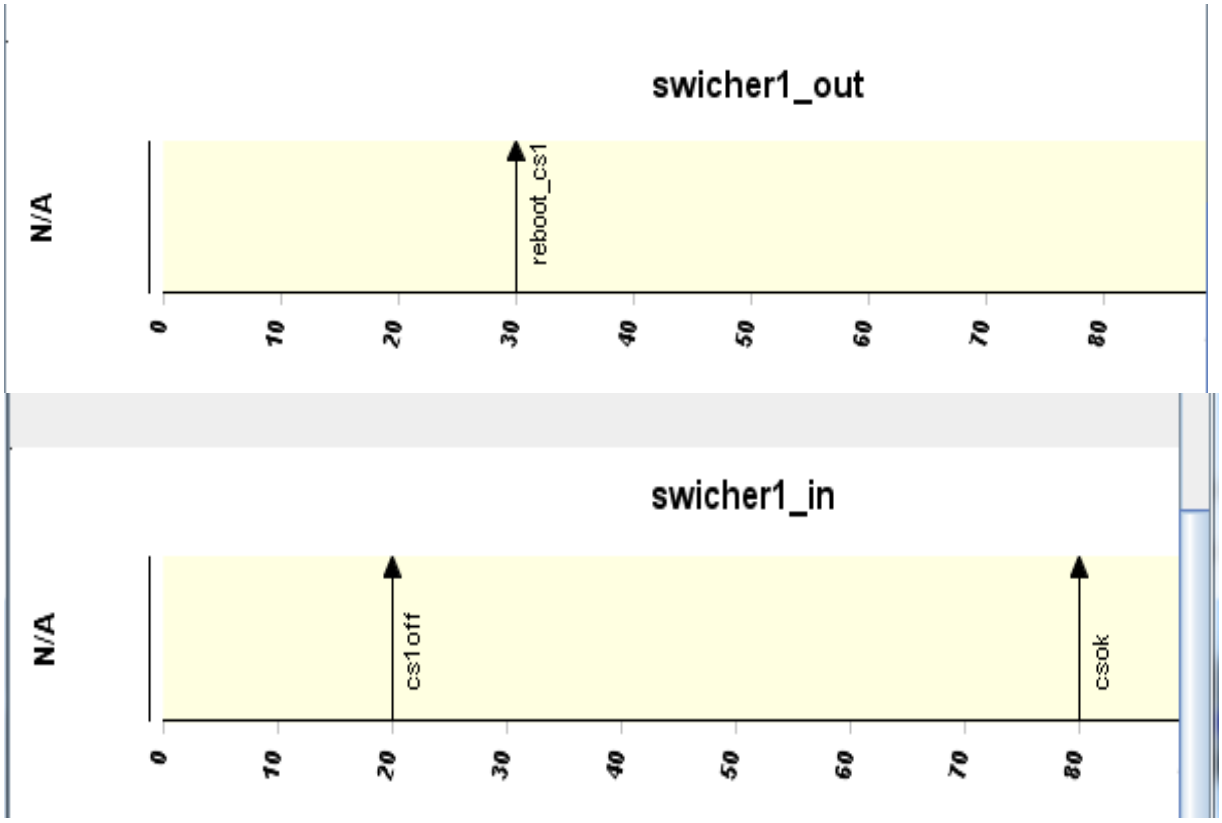

Figure 20: Les valeurs qui correspondant aux ports (entrée/sortie) de system de supervision.

9.2. Switch1/switch2/switch3

La figure 20 présente une vue en DEVS-Suite du modèle atomique switch .

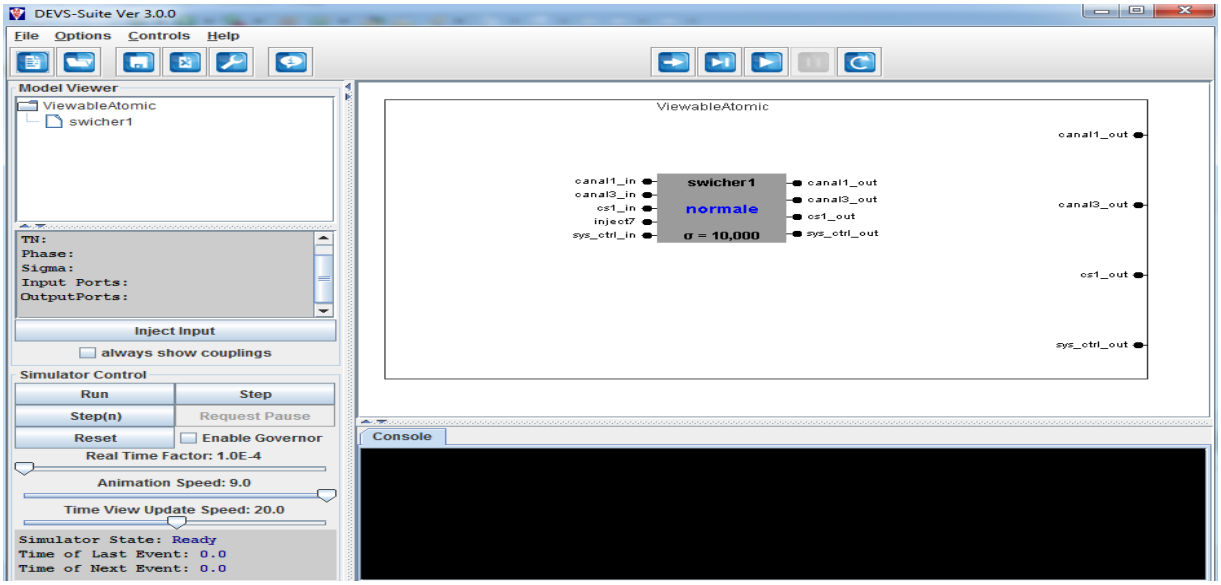

Figure 21: Présentation de Swicher1

Les figures 21 et 22 sont des vues en DEVS-Suite qui illustrent une situation pour laquelle on observe une panne sur le switch1. Il s'agit donc d'une rupture de la communication au Switch1.

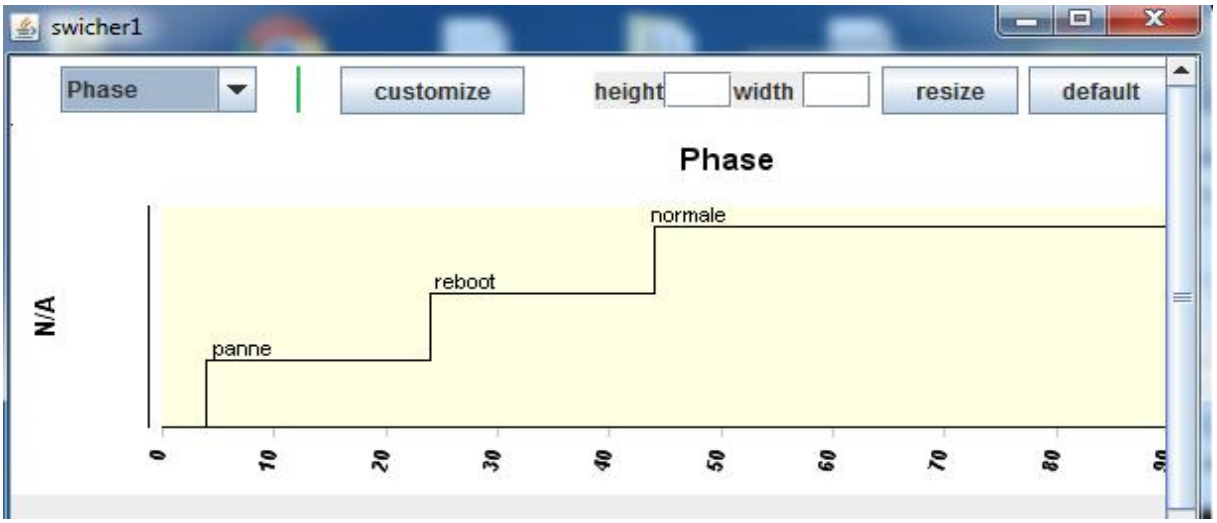

Figure 22: Changements d'état de swicher1

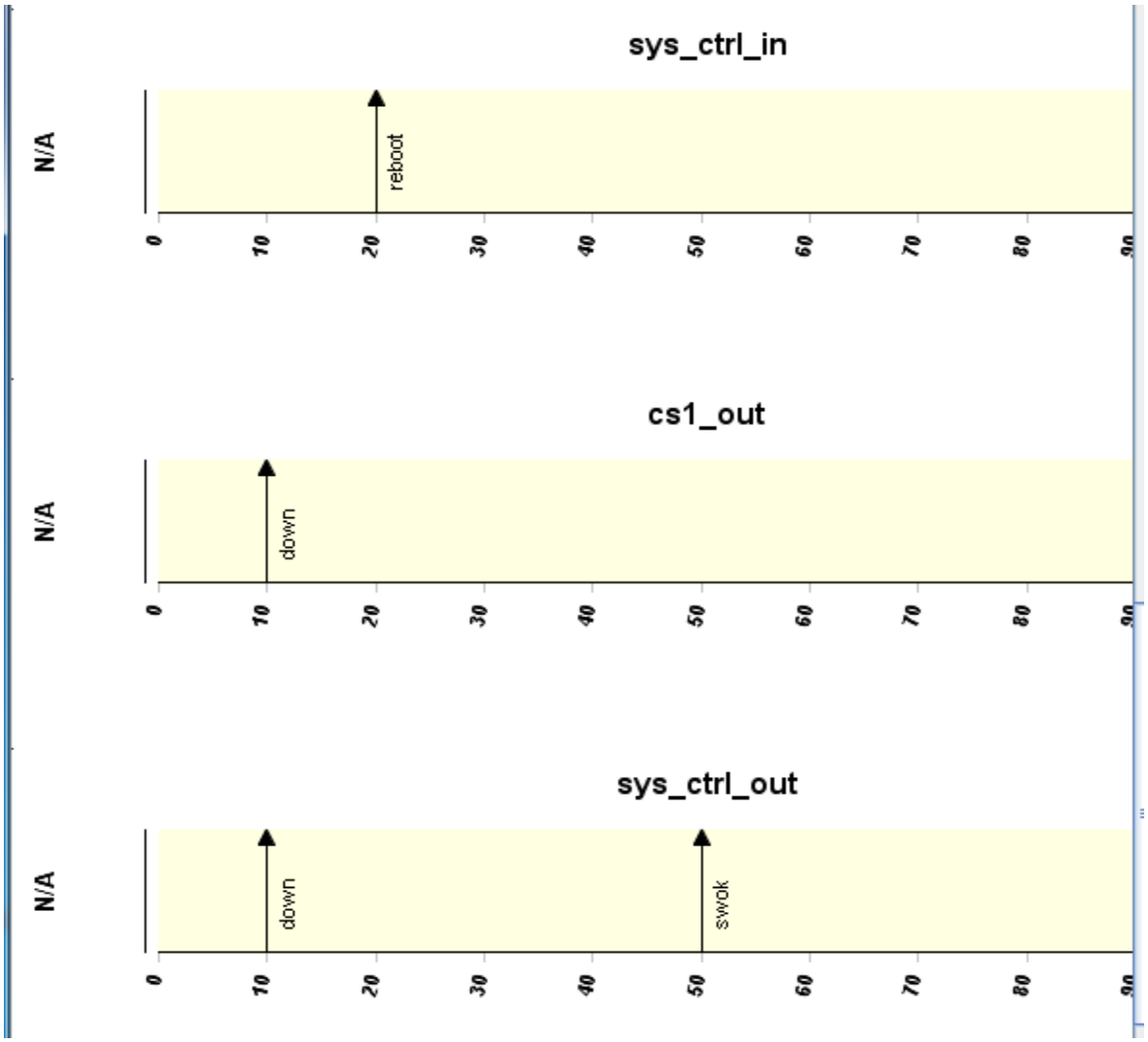

Figure 23: Les valeurs qui correspondant aux ports (entrée/sortie) de swicher1

9.3. Canal1/Canal2/Canal3

La figure 23 est une vue en DEVS-Suite du modèle atomique du canal 1.

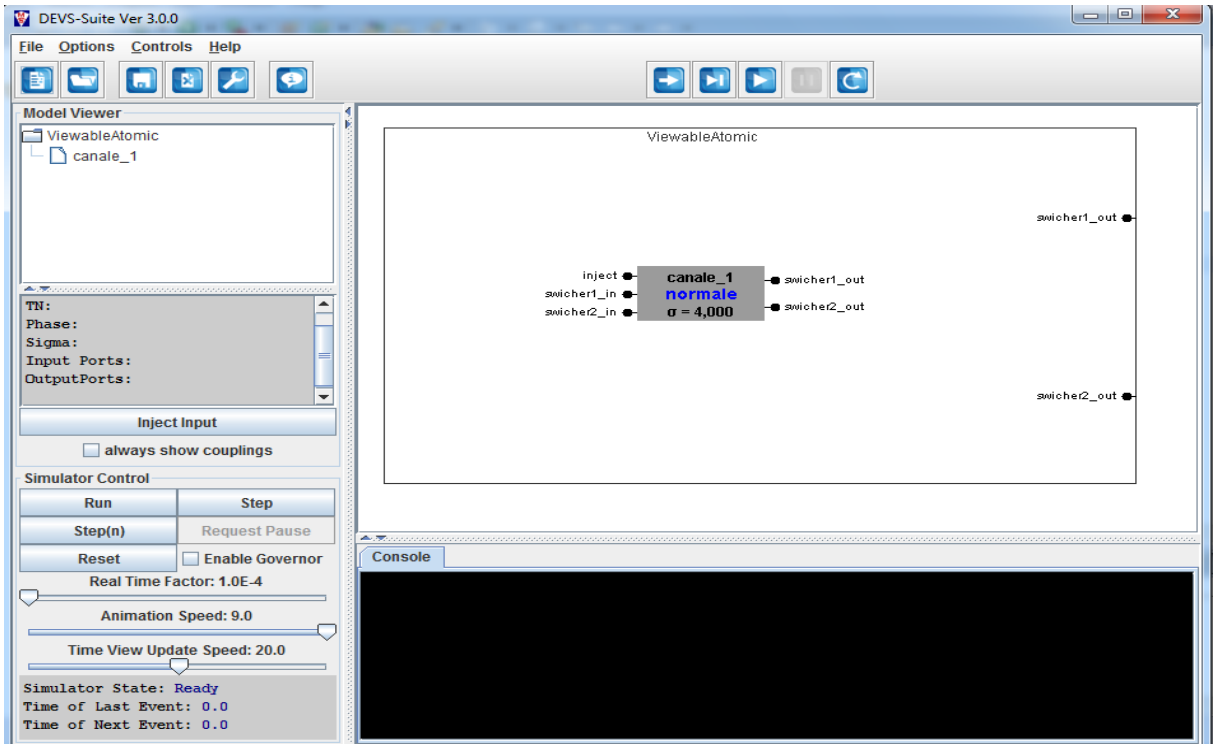

Figure 24: Présentation de canal1

Les figures 24 et 25 présentées en DEVS-Suite montrent que Le canal 1 est diagnostiqué comme étant en panne ou défectueux lorsqu'il perd la communication bidirectionnelle avec le switch

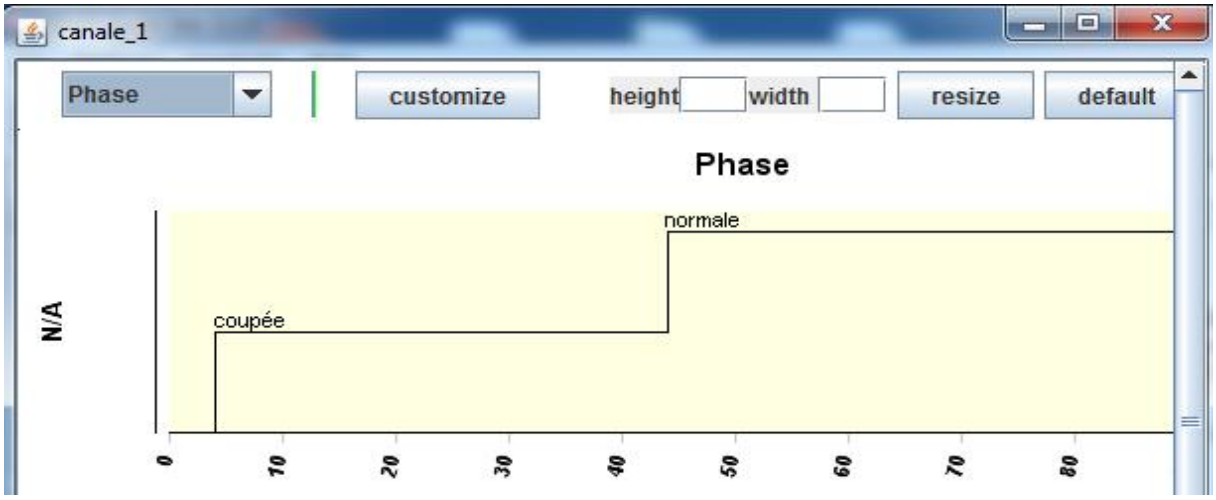

Figure 25: Changements d'état de Canal1

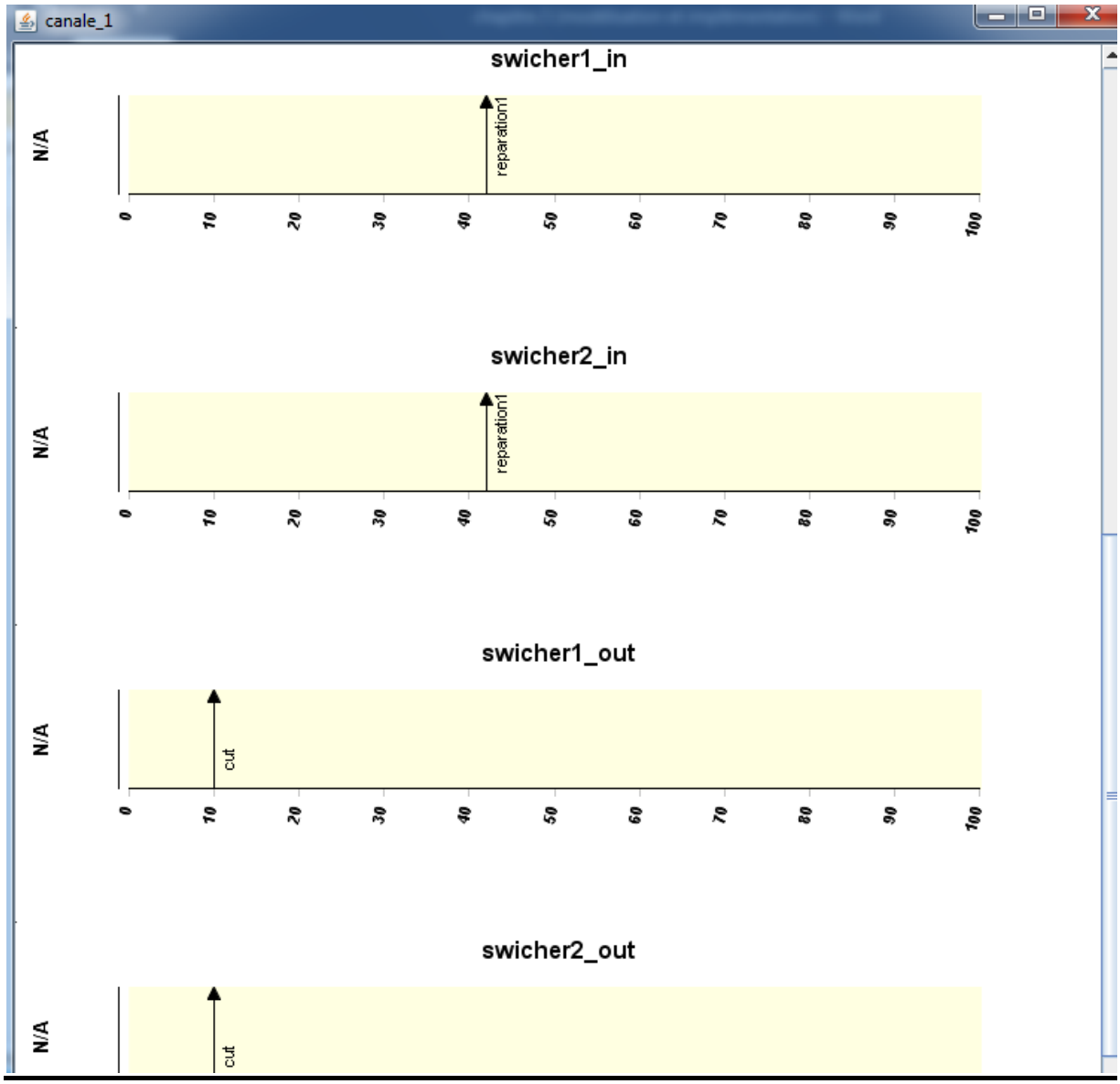

Figure 26 : Les valeurs qui correspondant aux ports(entrée/sortie) de canal1

9.4. Computer system1/Computer sytem2/Computer system 3

La figure 26 est une vue en DEVS-Suite du modèle atomique du Computer system1.

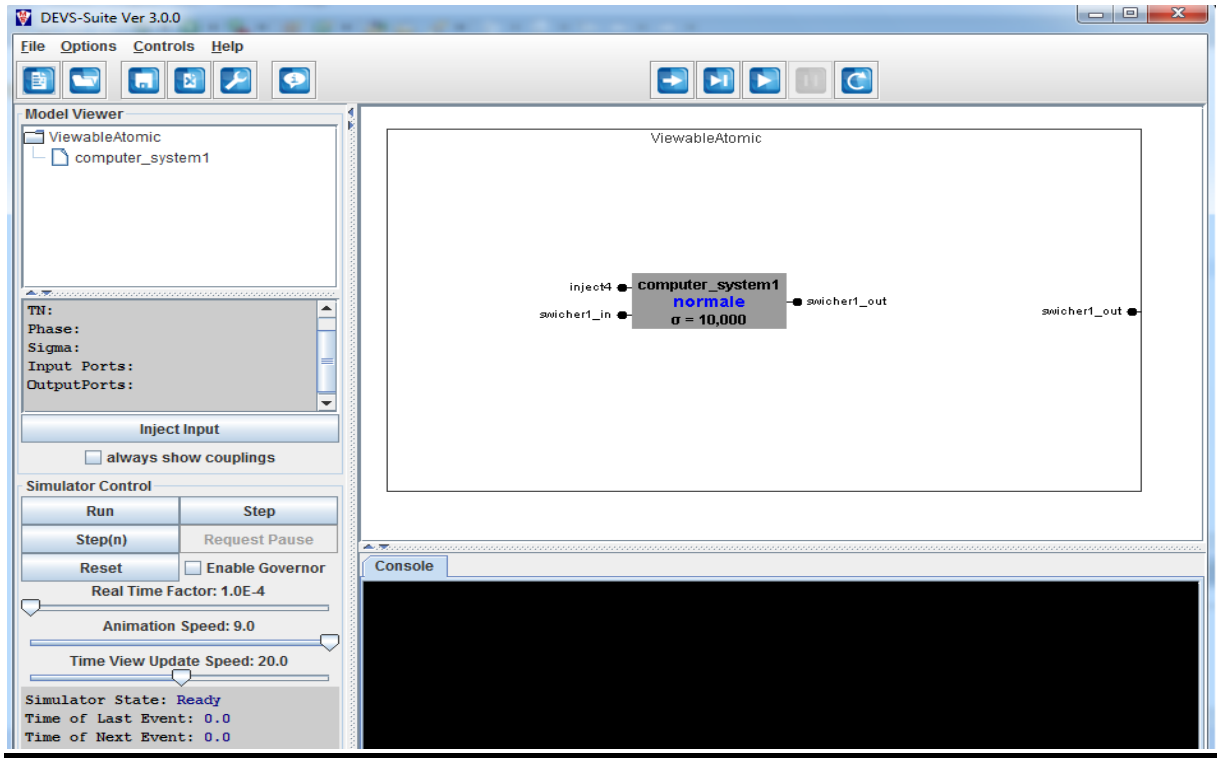

Figure 27: Présentation de Computer system

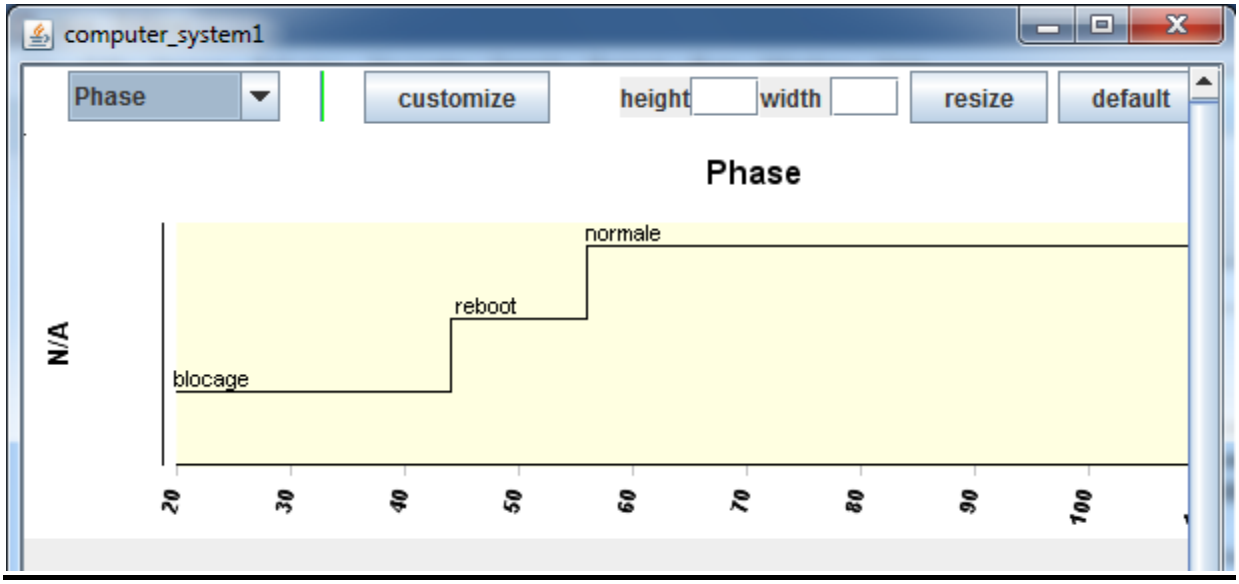

Figure 28 : Changements d'état de Computer system1

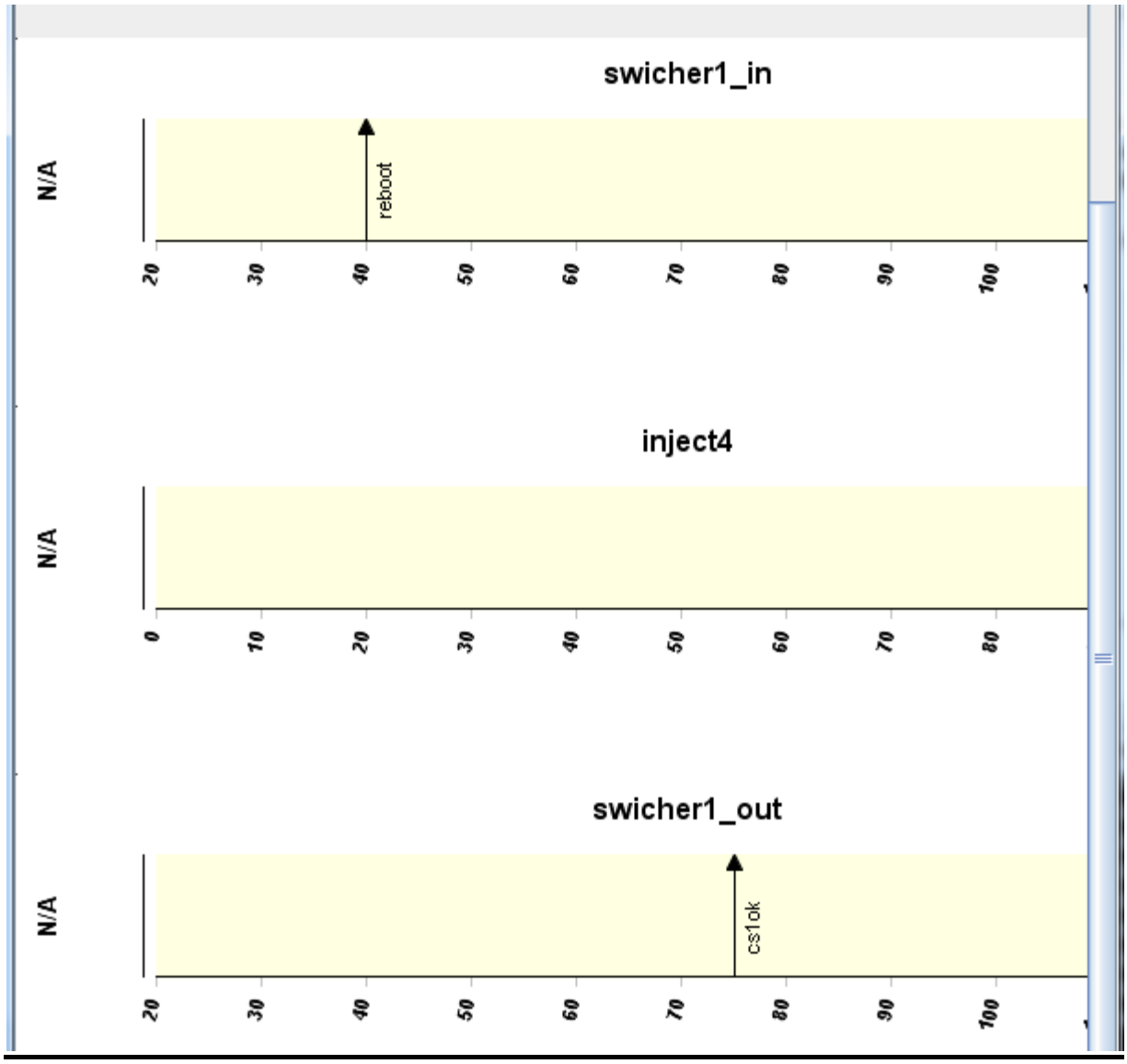

Figure 29: Les valeurs qui correspondant aux ports(entrée/sortie) de Computer system1

Les figures 27 et 28 sont des vues en DEVS-Suite montrent que une perte de la communication entre Computer system1 et le switch 1 peut être due au blocage du Computer system1.

10. Le Modèle Global du système de télécommunication

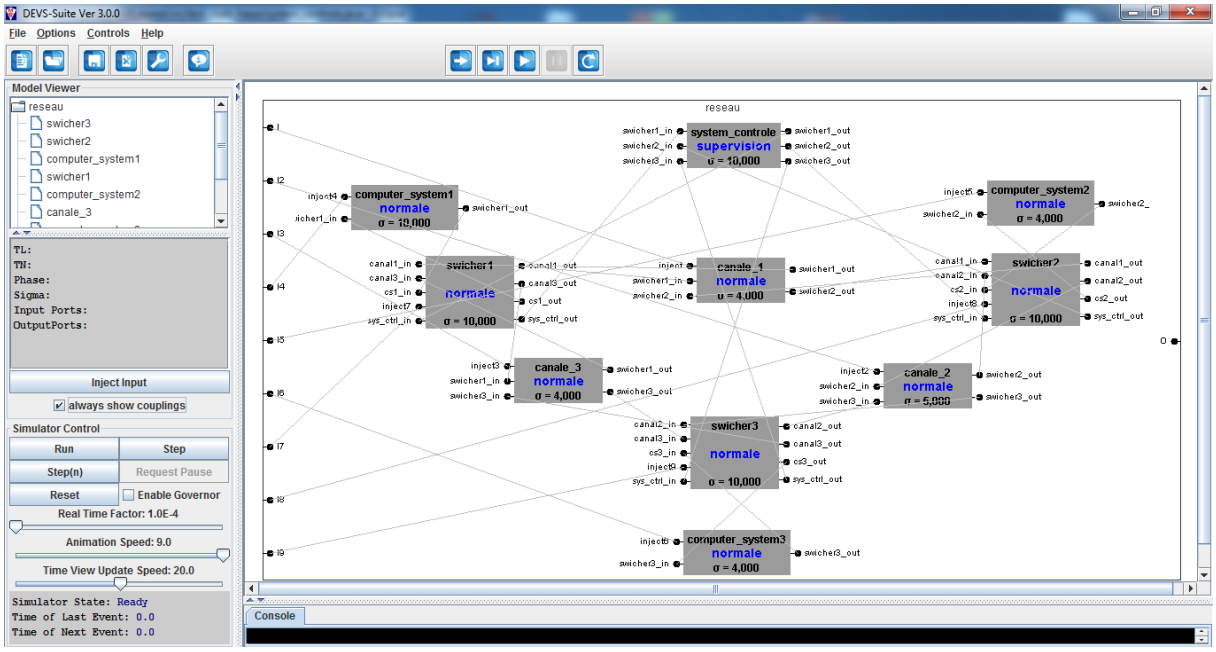

Figure 30 : Présentation du modèle global

Les figures ci-dessus montrent que les résultats de diagnostic obtenus avec le modèle DEVS du réseau ToyNet sont cohérents. En effet, ces résultats ont été inspectés manuellement afin de vérifier leur fiabilité. Ceci démontre bien que le diagnostic automatique d'un réseau de télécommunication avec un modèle à base du formalisme DEVS est une méthode pertinente et très prometteuse.

11. Conclusion

Dans ce chapitre, nous avons exposé la modélisation et simulation du diagnostic des pannes dans les systèmes à évènement discrets des réseaux de télécommunication par le formalisme DEVS ainsi que l'implémentation du modèle proposé sur environnement de modélisation et simulation **DEVS SUITE.**

Tous d'abord, nous avons présenté la spécification formelle détaillée en utilisant le formalisme DEVS. Ensuite nous avons présenté les résultats de simulation de notre système en utilisant l'environnement de modélisation et simulation DEVS-Suite.

Conclusion générale

Conclusion générale

Les opérateurs de télécommunication font de plus en plus d'efforts pour fournir des services de très bonne qualité à leurs abonnés. En outre, les réseaux de télécommunication doivent être fiables et robustes pour garantir également la haute disponibilité de ces services. La gestion des réseaux est de ce fait devenue un problème central pour les opérateurs de télécommunication qui développent des travaux de recherches afin d'automatiser autant que possible les opérations complexes de gestion et d'administration des réseaux, telles que la gestion de pannes. Le diagnostic de pannes est un aspect central de la gestion et de l'administration d'un réseau.

Dans ce travail nous avons proposé une nouvelle méthode de diagnostic des systèmes à d'événements discrets (DES). Cette approche est basée sur le formalisme DEVS qui fournit une méthodologie pour la construction de modèles de simulation modulaires, hiérarchiques et réutilisables dans le but de simuler des systèmes dynamiques complexes. Ce formalisme de modélisation et de simulation, issu de mathématiques discrètes, permet de modéliser des systèmes compliqués dans une très grande variété de domaines. Il est basé sur des événements discrets pour la modélisation de systèmes discrets et continus. Le modèle est vu comme un réseau d'interconnexions entre les modèles atomiques et couplés formant une hiérarchie de modèles. Les modèles interagissent par l'échange d'événements pampilles.

La modélisation du comportement d'un réseau est un problème central pour le développement d'une méthode de diagnostic à base d'un modèle de ce réseau. Dans ce travail nous avons construit un modèle pour le diagnostic des pannes des réseaux de télécommunication, ce modèle est basé sur le formalisme DEVS. Afin de valider le modèle proposé nous avons utilisé l'environnement **DEVS SUITE** comme un outil de simulation et modélisation.

Les résultats de simulation ont démontré bien que le diagnostic automatique d'un réseau de télécommunication avec un modèle à base du formalisme DEVS est une méthode pertinente et très prometteuse.

Perspective

Les travaux futurs visent à développer une approche mixte du diagnostic de systèmes d'événements discrets complexes qui consiste à combiner le formalisme DEVS et l'une des théories de l'incertitude pour étudier les systèmes dynamiques de grande taille afin d'assurer les objectifs de diagnostic prédéfinit dans cette thèse.

Bibliographie

[1]:Livres de simulation a evenement discret ,cours : ***Introduction à la Simulation*** *Dr. Mesut Güneş*

[2] : Mémoire de fin d'étude : LE DIAGNOSTIC A BASE MODELE DES SYSTEMES MULTI-AGENTS réalisée par ***TAHRI Zohair*** *en 2012*

[3] : Mémoire de fin d'étude : Diagnostic à base de modèles des systèmes temporisés et d'une sous-classe de systèmes dynamiques hybrides réalisée par ***Haithem Derbel*** *en 2009*

[4]: M. Sampath, R. Sengupta, and S. Lafortune. Diagnosability of Discrete-Event Systems. IEEE Transactions Automatic Control, 40(9):1555–1575, 1995.

[5] : S. Jiang, Z. Huang, V. Chandra, and R. Kumar. A Polynomial Algorithm for Testing Diagnosability of Discrete Event Systems. IEEE Transactions Automatic Control, 46(8):1318–1321,2001.

[6] : T. Yoo and S. Lafortune. Polynomial-Time Verification of Diagnosability of Partially Observed Discrete-Event Systems. IEEE Transactions Automatic Control, 47(9):1491–1495, 2002.

[7]: Y. Wen and M. Jeng. Diagnosability Analysis Based on T-invariants of Petri Nets. In Networking, Sensing and Control, pages 371–376, 2005.

[8] : M.P. Cabasino, A. Giua, and C. Seatzu. Diagnosability of Bounded Petri Nets. In Proc. of the 48th IEEE Conf. on decision and control. Shanghai, China. December, pages 1254–1260, 2009.

[9] : F. Basile, P. Chiacchio, and G. De Tommasi. On K-diagnosability of Petri Nets via Integer Linear Programming. Automatica, 48(9):2047–2058, 2012.

[10] : M.P. Cabasino, A. Giua, S. Lafortune, and C. Seatzu. A New Approach for Diagnosability Analysis of Petri Nets Using Verifier Nets. IEEE Transactions Automatic Control, 57(12):3104–3117, 2012.

[11] : B. Liu, M. Ghazel, and A. Toguyéni. Toward an Efficient Approach for Diagnosability Analysis of DES Modeled by Labeled Petri Nets. In 13th European Control Conference - ECC'2014, 2014.

[12] : Diagnosticabilité des systèmes à événements discretsFarid Nouioua et Philippe Dague

[13] : Mémoire de fin d'étude : Contribution au diagnostic et pronostic des systèmes à évènements discrets temporisés par réseaux de Petri stochastiques réalisée par ***Ammour Rabah*** *en 2017*

[14][Zeigler et al, 2000] B. P. Zeigler, T. G. Kim et H. Praehofer, Theory of Modeling and Simulation. Academic Press

[15] *[Zeigler84] (en) Bernard Zeigler, Multifacetted Modeling and Discrete Event Simulation, Academic Press, London; Orlando, 1984, 372 p.*

[16][Ramat, 2003] E. Ramat, Contributions à la modélisation et à la simulation des systèmes complexes, Habilitation à diriger des recherches, 2003.

[17][Quesnel, 2006] G. Quesnel, Approche formelle et opérationnelle de la multimodélisation et de la simulation des systèmes complexes, Laboratoire d'Informatique du Littoral, 2006.

Liste des abréviations

DEVS : Discret Event System Spécification.

Systèmes à ED : systèmes à Evènement Discret.

RdP : Réseaux De Pétri.

SDH : Systèmes Dynamiques Hybrides.

FDI : Fault Detection and Isolation

MBRG :Modified Basis Reachability Graph

BRD : Basis Reachability Diagnoser

VN :Verifier Net

MA : Modèle Atomique

MC : Modèle Couplé

IC : Infernal Coupling

EIC : External Input Coupling

EOC : External Output Coupling

MSDL : Modelling Simulation and Design Lab

VLE : Vertual Laboratory Environnement

Résumé

La tâche de diagnostic consiste à détecter les anomalies intervenant sur un système puis à expliquer ces erreurs en indiquant les composants pouvant être défectueux. Pour cela, le diagnostic à base de modèles utilise une description du fonctionnement du système .Le défaut est détectée lorsque les données du modèle et les données du systèmes sont incohérentes.

Le but de ce mémoire est d'utiliser le formalisme DEVS pour diagnostiquer les systèmes d'événements discrets (DES). Le diagnostic basé sur le formalisme DEVS permet de détecter et localiser toute anomalie pouvant survenir dans le système. Ce système peut être simple ou complexe selon les besoins. Le formalisme DEVS nous a permis de modéliser certaines applications industrielles. Cette approche est basée sur une nouvelle structure qui sépare explicitement les états normaux des états défectueux dans le diagnostiqueur. Une telle distinction permet de suivre séparément l'évolution des traces normales et défectueuses dans le diagnostiqueur. L'évaluation de cette approche est illustrée par La modélisation et simulation d'un système complexe de réseau de télécommunication sous le simulateur DEVS SUITE.

Mots clés : DES, DEVS, diagnostiqueur, états normaux, états défectueux.

Abstract

The diagnostic task consists of detecting anomalies occurring on a system and then there errors by indicating the components that may be defective. For this, the Model-based diagnostics use a description of how the system works.is detected when model data and system data are inconsistent.The purpose of this thesis is to use the DEVS formalism to diagnose discrete event systems (DES).

Diagnosis based on the DEVS formalism makes it possible to detect and locate any anomaly that may arise in the system. This system can be simple or complex as required. The DEVS formalism allowed us to model certain industrial applications. This approach is based on a new structure that explicitly separates normal states from faulty states in the diagnostic tool. Such a distinction makes it possible to follow the evolution of normal and defective traces in the diagnostician. The evaluation of this approach is illustrated on a complex telecommunication network system. We will see the modeling of this system and their implementation in the DEVS SUITE simulator.

Keywords: DES, DEVS, diagnoser, normal states, faulty states.

ملخص

تمكن مهمة التشخيص من اكتشاف حالات الأخطاء، مع الإشارة إلى المكونات التي قد تكون معيبة. لهذا، فإنها تستخدم التشخيصات القائمة على النموذج وصفًا لكيفية عمل النظام.يتم اكتشافه عندما تكون بيانات النموذج وبيانات النظام غير متسقة.الغرض من هذه الأطروحة هو استخدام شكلية DEVS لتشخيص أنظمة الأحداث المنفصلة (DES) .

يتيح التشخيص المستند إلى شكلية DEVS اكتشاف وتحديد أي شذوذ قد ينشأ في النظام. يمكن أن يكون هذا النظام بسيطًا أو معقدًا حسب الحاجة. سمحت لنا شكليات DEVS بنمذجة الشاذة التي تحدث في النظام بعض التطبيقات الصناعية. يعتمد هذا النهج على بنية جديدة تفصل صراحةً الحالات الطبيعية عن الحالات السيئة في أداة التشخيص. مثل هذا التمييز يجعل من الممكن متابعة تطور الآثار الطبيعية والمعيبة في أداة التشخيص بشكل منفصل. يتم توضيح تقييم هذا النهج في نظام شبكة اتصالات معقد. سنرى نمذجة هذا النظام وتنفيذها في محاكي DEVS SUITE.

الكلمات المفتاحية: DES، DEVS ، التشخيص ، الحالات الطبيعية ، الحالات المعيبة.