Contribution à la quantification des programmes de maintenance complexes

Thomas Ruin

École Doctorale IAEM Lorraine

Département de Formation Doctorale Automatique

Pôle scientifique AM_2I

Thèse

Présentée pour l'obtention du grade de

Docteur de l'Université de Lorraine

en Automatique, Traitement du Signal et des Images, Génie Informatique

par **Thomas RUIN**

Contribution à la quantification des programmes de maintenance complexes

Soutenue publiquement le 09/12/2013 devant la commission d'examen composée de :

Rapporteurs :	M. Frédéric KRATZ	Professeur à l'ENSI de Bourges
	M. Olivier SENECHAL	Professeur à l'Université de Valenciennes et Hainaut Cambrésis
Examinateurs :	Mme Anne BARROS	Professeur à l'UTT
	M. Antoine DESPUJOLS	EDF R&D
	M. Benoît IUNG	Professeur à l'Université de Lorraine (Directeur de thèse)
	M. Eric LEVRAT	Maitre de conférences à l'Université de Lorraine (Co-directeur de thèse)

Centre de Recherche en Automatique de Nancy, UMR 7039, Université de Lorraine, CNRS

Fédération Charles Hermite, FR 3198, Université de Lorraine, CNRS

« Je n'ai que des questions à vos réponses »

Woody Allen

Remerciements

Les travaux présentés dans ce mémoire ont été réalisés au Centre de Recherche en Automatique de Nancy (CRAN) dans le cadre du projet DEPRADEM 2 supporté par le GIS 3SGS.

Je tiens en premier lieu à remercier messieurs Eric Levrat et Benoît Iung, professeurs à l'Université de Lorraine, pour la confiance qu'ils m'ont accordée. Leurs précieux conseils m'ont permis d'acquérir la rigueur nécessaire à la réalisation d'un travail scientifique.

Je remercie vivement monsieur Olivier Sénéchal, professeur à l'Université de Valenciennes et du Hainaut Cambrésis, ainsi que monsieur Frédéric Kratz, professeur à l'ENSIB, pour avoir accepté de rapporter mon travail. Leur remarques pertinentes et complémentaires m'ont permis de prendre le recul nécessaire à la concrétisation de mes travaux.

De même, je tiens à remercier monsieur Antoine Despujols, ingénieur de recherche à EdF R&D, pour avoir accepté d'examiner cette thèse. Sa participation active à ce projet et nos échanges réguliers au cours de ces quatre années m'ont permis d'intégrer les enjeux industriels ainsi qu'une vision plus pragmatique du sujet nécessaire à la réalisation de ce travail. Je remercie également madame Anne Barros, professeur à l'UTT, d'avoir accepté de présider ce jury, mais aussi de m'avoir encouragé dans mon choix de poursuivre ma formation par la recherche.

Merci également à Yongxin, Alex, Fabien, Pierre, Mario, Leila, Esma, Jérémy, Sylvain, Ludovic, Gabriela, Mario et à tous les doctorants, maitres de conférences ou professeurs du CRAN pour avoir largement contribué à la réussite de mon aventure nancéenne.

J'en profite pour placer une mention à mes amis troyens pour leur fidélité et leur soutien parfois inconscient mais toujours précieux.

Last but not least, merci à mon père, à ma mère, à mon frère ainsi qu'à toute ma famille. Ils m'ont toujours supporté dans les moments difficiles. Ils furent le moteur de ma détermination à mener à bien ce travail.

Sommaire

Liste de figures

Liste des tableaux

Abréviations

ADF AltaRica DF

AM Action de maintenance

BDD Diagramme de définition de blocs (Block Definition Diagram)

CMPQ Quantification des programmes de maintenance complexes (Complex Maintenance Programs Quantification)

EDF Electricité De France

KPI Indicateur de performance clé (Key Performance Indicator)

MCO Maintien en conditions opérationnelles

PD Diagrammes paramétriques (Parametric Diagram)

SD Diagrammes de séquence (Sequence Diagram)

SdF Sûreté de Fonctionnement

SP Système principal

SS Système de Soutien

STM Diagrammes d’états (State Machine Diagram)

SysML System Modeling Language

Introduction générale

Face aux nouveaux cadres législatifs, sociaux, environnementaux dans lesquels ils doivent évoluer, les systèmes industriels actuels sont, en plus d'être contraints par des exigences classiques de productivité et de cout, sujets au respect de nouvelles exigences relatives à la sûreté, la sécurité ou au développement durable notamment. Ceci a pour conséquence essentielle une complexité croissante dans leur fonctionnement à la fois en vision externe pour que la coopération/collaboration des activités réalisées satisfasse à ces différentes exigences et en vision interne pour que les architectures déployées supportent efficacement ces activités. Un enjeu majeur en synergie de cette double vision est donc relatif à la maîtrise des performances attendues tout en conservant le système aussi proche que possible de son espace de fonctionnement nominal. En réponse à cet enjeu, la phase de Maintien en Conditions Opérationnelles (MCO) du système (ou système principal SP au sens Ingénierie Système (AFIS, 2009)) joue un rôle fondamental tel que souligné par (Alsyouf 2007). Ce MCO est supporté par les systèmes contributeurs du SP, et notamment son système de soutien (SS), dont la composante principale est le sous-système maintenance. De plus, les couts de maintenance représentent environ 15 à 70% des couts globaux de possession du système en considérant à la fois les couts directs (liés à la maintenance proprement dite), mais surtout les couts indirects relatifs à l'impact de la satisfaction ou non des performances attendues (Bevilacqua & Braglia 2000). Cette considération de cout global de possession requiert une évolution dans la manière de concevoir, d'exploiter, de démanteler le système principal, pour être en synergie avec ses systèmes contributeurs tel que préconisée par l'Ingénierie Système. Par conséquent, l'objectif doit être à terme de disposer d'outils/méthodes permettant de quantifier a priori l'impact de programmes de maintenance complexes sur les performances d'un SP, c'est-à-dire de d'évaluer les Key Performance Indicators (KPIs) globaux relatifs aussi bien au SP (e.g. disponibilité, cout) qu'au SS (e.g. organisation, ressources), mais aussi relatifs au contexte du SP et à l'activité de maintenance faisant interagir SS et SP. Ainsi, en utilisant des résultats fournis par la simulation du comportement et de l'évolution de modèles de ces systèmes, la prise de décision tout au long du cycle de vie du système (par exemple la re-conception du SP en fonction de contraintes du SS; le changement de stratégie de maintenance en exploitation) est facilitée. De tels modèles permettent, dès la conception des systèmes, d'identifier par exemple l'action de maintenance la plus adaptée pour agir sur ces KPIs, ou encore les éléments du SS critiques sur ces KPIs.

De nombreux travaux ont déjà cherché à contribuer au développement de tels modèles (outils/méthodes) basés sur une vision système conjointe du SP et du SS. (Monnin et al. 2011) se sont par exemple intéressés à la disponibilité opérationnelle de systèmes de combat. (Piero Baraldi et al. 2011) (Zille 2009) se sont quant à eux intéressés à l'évaluation des KPIs de système de production d'énergie. Ces recherches ont mis en évidence plusieurs verrous scientifiques à attaquer, parmi lesquels les principaux sont :

- L'identification et la formalisation des KPIs requis, conventionnels et émergents, dans une vision de synergie entre le SP et le SS,
- L'identification de la connaissance générique (indépendamment du cadre applicatif) statique (hors comportement) et comportementale impliquée dans la quantification des programmes de maintenance complexes,
- L'élicitation du retour d'expérience (REX) sur le SP et le SS pour obtenir les données nécessaires à l'évaluation des KPIs dans un cadre d'application particulier ;

- La formalisation d'un modèle de simulation exploitant la connaissane identifiée relativement au SP et au SS.

Nos travaux de thèse s'inscrivent totalement en cohérence avec ces verrous et en support de l'objectif global mais plus précisément sur le sous objectif de quantification de la Maintenance (e.g. KPIs Maintenance) à son niveau opérationnel, à savoir lors de l'application au SP d'un programme de maintenance faisant intervenir pour son implantation une synergie entre les différents systèmes contributeurs. Cette thèse trouve sa genèse industrielle dans le cadre d'un partenariat avec EDF au sein du projet DEPRADEM 2 (modélisation de la DEgradation pour le PRonostic et l'Aide à la DEcision de Maintenance) du GIS 3SGS (Groupement d'intérêt Scientifique – Sûreté, Sécurité et Surveillance des Grands Systèmes). Notre objet de recherche principal est ainsi plus spécifiquement un système industriel de production d'électricité, de type centrale nucléaire, système fortement contraint en raison des risques induits, voire des évènements récents.

Pour contribuer à la maîtrise des performances de ces systèmes particuliers à EDF, une démarche d'Optimisation de la Maintenance basée sur la Fiabilité (OMF) (Despujols, 2001) a déjà été mise en place sur les centrales en exploitation. Cette approche permet de choisir une politique de maintenance préventive en fonction d'objectifs de sûreté, de disponibilité et de couts. Compte tenu de l'évolution des contextes dans lesquels ces centrales fonctionnent, EDF souhaite faire évoluer cette démarche vers une démarche (INPO, 2001) reprenant un certain nombre d'éléments de la démarche OMF, en y ajoutant des tâches complémentaires nouvelles telles que l'analyse des causes de défaillance, la gestion de l'obsolescence ou la définition et l'utilisation d'indicateurs concernant les tâches d'organisation de la maintenance. Elle doit en effet pouvoir s'appliquer à des systèmes de grande dimension (un programme de maintenance pour un système sujet à cette démarche se compose d'environ 1000 tâches réparties), non critiques en termes de sûreté, et dont les arrêts de tranche coutent environ un million d'euros en moyenne.

Par conséquent à partir du besoin industriel EDF et en répondant en partie aux verrous scientifiques mis en évidence précédemment, cette thèse a pour objet de développer, à partir de l'identification des connaissances métier sous la forme de concepts liées à la quantification des programmes de maintenance complexes (CMPQ), une méthodologie. Cette méthodologie est basée sur la construction de diagrammes statiques et dynamiques à base du langage semi formel SysML (System Modeling Language, (OMG, 2010), puis d'une exploitation de ces diagrammes pour créer un modèle support à la simulation stochastique à base du langage AltaRica Data Flow (ADF,(Rauzy, 2002)), en établissant des correspondances entre les différents éléments sémantiques de chaque langage utilisés dans les deux modèles uniquement. Par rapport au verrou lié à la modélisation de la connaissance statique, le langage SysML propose, à travers différents diagrammes, une vue statique de niveau conceptuel permettant de modéliser cette connaissance de manière fidèle (e.g. utilisation des normes, standards, des concepts de la sûreté de fonctionnement). Afin d'obtenir un modèle de dynamique de la connaissance, SysML propose des diagrammes comportementaux, ainsi que des méthodes et liens permettant la construction cohérente de ces diagrammes comportementaux en fonction de la connaissance ajoutée et des diagrammes statiques précédemment définis. Le résultat de cette modélisation est un ensemble de diagrammes génériques, à la fois statiques et comportementaux, représentant toute la connaissance métier nécessaire à la CMPQ. Afin de rendre cette connaissance utilisable pour calculer et évaluer les KPIs requis, il est nécessaire, dans un second temps, de proposer

et formaliser les correspondances entre les éléments sémantiques des diagrammes vers ceux d'un langage supportant la simulation stochastique. Le langage ADF a été choisi pour cette simulation et les objets modélisés dans ce langage sont construits de façon générique, et en cohérence avec les différents diagrammes SysML proposés. Afin de valider la démarche, une instanciation du modèle ADF a été réalisée afin d'obtenir les KPIs requis sur un système réel fourni par EDF : le système ARE. Par conséquent, notre contribution méthodologique de thèse propose les originalités suivantes :

- L'identification de la connaissance conjointe à quatre axes d'intérêts : le SS, le SP, son contexte et l'activité de maintenance,
- L'expression textuelle, organisé sous la forme de concepts de connaissance, de la sémantique de la connaissance préalablement identifiée,
- La proposition de deux modèles génériques, et de règles de correspondances entre les éléments sémantiques d'intérêt de ces modèles. Un premier, dans un langage décorrélé de tout formalisme de simulation stochastique, permettant de capitaliser les concepts de connaissance préalablement exprimé textuellement en évitant les pertes sémantiques, et un second, dans un langage plus formel, permettant d'effectuer des simulations stochastiques de systèmes globaux relatifs aux quatre axes d'intérêts par instanciation des concepts génériques,
- L'application et la validation de la démarche lors d'un passage à l'échelle sur un cas d'étude réel fourni par EDF.

Ces contributions sont développées au sein de ce mémoire sous la forme de six chapitres. Le chapitre 1 détaille le besoin industriel du point de vue EDF avant de généraliser ce besoin quel que soit le cadre applicatif. Les besoins de prise en compte de la dimension système sont alors mis en avant, et l'émergence de nouveaux KPIs inhérents à la prise en compte de cette dimension est justifiée. La décision de maintenance est également abordée et classifiée selon les différentes modélisations de la dégradation existantes. L'orientation vers une quantification des programmes de maintenance est identifiée comme le type d'approche adéquate pour répondre au problème industriel posé, et des prérequis concernant l'utilisabilité du langage de modélisation et de simulation adapté sont définis. Une démarche adaptée à la résolution d'un tel problème est alors proposée, constituée d'une phase de construction d'un modèle générique, puis d'une phase d'instanciation de ce modèle à un cas applicatif particulier et enfin d'une phase d'exécution de ce modèle par instanciation à un cas particulier. Cette démarche nécessite cependant d'identifier tout d'abord la connaissance nécessaire à la quantification des programmes de maintenance, puis les langages les plus adaptés à la simulation de cette connaissance.

En ce sens, dans un second chapitre, un état de l'art est réalisé sur les différentes connaissances ou concepts de connaissance métiers d'intérêt nécessaires à un modèle de quantification des programmes de maintenance. Cette connaissance est ensuite représentée sous la forme de concepts génériques, englobant ces aspects statiques et dynamiques, en langage naturel. De cette connaissance sont définis des prérequis relatifs à la puissance de modélisation du langage support à la simulation.

En reprenant les constats du chapitre 2 sur la puissance de modélisation nécessaire et ceux du chapitre 1 sur les besoins en termes d'utilisabilité du modèle, le troisième chapitre présente un état de l'art sur les principaux travaux traitant de cette quantification. Deux catégories d'approches sont mises en évidence : les approches se focalisant sur un langage de modélisation et de simulation en essayant

d'en exploiter au maximum les capacités de modélisation, et les approches proposant une étape de modélisation de la connaissance dans un langage naturel ou décorrélé de tout formalisme de simulation stochastique, avant le passage vers un langage de simulation stochastique. Cette deuxième catégorie d'approche est privilégiée et les langages d'intérêts sont identifiés par comparaison avec les prérequis précédemment identifiés. Cependant, les travaux existants ne proposent pas de modèle réutilisable et cohérent pour répondre au besoin. Ce chapitre conclut donc sur le choix de deux langages : le langage SysML pour la modélisation décorrélée de tout formalisme de simulation stochastique (modèle nommé SysML4CMPQ), et le langage ADF pour la simulation stochastique, et sur le besoin d'une étape de mapping permettant d'assurer la cohérence sémantique dans la chaine de modélisation.

Par rapport à cette contrainte de cohérence, le chapitre 4 développe une démarche permettant d'assurer la correspondance sémantique entre la connaissance identifiée chapitre 2 et les différents diagrammes SysML pour permettre la construction du modèle générique, à la fois statique et comportemental, SysML4CMPQ. Pour ce faire, cette connaissance est modélisée sous la forme de concepts de modélisation génériques à travers quatre types de diagrammes : les diagrammes de définition de blocs, les diagrammes paramétriques, les diagrammes de séquences et les diagrammes d'états.

Le chapitre 5 reprend certains des diagrammes précédents pour investiguer leurs liens avec un formalisme état transition : les automates de mode simulables ADF, afin de pouvoir effectuer des simulations stochastiques sur le modèle du système. Pour ce faire, nous établissons des correspondances (ou mapping) entre les éléments sémantiques du modèle SysML4CMPQ et les éléments sémantiques du langage ADF, puisque certaines vues SysML doivent être contraintes pour permettre l'obtention des KPIs identifiés. Ce chapitre met en évidence au final l'obtention d'une bibliothèque de modèles ADF atomiques génériques, et un guide d'assemblage et d'instanciation de ces modèles par un utilisateur afin de pouvoir modéliser un SP particulier et son SS.

Finalement, dans le sixième chapitre, les contributions formalisées aux chapitres précédents sont mises en œuvre sur un cas applicatif industriel permettant d'illustrer, suite à un passage à l'échelle, la faisabilité de l'approche globale. Le cas d'étude est un système réel de robinetterie ARE fourni par EDF.

La Figure 1 schématise la méthodologie proposée.

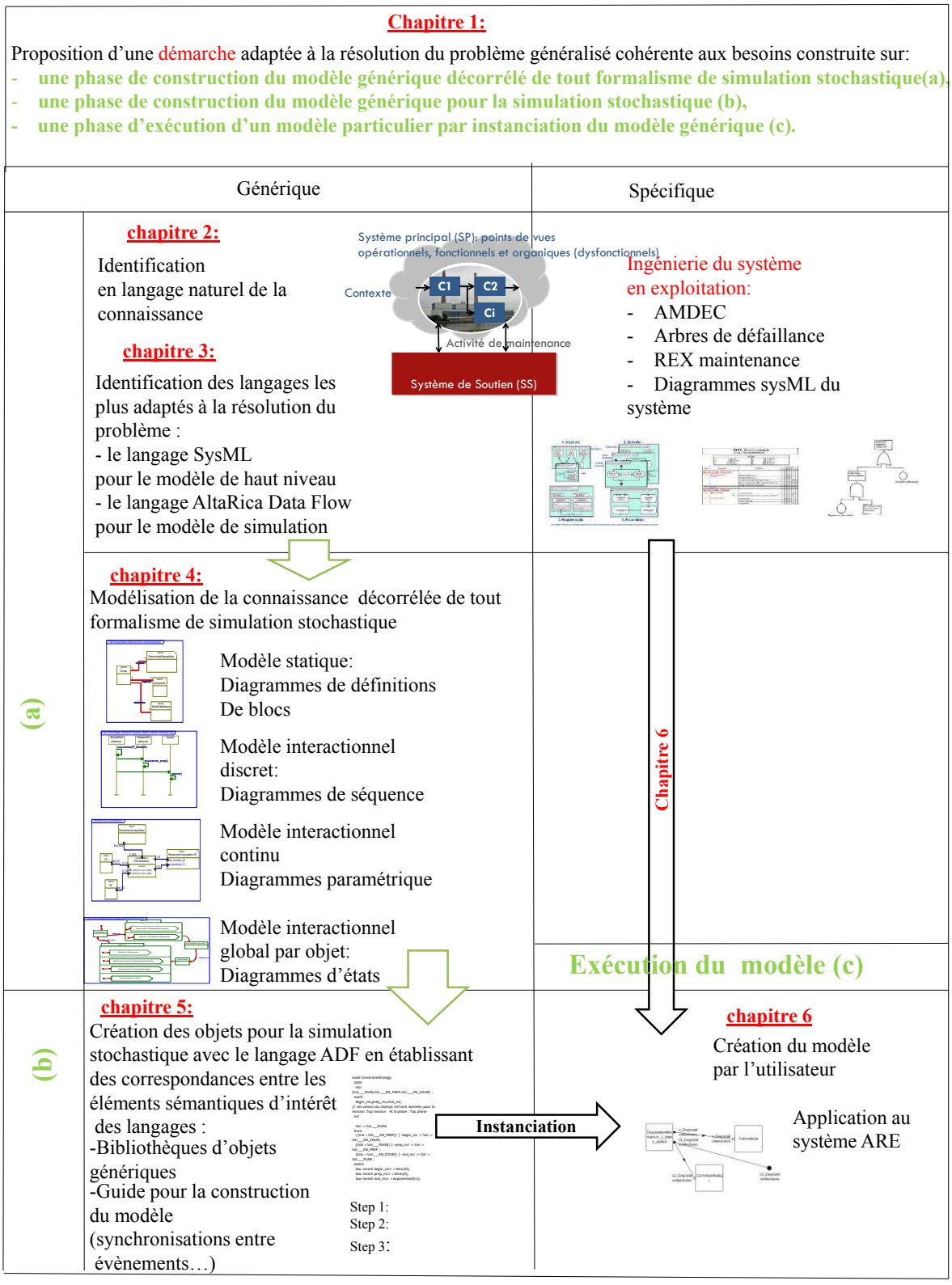


Figure 1 – Méthodologie proposée

Chapitre I. Du besoin industriel aux verrous scientifiques : la quantification des programmes de maintenance complexes comme un levier de la maitrise des performances

I.1. Un besoin industriel fort

I.1.1 Du point de vue d'EDF…

Dans le contexte actuel d'ouverture des marchés de l'énergie, les entreprises, dont l'Electricité de France (EDF), doivent améliorer leur compétitivité et donc leur productivité. Il s'agit de produire plus pour des couts moindres, et donc d'avoir une meilleure disponibilité des moyens de production tout en dépensant moins. En outre, la complexité croissante de ces systèmes opérants (système principal au sens de l'ingénierie système (IS) (AFIS, 2009)) devant répondre de manière complémentaire à des enjeux de sûreté, de sécurité et de durabilité fait du maintien en conditions opérationnelles (MCO) un enjeu crucial dans la maitrise des performances et exigences du SP (Alsyouf 2007). Ces activités supports à l'opérationnalité se matérialisent de manière concrète au niveau d'un système de soutien (SS), dont la fonction principale est la fonction maintenance. En effet, les couts de maintenance représentent environ 15 à 70% des couts totaux d'exploitation des systèmes de production (Bevilacqua & Braglia 2000).

En ce sens, EDF, dont les activités de maintenance réalisées sur parc nucléaire représentant un sixième du cout d'exploitation total en 2007 (EDF, 2007), a mis en place sur ses centrales en exploitation la démarche d'Optimisation de la Maintenance basée sur la fiabilité (OMF) (Despujols, 2001). Cette approche permet de choisir une organisation de l'activité de maintenance (définie par la norme NF EN 13306 (AFNOR 2001) comme l'« ensemble de toutes les actions techniques, administratives et de management durant le cycle de vie d'un bien, destinés à le maintenir ou à le rétablir dans un état dans lequel il peut accomplir la fonction requise ») compte tenu d'objectifs de disponibilité, de sûreté, et de couts. Cependant, une telle approche manque d'outils permettant de quantifier l'impact de cette activité sur ces ses systèmes (Zille 2009).

Aujourd'hui, EDF souhaite étendre cette démarche à une démarche AP-913- EDF. En effet, la méthode OMF présente des lacunes, puisque les études menées ont conduit (Zille 2009) :

- A faire évoluer la période des tâches de maintenance préventive, mais rarement à changer leur nature,
- A introduire peu de tâches de maintenance conditionnelle.

Ainsi, cette démarche de l'Institute of Nuclear Power Operators (INPO, 2001) (Figure 2) reprend un certain nombre d'éléments de la démarche OMF, en y ajoutant des tâches complémentaires nouvelles telles que l'analyse des causes de défaillance, la gestion de l'obsolescence ou la définition et l'utilisation d'indicateurs concernant les tâches d'organisation.

En effet, cette approche d' « asset management » repose sur six étapes :

- L'identification des matériels critiques des installations, avec l'identification des fonctions et des performances de chaque matériel et la caractérisation de leur importance vis-à-vis des grands objectifs (sûreté, production, préservation de l'environnement…),
- La surveillance en fonctionnement, avec l'analyse et la compréhension des mécanismes de dégradation, la mise en œuvre de tâches de maintenance conditionnelle et la recherche d'indicateurs de prédiction,
- L'action corrective, avec l'identification des défaillances rencontrées pour en déterminer les causes, les conditions d'amorçage et d'évolution, les améliorations potentielles, ainsi que la détection des obsolescences,
- L'amélioration continue de la fiabilité des matériels, de la même manière que la sélection des tâches de maintenance de la méthode OMF,
- La gestion du cycle de vie et des programmes sur le long terme, avec l'évaluation périodique de l'état de santé des matériels, le suivi et la prévision d'évolution des mécanismes de vieillissement et la mise en place de stratégies sur le long terme pour les maitriser,
- La mise en œuvre de la maintenance préventive, avec la rédaction et l'application des programmes de maintenance.

Cependant, comme pour la démarche OMF, l'AP-913 souffre d'un manque d'outils permettant de supporter les besoins liés à ces six étapes, résidant principalement :

- dans la façon de modéliser ces différents phénomènes, qu'ils aient déjà été présents dans la démarche OMF (sélection des tâches de maintenance…), ou qu'ils soient des nouveautés proposées par la méthode (surveillance en fonctionnement, action corrective…),
- dans l'évaluation des Key Performance Indicators (KPIs) liés à la simulation **a priori** (c'est-à-dire avant son application) du PBMP (Programme Basé sur la Maintenance Préventive), matérialisant l'impact des phénomènes précédemment énoncés sur les KPIs, encerclée dans la Figure 2.

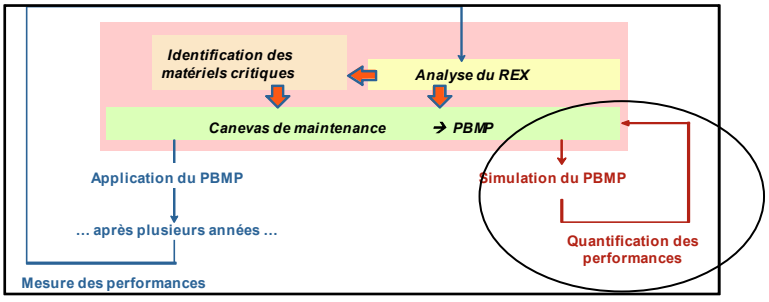


Figure 2 – Extrait démarche AP-913 [INPO, 2001]

L'évaluation par simulation de ces KPIs est d'autant plus importante que cette démarche doit pouvoir s'appliquer à moyen ou à long terme, à des systèmes de grande dimension, ou systèmes complexes (un programme de maintenance pour un système sujet à cette démarche se compose d'environ 1000 tâches réparties), non critiques en termes de sûreté, et dont les arrêts de tranche coutent une somme de l'ordre du million d'euros en moyenne.

EDF a préalablement défini les KPIs liés à ces six étapes et à la simulation du PBMP (Zille 2009). Ils sont les suivants :

- La disponibilité du système sur une période donnée,
- La qualité des produits et services fournis,
- Les couts de MCO,
- La sûreté des personnes, de l'environnement et des biens.

Ces KPIs fournissent un support complet au décisionnaire de maintenance afin d'appliquer le programme de maintenance préventive fournissant le meilleur compromis en termes d'objectifs technico-économiques au système. Les types de simulation envisagés par EDF sont par exemple :

- La comparaison selon ces KPIs de plusieurs programmes de maintenance pour un système donné sur une période allant de un à dix ans (simulation du PBMP),
- La comparaison selon ces KPIs de plusieurs contextes d'utilisations d'un système pour un programme de maintenance donné sur une période allant de un à dix ans,
- La comparaison selon ces KPIs de composants ou sous-systèmes de fiabilité variables pour un programme de maintenance donné sur une période allant de un à dix ans, dans le cas de systèmes en conception.

Comme point d'entrée de ces simulations, EDF dispose de plans physiques du système, de résultats d'études AMDEC et/ou HAZOP, d'arbres de défaillances, ainsi que d'un Retour d'EXpérience (REX) parfois peu fourni. Ainsi, il est nécessaire pour la simulation de pouvoir s'adapter à la granularité des informations disponibles données par les outils précédents, et donc de pouvoir évaluer les KPIs selon les points de modélisation définis.

Ces études (arbres de défaillances, études AMDEC…) sont principalement réalisés par des responsables de maintenance ou de production quel que soit leur niveau décisionnel (à un point de vue opérationnel comme à un point de vue stratégique) et quelles que soient leurs compétences informatiques. Ce type d'utilisateur est donc amené à également utiliser l'outil d'évaluation par simulation des KPIs. De plus, les types de simulations envisagées étant destinées à effectuer des comparaisons, il est essentiel pour l'outil de simulation support de pouvoir réduire l'effort de modélisation à chaque étude. Le type d'utilisateur destiné à utiliser cet outil de simulation conduit à définir une granularité dans les KPIs, et plus spécifiquement des KPIs de couts. En effet, étant donné les périodes des simulations envisagées, des paramètres liés à des facteurs impactant les couts non seulement internes mais aussi externes doivent être envisagés (Senechal 2004). Par rapport au contexte industriel EDF de cette thèse et aux pratiques de cout qui sont mises en œuvre par le département concerné par ce travail, ces facteurs ne sont pas pris en compte dans le calcul des KPIs lors des simulations du PBMB.

Ainsi, cet outil de simulation doit donc permettre :

- la réduction de l'effort de modélisation à chaque étude,
- via une interface conviviale par exemple, une utilisation par un utilisateur aux compétences métiers (maintenance, ingénierie système…) ne disposant pas spécialement de compétences dans les outils de la sûreté de fonctionnement.

Ces deux prérequis seront par la suite englobés dans le prérequis plus global d'utilisabilité de l'outil de simulation support.

Cette thèse s'inscrit dans ce contexte d'évolution de la démarche OMF avec comme objectif principal de faire évoluer l'outillage actuel d'EDF vers un outillage permettant :

- la modélisation du comportement de systèmes globaux en prenant en compte non seulement les nouvelles contraintes liées à cette évolution (nouvelles tâches de maintenance…), mais aussi la réduction de l'effort de modélisation et l'utilisabilité du modèle,
- la simulation de leur PBMP pour l'évaluation des KPIs.

En ce sens cette thèse a bénéficié des moyens déployés au sein des projets DEPRADEM et DEPRADEM 2 du GIS 3 SGS pour s'attaquer aux objectifs précédemment mentionnés.

Ces différents besoins, relatifs aux KPIs, aux types de simulation et à l'utilisation de l'outil requis constituent le cahier des charges d'EDF afin d'acquérir une solution compatible avec cette évolution de la démarche. Ce cahier des charges est tout à fait généralisable à d'autres domaines d'application puisque la problématique de modélisation et de simulation du PBMP est à ce jour un véritable challenge pour de nombreuses entreprises dans un enjeu de pérennité et de profit de l'outil de production.

I.1.2 … Comme pour l'industrie en général

D'un point de vue théorique, un KPI peut est défini comme « une information devant aider un acteur, individuel ou plus généralement collectif, à conduire le cours d'une action vers l'atteinte d'un objectif ou devant lui permettre d'en évaluer le résultat » (Lorino 2001). Le KPI est donc construit par l'acteur d'un processus, ou l'utilisateur d'un outil tel que l'outil de simulation requis. Ainsi, afin d'identifier les KPIs d'intérêts dans notre contexte, il est nécessaire de s'intéresser tout d'abord aux KPIs liés à des systèmes de production en général (Pérès & Noyes 2003), (Sénéchal, 2004).

Les plus traditionnels sont les couts d'exploitation, intégrant les bénéfices liés à la production et les dépenses réalisées pour effectuer les tâches de maintenance, la disponibilité du système et plus particulièrement ses durées d'indisponibilité fortuite, liée aux pannes, et d'indisponibilité programmée, résultant des opérations de maintenance, et donc les requis d'EDF à ce niveau.

Sur un axe complémentaire plus centré sur l'activité de maintenance, (Parida 2006) a effectué un état de l'art identifiant, sur un ensemble de travaux relatifs à la maintenance et à la sûreté de fonctionnement, les différents KPIs d'intérêts. Après une synthèse de ces travaux, l'auteur met en valeur l'aspect multicritère de la décision de maintenance, et classe les principaux indicateurs selon 7 familles de critères en interrelation :

- Indicateurs de satisfaction client : nombre de plaintes, nombre de retours… ,
- Indicateurs de cout : couts de maintenance et de production,
- Indicateurs d'équipements : nombre d'arrêts fortuits et planifiés…,
- Indicateurs de tâches de maintenance : nombre de tâches planifiées, nombre de tâches non planifiées…,
- Indicateurs d'amélioration interne selon le retour d'expérience,

- Indicateurs de sûreté et d'environnement : nombre d'accidents, légaux ou non…,
- Indicateurs de satisfaction des employés,

D'autres auteurs insistent sur la prépondérance de certains de ces KPIs, et notamment ceux liés aux couts, souvent utilisés pour agréger d'autres indicateurs (Proverbs & Holt 2000).

Ces différents critères s'évaluent et ont un sens différent selon l'horizon de la décision et la nature du système considéré. Dans notre contexte de la simulation de PBMP **a priori**, les aspects liés aux critères tels que l'amélioration continue, la satisfaction des employés, et la satisfaction client ne peuvent pas être pris en compte, puisque ceux-ci s'évaluent a posteriori grâce au REX. Par ailleurs, ils touchent au facteur humain et s'avèrent particulièrement difficiles à évaluer. De même, certains aspects sociétaux et environnementaux s'évaluant plutôt a posteriori paraissent difficiles à prendre en compte, comme par exemple le nombre de plaintes enregistrées.

Ainsi, parmi ces sept catégories d'indicateurs ou KPIs, trois catégories seulement sont applicables dans notre contexte : la catégorie des indicateurs relatifs aux équipements, la catégorie des indicateurs relatifs aux couts, et la catégorie des indicateurs relatifs à la maintenance. La liste suivante reprend ces catégories ainsi que les principaux KPIs les composant :

- Indicateurs ou KPIs **relatifs aux équipements** :
    - Disponibilité,
    - Taux de production,
    - Qualité,
    - Nombre d'arrêts,
- Indicateurs ou KPIs **relatifs aux couts** :
    - Cout de la maintenance,
    - Cout de la production,
- Indicateurs ou KPIs **relatifs à la maintenance** :
    - Tâches de maintenance planifiées (nombre réalisé, retards…),
    - Tâches de maintenance non planifiées (nombre réalisé…).

La vision actuelle tend à considérer ces KPIs dès la phase de conception des systèmes (Lee, 1995) (AFIS, 2009) et à les évaluer de manière conjointe. De telles approches permettent de faciliter la prise de décision tout au long du cycle de vie du système (par exemple re-conception du SP en fonction de contraintes du SS).

De plus, chaque système maintenu possède ses spécificités, et donc des KPIs particuliers. Par exemple, (Monnin et al. 2011) s'intéressent à des indicateurs représentatifs de la régénérabilité de systèmes de combat, ou (Betous-Almeida & Kanoun 2004a) s'intéressent à la disponibilité opérationnelle de système informatiques formés par des composants aussi bien virtuels que physiques. Dans le contexte d'EDF, le niveau de sûreté du système et les risques encourus pour les personnes, l'environnement et les installations est identifié comme un type de KPI complémentaire à évaluer. Par ailleurs, l'évaluation a priori de ces KPIs se fait par élicitation d'avis d'experts ou par analyse du REX. Si de nouvelles techniques ont vu le jour pour prendre en compte les avis d'experts de manière floue ou possibiliste (Ribeiro 1996), les techniques probabilistes continuent d'être les plus utilisées en

entrée des études pour l'évaluation de ce type de KPIs. Ainsi, il a été établi que la **simulation stochastique** (Bouissou, 2007) est nécessaire pour l'évaluation de ces KPIs liés à la sûreté de fonctionnement des systèmes. Cette prérequis fait également partie de l'prérequis plus globale d'utilisabilité de l'outil à fournir à la fin de ce travail.

Outre ces KPIs, l'autre principal besoin lié à l'implantation de cette démarche évoluée est la modélisation des divers phénomènes liés aux six étapes de la démarche. De par le cadre de sûreté imposé par EDF, ces différents points de vue peuvent se généraliser. Cependant, pour disposer d'un outil permettant la simulation de différents cadres applicatifs relatifs à d'autres milieux industriels, il est nécessaire d'identifier ces points de vue (modélisation fonctionnelle et dysfonctionnelle…) de manière générique, et en cohérence avec les KPIs ci-dessus. Cette identification est l'objet du chapitre 2, et fait émerger un nouveau prérequis sur l'outil d'évaluation des KPIs par simulation : sa puissance de modélisation. Par ailleurs, si EDF pense plutôt utiliser cette démarche pour simuler des PBMP sur des systèmes en exploitation, on peut considérer la simulation dès la conception de systèmes dans un contexte plus global d'Ingénierie Système. En effet, il y a aujourd'hui un manque d'outils de vérification des exigences, et donc d'évaluation des KPIs à la conception. De plus, dans ce contexte plus global, les prérequis liées à l'utilisabilité de l'outil restent des contraintes fortes dans le cadre de la gestion de la complexité de tout système (Bouissou, 2007).

Par cette phase de généralisation, ce travail est un travail de recherche et non pas un travail d'ingénierie focalisé sur la recherche d'une solution à un problème particulier, et il cherche à fonder une méthodologie générique exploitable pour diverses classes d'applications. Cette généralisation a fait apparaitre la définition de nouveaux KPIs, faisant émerger un prérequis lié à la puissance de modélisation de l'outil d'évaluation des KPIs par simulation. En ajoutant ce prérequis à ceux précédemment identifiés comme portant sur l'utilisabilité de cet outil, des grandes lignes se dégagent pour l'orientation d'une méthodologie pour la construction d'un tel outil. Cependant, avant de définir plus précisément cette puissance de modélisation, il est tout d'abord nécessaire de positionner la simulation du PBMP par rapport à certaines définitions de l'activité de maintenance en général, pour identifier le type de décision globale à prendre et les démarches traditionnellement employées pour supporter ce type de décision.

I.2. L'activité de maintenance

I.2.1 La maintenance : des objectifs à différents niveaux de l'entreprise

D'après la définition de la maintenance de (AFNOR 2001), la décision de maintenance peut intervenir aussi bien aux niveaux « techniques, administratifs et de management » de l'entreprise. Ainsi, différentes couches décisionnaire de celle-ci sont concernées par l'activité de maintenance au sens large.

La politique de maintenance, tout d'abord, est fixée à un niveau très haut dans l'entreprise. Selon la philosophie de celle-ci, elle peut s'assimiler aux objectifs de maintenance définis dans (AFNOR 2001) comme « buts fixés et acceptés pour les activités de maintenance ». Par exemple, une entreprise peut miser sur la sûreté de ses installations, comme privilégier la production quitte à encourir des accidents. Une fois cette politique de maintenance définie, alors des stratégies de maintenance peuvent être définies sur les différents biens de l'entreprise.

Une stratégie de maintenance se définit dans (AFNOR 2001) comme « la méthode de management utilisée pour atteindre les objectifs de maintenance ». Elle consiste donc en la définition du portefeuille d'activités de la production de maintenance et conjointement à organiser structurellement le système de conduite et les ressources productives pour y parvenir. Ces stratégies se matérialisent donc par l'application au niveau opérationnel d'une politique de maintenance

Différentes politiques de maintenance classiques existent, et peuvent être classifiées comme représenté Figure 3.

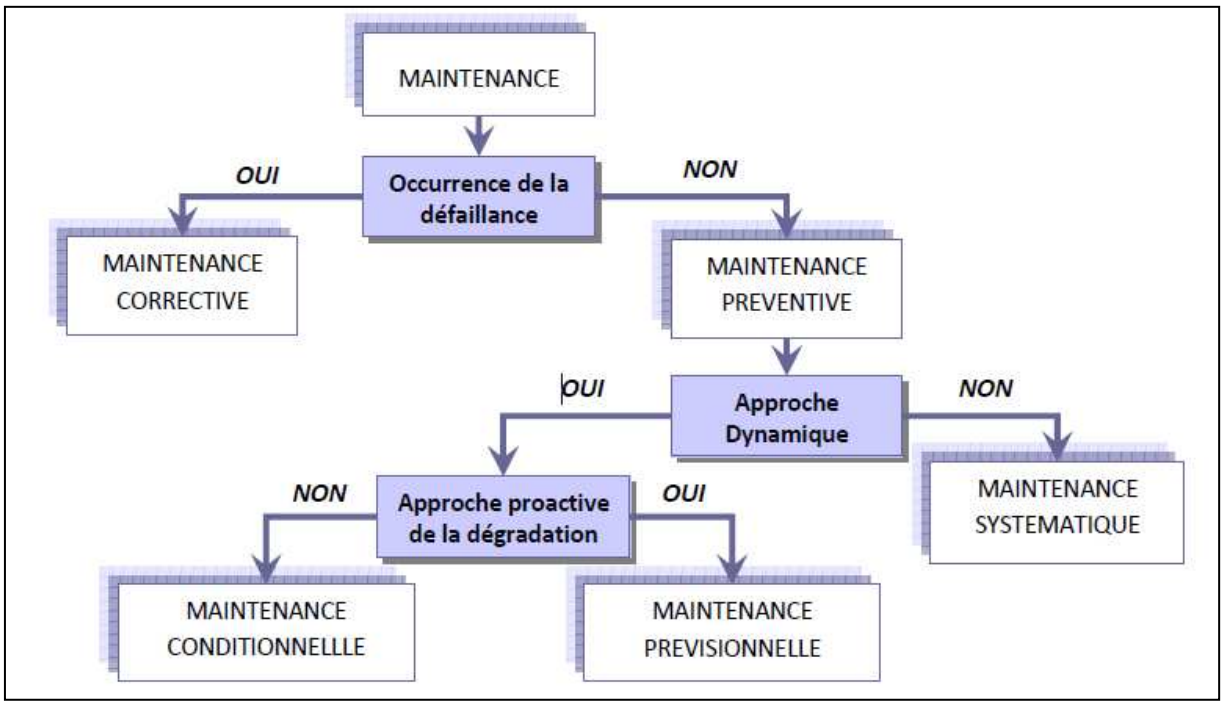


Figure 3 – Description des approches de maintenance

D'après (AFNOR, 2001), les politiques et stratégies de maintenance peuvent se décrire selon différents types d'approches :

- Maintenance corrective : Maintenance exécutée après détection d'une panne et destinée à remettre un bien dans un état dans lequel il peut accomplir une fonction requise,
- Maintenance préventive : Maintenance exécutée à des intervalles prédéterminés ou selon des critères prescrits et destinée à réduire la probabilité de défaillance ou la dégradation du fonctionnement d'un bien,
- Maintenance systématique : maintenance préventive exécutée à des intervalles de temps préétablis ou selon un nombre défini d'unités d'usage mais sans contrôle préalable de l'état du bien,
- Maintenance conditionnelle : maintenance préventive basée sur une surveillance du fonctionnement du bien et/ou des paramètres significatifs de ce fonctionnement intégrant les actions qui en découlent,
- Maintenance prévisionnelle : maintenance conditionnelle exécutée en suivant les prévisions extrapolées de l'analyse et de l'évaluation de paramètres significatifs de la dégradation du bien.

A ces différentes politiques de maintenance classiques, nous ajouterons les politiques de maintenance opportunes ou opportunistes, se composant d'actions de maintenance opportunes. Les définitions ci-dessous sont issues de (E. Thomas 2009).

Une action de maintenance est qualifiée d'action opportune de première espèce si elle s'effectue durant un arrêt de production.

Une action *C*-opportune de seconde espèce est une action de maintenance préventive réalisée, en coïncidence avec une action sur le composant *C*, sur un composant dégradé *D* qui vérifie au moins l'une des conditions suivantes :

- le composant *D* est proche de *C* au sens d'un critère de proximité (physique ou géographique) rigoureusement défini. Dans le cas où des enceintes (par exemple de confinement) existent dans le système, les deux composants sont situés dans la même enceinte,
- la gamme de consignation – déconsignation propre à cette action est identique à la gamme de consignation – déconsignation mise en oeuvre pour l'action de maintenance sur le composant *C,*
- un flux de matière ou d'énergie relie l'un des composants à l'autre,
- les deux composants participent à une même fonction identifiée du système,
- les outils et compétences nécessaires à l'accès au composant *C* permettent également d'accéder au composant *D,*
- si les deux actions de maintenance considérées sont opportunes de première espèce, et si les ressources disponibles permettent de réaliser ces deux actions simultanément, alors leur arrêt de production associé est le même.

Ces actions de maintenance peuvent s'appliquer à la fois en complément à des politiques conditionnelles (Savsar 2006) ou prévisionnelles.

L'application de ces différentes politiques à un niveau opérationnel se réalise selon un plan ou un programme de maintenance, défini comme « un ensemble structuré de tâches qui comprennent les activités, les procédures, les ressources et la durée nécessaire pour exécuter la maintenance ».

Ainsi, l'activité de maintenance implique des décisions à plusieurs niveaux de l'entreprise, du niveau stratégique au niveau opérationnel en passant par le niveau tactique. Il est donc nécessaire d'investiguer les différents types de décision agissant à ces différents niveaux, et de les comparer aux besoins actuels d'EdF (décision à long et moyen terme, simulation d'un **programme** de maintenance).

I.2.2 La décision de maintenance

Dans le cadre du projet DEPRADEM 1, (Monnin, 2009) a réalisé une classification en trois pôles des approches scientifiques relatives à la maintenance selon leur horizon de décision: l'optimisation de stratégies de maintenance, la quantification des programmes de maintenance et la rationalisation de la maintenance.

a. **L'optimisation de stratégies de maintenance** a pour objectif de déterminer les règles de décision qui répondent au mieux, voire optimisent, les objectifs de maintenance. Elle s'appuie sur des modèles de couts ou de performances qui permettent de déterminer les niveaux de dégradation associés aux actions de maintenance et les intervalles entre ces actions optimisant le cout ou les

performances. Ces travaux s'appuient principalement sur une optimisation analytique d'une dégradation modélisée sous la forme de processus de renouvellement permettant de prendre en compte les diverses actions de maintenance sur cette dégradation, rendant ainsi difficile leur application à des systèmes multi-composants. D'après (Dekker 1996), ces travaux sont présents en nombre majoritaire parmi les communautés liées à la maintenance. Au niveau national, les laboratoires traitant cette problématique sont par exemple le LM2S à l'Université Technologique de Troyes. Au niveau international, de nombreux surveys ont été réalisés, comme par exemple ceux de (Cho & Parlar, 1991) sur les approches de d'optimisation de stratégies sur des systèmes multi-composants ou (Vasquez Flores & Feldman, 1989) sur la complexification des stratégies sur des systèmes mono-composants. Ce type de décision est cependant plutôt adapté au fournisseur du composant qu'à la prise de décision au niveau tactique dans l'entreprise client, étant donné les hypothèses simplificatrices réalisées sur les dégradations des composants (dégradation des composants indépendantes des facteurs extérieurs…).

b. **La quantification des programmes de maintenance** a pour but de comparer l'application de différentes politiques sur un même système ou encore d'évaluer un programme éprouvé sur un nouveau système. Autrement dit, il s'agit, au niveau opérationnel, d'un ensemble de politiques de maintenance appliquées à un ensemble d'éléments (composants, sous-systèmes).

Cette décision est supportée par des modèles visant à mesurer l'impact de l'application d'un ensemble de règles de décision définissant le programme. Les travaux liés à cette problématique sont principalement issus de collaboration entre les milieux industriels (par exemple Total ou EDF) et universitaires : (Boiteau et al. 2006) (Bouissou 2005) (Ramesh et al. 1999) (Betous-Almeida & Kanoun 2004a). Schématiquement, la Figure 4 illustre un programme de maintenance, assimilable au PBMP.

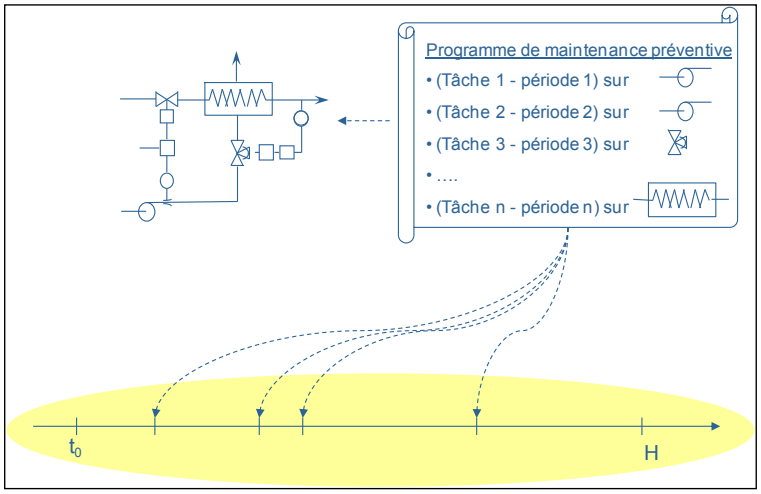


Figure 4 – Illustration d'un programme de maintenance

c. **La rationalisation des actions de maintenance** a pour objectif de rendre dynamique la prise de décision. Le choix de l'action de maintenance à mettre en œuvre est basé sur l'extrapolation de l'évolution de l'état d'un matériel ou d'un système. Il s'agit ici de développer des modèles pour évaluer l'état courant du matériel, identifier ses modes de dégradation en cours et leur évolution pour évaluer une durée de vie résiduelle. Ces travaux sont notamment traités au-delà du CRAN par la

communauté PHM (Prognosis and Health Management) et notamment les auteurs J. Lee (Zhou et al. 2007) ou A.K.S. Jardine (Jardine et al. 2006).

Ainsi, ces trois types de décision sont différentiables selon 2 principaux critères : la granularité de la maintenance appréhendée, et l'horizon de la décision (cf. Figure 5, adaptée de (Medina Oliva 2011)).

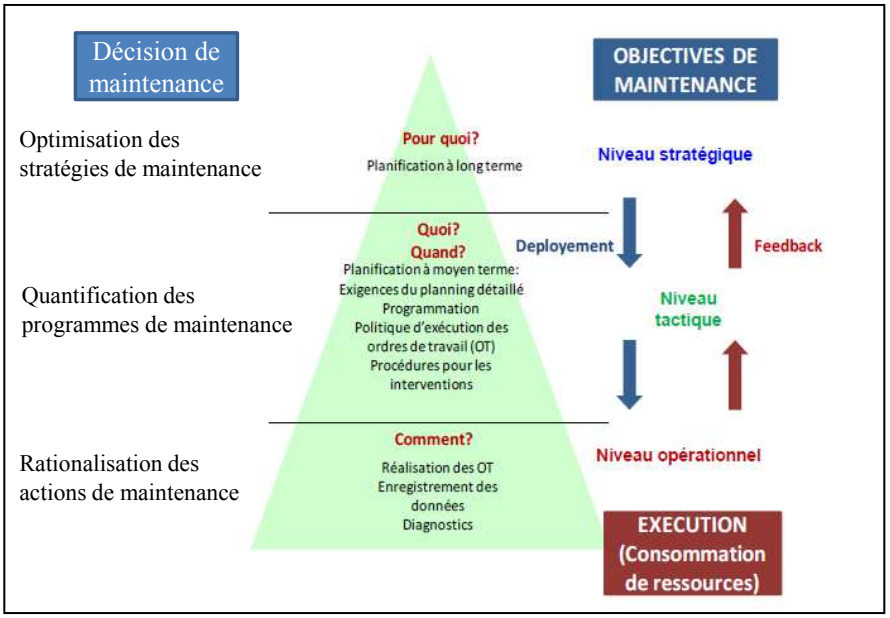


Figure 5 – La décision de maintenance en fonction des objectifs

En synthèse, la rationalisation concerne la prise de décision sur des actions de maintenance, et est destinée à une décision à court terme. Dans la littérature PHM, l'action de maintenance n'y est que rarement traitée. (Jardine et al. 2006) (Khanh Le Son, 2011) préfèrent en effet se focaliser sur l'évaluation de durée de vie résiduelle. L'optimisation est quant à elle destinée à être appliquée à long terme dans l'entreprise. Cependant, elle ne s'applique tout au plus qu'à des systèmes composés de peu de composants étant donné qu'elle s'adresse à de simples stratégies. La quantification des programmes de maintenance est quant à elle plus adaptée à notre problématique. En effet, en plus de pouvoir intégrer une grande variabilité dans le modèle métier proposé, elle peut s'appliquer à moyen et long terme et permet également d'adresser des programmes de maintenance, sans toutefois identifier des solutions de maintenance optimales. Elle englobe donc la problématique de simulation d'un PBMP exigé par l'AP-913. Cependant, les travaux relatifs à la quantification des programmes de maintenance sont parfois identifiés comme des travaux visant à l'évaluation des « stratégies de maintenance sur des systèmes multi-composants » (Zille 2009). Or cette définition apparait réductrice en regard du problème, et peut renvoyer à la notion d'optimisation de la maintenance. Nous allons donc chercher à fonder une définition pour la quantification des programmes de maintenance, voir à faire évoluer cette terminologie, afin d'éliminer toute ambiguïté.

Un programme de maintenance implique la notion de mise en œuvre au niveau opérationnel des politiques sur un ensemble d'éléments (composants, sous-systèmes). Ce terme souligne ainsi mieux les besoins en termes de modélisation des tâches de maintenance ou du SS (ressources humaines et matérielles). En outre, le terme multi-composant parait peu cohérent avec la terminologie actuelle des industries faisant état de systèmes complexes. Nous préfèrerons donc parler de « **programme de**

maintenance complexe », qui englobe la notion de politique de maintenance sur des systèmes multi-composants.

Cette prépondérance des interactions entre SP et SS dans le contexte des programmes de maintenance complexes conduit à se questionner sur la manière d'évaluer les KPIs liés à ce type de décision. Dans le cadre de l'IS, lors de la conception d'un système complexe, l'IS envisage le SS à mettre en place pour maintenir le SP, et donc de concevoir ces deux systèmes de manière parallèle (Figure 6). En effet, dans le contexte d'un système construit selon une démarche d'IS, les modèles du SS sont censés exister au même titre que ceux du SP.

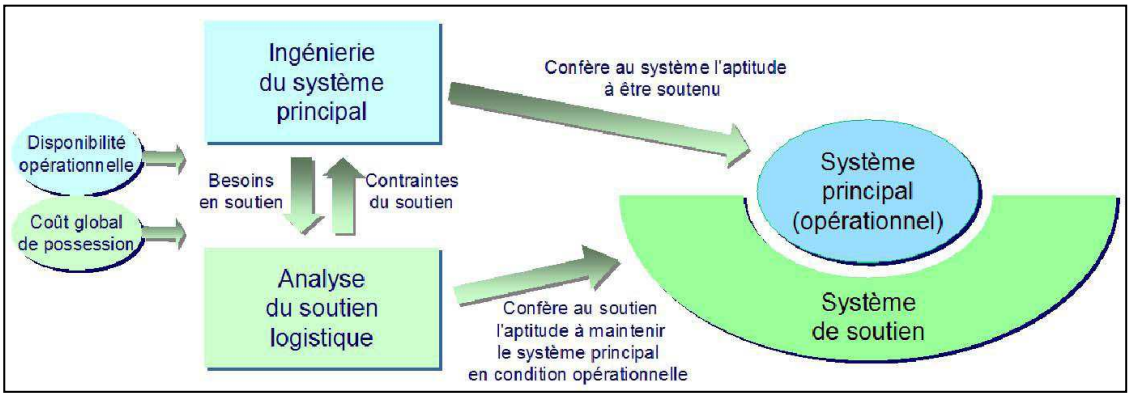


Figure 6 - Approche du soutien logistique intégré (AFIS, 2009)

Dans ce contexte, il est nécessaire de pouvoir modéliser l'adéquation entre le SP et le SS. Elle se fait, dans le contexte de la CMPQ, par les stratégies de maintenance, qui permettent de faire interagir ces deux sous-systèmes, à travers l'application de programmes et de politiques de maintenance à un niveau opérationnel, permettant ainsi l'évaluation conjointe des différents KPIs.

Cette évaluation peut être conduite de deux façons (Betous-Almeida & Kanoun 2004b):

- L'évaluation ordinale : destinée à identifier, classer et ordonner les défaillances ou les méthodes et les techniques mises en œuvre pour les éviter,
- L'évaluation probabiliste, englobant les modèles statiques (ou structurels) et les modèles dynamiques (ou comportementaux) destinée à évaluer en termes de probabilités le degré de satisfaction des KPIs/prérequis, soit :
    - Par simulation : approches permettant le calcul approché des indicateurs de performance,
    - Par résolution analytique : approches permettant le calcul exact des indicateurs de performance,
    - Par expérimentation : approches s'appuyant sur les observations du système en cours d'exécution.

Etant donné la grande complexité d'une telle étude, la résolution par simulation parait donc plus adaptée (A. Rauzy 2002) (Bouissou, 2007) à cette évaluation des KPIs. Ainsi, le terme de « **quantification** », défini par le Larousse comme « l'action de traduire quelque chose en une quantité mesurable » parait plus adapté à notre problématique que le terme « évaluation », défini par cette même source comme « l'action de déterminer la valeur de quelque chose ». En effet, quantifier un programme de maintenance renvoie une dimension de modélisation absente dans l'évaluation des

programmes de maintenance. De plus, ce terme sous-entend également la quantification de l'impact des programmes de maintenance, sur les autres KPIs plus liés au SP (disponibilité, performances…), tandis que le terme d'évaluation peut renvoyer à la notion d'optimisation (évaluation analytique permettant d'optimiser les périodes des politiques de maintenance préventives en termes de couts), ainsi qu'à la notion d'interprétation des phénomènes quantifiés au sens de (Lorino 2001).

Dans ce sens, et en accord avec la définition proposée par (Monnin, 2009), nous proposons la définition suivante pour la **quantification des programmes de maintenance complexes (CMPQ)** : Décision de maintenance ayant pour but, à partir d'une modélisation aussi exhaustive et réaliste que possible du comportement d'un système principal ciblé, de son contexte, de son système de soutien, de l'activité de maintenance et de leurs relations, de quantifier les performances globales du système dans son ensemble, à savoir les KPIs identifiés, pour permettre le choix du programme de maintenance le plus adéquat à appliquer au système cible.

Ainsi, la modélisation du système principal ciblé et de ses systèmes contributeurs constitue la pierre angulaire de la réalisation de toute étude de CMPQ. De par la complexité sous-jacente à cette modélisation, il est nécessaire de s'intéresser à la démarche préconisée par l'IS pour la réalisation de ce type d'étude qui peut se décomposer en trois phases (AFIS, 2009) :

- Choix des KPIs à évaluer (1),
- Construction du modèle : cette étape a pour but de modéliser le comportement du système global (SP et ses systèmes contributeurs) en fonction des KPIs à évaluer (2),
- Traitement du modèle : évaluation des KPIs (3).

Dans le contexte de la CMPQ, il est nécessaire d'adapter cette démarche, puisqu'une telle étude implique d'être en phase avec la définition proposée pour la CMPQ, et notamment avec l'important effort de modélisation à réaliser pour chaque étude. L'étape (2) doit en effet dans notre contexte être précédée d'une étape (1') devant permettre d'aboutir à la **construction d'un modèle générique**, conforme aux exigences de modélisation. Ce modèle générique peut alors être instancié, réduisant ainsi considérablement l'effort de modélisation à chaque étude, pour permettre la création d'un modèle de système global particulier. L'étape (2) devient alors la construction **par instanciation** du modèle générique issu de l'étape (1'), d'un modèle particulier relatif à un cas d'étude donné. Le modèle disponible à l'étape (2) doit également être simulable pour permettre la réalisation de l'étape (3) d'évaluation des KPIs. Le problème du niveau formalisation de la connaissance (à l'étape (1') ou à l'étape (2)) est alors mis en avant. De manière parallèle, il est bien sur essentiel de s'assurer que le modèle utilisé à chaque étape est correct du point de vue du concepteur comme du point de vue de l'utilisateur. Ceci fait appel aux notions de vérification et de validation.

Les définitions ci-dessous sont issues de (ISO 8402). La vérification est la confirmation par examen et apport de preuves tangibles que les exigences spécifiées ont été satisfaites. Elle répond à la question « construisons-nous correctement le modèle?»

La validation est la confirmation que par examen et apport de preuves tangibles que les exigences particulières pour un usage spécifique sont satisfaites. Elle répond à la question « construisons nous le bon modèle ? ».

Ainsi, trois verrous scientifiques majeurs sont à résoudre pour l'application de cette démarche système à la CMPQ :

- Quelle est la connaissance générique nécessaire à la CMPQ ?
- A quel niveau (générique ou instancié) faut-il formaliser de la connaissance ? Comment le faire tout en répondant aux prérequis d'utilisabilité et de puissance de modélisation ?
- Comment vérifier et valider le modèle ?

I.3. Conclusion

A partir du besoin industriel et des prérequis en termes d'utilisation de l'outil envisagé par EDF afin de supporter la mise en place de la méthode AP-913, nous avons positionné notre problématique par rapport aux décisions de maintenance existantes. Ainsi, la quantification des programmes de maintenance complexes (CMPQ) a été fondée, et elle permet d'appréhender au mieux l'ensemble de ces prérequis. En effet, elle permet une représentation dynamique de systèmes complexes en vue de simuler son fonctionnement en y intégrant sa maintenance, son système de soutien, et son profil d'exploitation.. La CMPQ nécessite l'appréhension de quatre étapes successives : une étape (1) de définition des KPIs, une étape (1') de construction d'un modèle générique instanciable, une étape (2) de construction d'un modèle particulier par instanciation du modèle générique construit à l'étape (1'), et une étape (3) de simulation du modèle

Trois principaux verrous scientifiques sous-jacents à ces étapes ont pu être identifiés. Le premier consiste à identifier les connaissances d'intérêt pour la CMPQ à partir des différents travaux et normes existants. Le second consiste à modéliser cette connaissance pour permettre la capitalisation (permettant l'instanciation) et l'exécution du modèle. Enfin, il se pose la question de vérification et de validation du modèle. Ainsi, un ou plusieurs langages doivent être sélectionnés selon les prérequis liées aux objectifs de l'outil final :

- La capacité à modéliser la connaissance relative à la CMPQ (puissance de modélisation du langage, vérification et validation des modèles),
- La capacité d'être exploité par un utilisateur sans connaissance de ce langage,
- La capacité à effectuer des simulations stochastiques.

En ce sens, le chapitre 2 propose une démarche d'identification de la connaissance métier d'intérêt permettant de supporter l'évaluation des indicateurs relatifs à la CMPQ afin de traiter partiellement l'étape (1') de construction du modèle selon quatre axes de modélisation. Le prérequis lié à la puissance sera alors précisé pour permettre la définition de prérequis plus fins sur les phénomènes à modéliser.

Ainsi, des critères complémentaires à l'utilisabilité du langage pourront être mis en évidence, et permettront le choix du ou des langages de modélisation les plus adéquats pour capitaliser et simuler cette connaissance. Ce choix sera discuté dans le chapitre 3.

Chapitre II. L'identification de la connaissance métier pour la quantification des KPIs pour la CMPQ

II.1. Introduction

La connaissance générique métier (permettant de traiter partiellement l'étape (1') de la démarche) pour la CMPQ doit être identifiée pour permettre l'évaluation des KPIs industriels d'intérêt et intégrant les besoins en modélisation d'une étude de CMPQ.

Pour ce faire, nous allons tout d'abord identifier 4 axes principaux permettant d'organiser cette connaissance. Ensuite, une démarche d'identification de la connaissance est proposée, se séparant en trois principales étapes : l'identification indépendante des différentes connaissances relatives à chaque axe, l'identification des relations structurelles entre ces connaissances, et enfin l'identification de la dynamique de ces interactions. Chacune de ces étapes est détaillée en mettant en évidence les relations établies sur ces connaissances, complétées par des propositions cohérentes. En parallèle de ces étapes, une expression textuelle de cette sémantique statique et comportementale rattachée à chaque connaissance est proposée afin de faciliter le passage depuis cette connaissance alors représentée sous la forme de **concepts** vers un langage de modélisation. Chaque concept de connaissance peut être ensuite capitalisé (cf. chapitre 4 et 5) indépendamment ou non des autres concepts et être considéré dans le modèle de CMPQ.

Ainsi, un ensemble de connaissances de nature parfois très différente (statique, dynamique, quantitative, qualitative) est défini pour la CMPQ. De cette connaissance, des critères décisifs dans le choix des langages sont identifiés.

II.2. Identification du domaine de connaissance générique

Afin d'identifier le domaine de connaissance requis pour la CMPQ et permettant la construction du modèle générique, il est nécessaire de se baser sur les trois grandes catégories de KPIs (couts, maintenance et équipements) définies au chapitre 1. Il est également important de respecter les limites définies permettant de respecter la granularité de ces KPIs (non prise en compte de l'inflation dans le calcul des couts par exemple).

Nous allons dans un premier temps nous focaliser sur la manière de décrire un système dans son ensemble, et d'identifier ses concepts d'intérêts en vue de la CMPQ. En reprenant les définitions de l'IS, la Figure 6 décrit tout système en deux principaux sous-systèmes : le Système Principal ou SP et le Système de Soutien ou SS. Nous avons identifié dans le chapitre précédent les stratégies de maintenance comme étant le lien permettant de faire interagir ces 2 sous-systèmes de manière conjointe, et donc de quantifier les KPIs liés à la CMPQ en utilisant la connaissance de ces trois entités (SS, SP et activité de maintenance. Le premier grand axe de connaissance d'intérêt pour la CMPQ est donc formé par **l'activité de maintenance**), bien que cette activité soit désignée comme une des dix composantes du SS. Le **système de soutien (SS)** est ainsi considéré comme un axe de connaissance à part entière, tout comme le **système principal** (SP). Nous choisissons de séparer la connaissance du SP de la connaissance de son aspect opérationnel permettant de modéliser son usage. Cette connaissance est relative aux différents contextes dans lesquels évolue le SP. Bien que

relativement récente, cette prise en compte est également définie comme un besoin général aux différentes industries (Brissaud et al. 2010), et de nombreux standards ont vu le jour en ce sens (UTE 2010). Pour être en phase avec cette vision, le **contexte** du système principal est pris en compte comme un quatrième axe de connaissance.

Ainsi, quatre grands axes de connaissance en interactions les uns avec les autres doivent donc être considérés pour permettre de modéliser un modèle métier relatif à la CMPQ. La connaissance relative à chacun de ces grands axes permet de supporter en partie l'étape (1') de construction du modèle générique. La Figure 7 représente ces différents axes

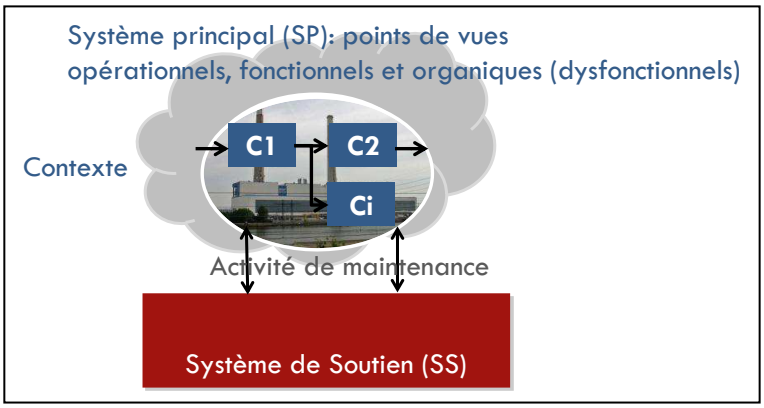


Figure 7 – Les quatre grands axes de connaissance pour la CMPQ

II.3. Proposition d'une démarche d'identification de la connaissance

Les travaux actuels relatifs à la simulation des KPIs de la CMPQ se révèlent insuffisants au niveau de l'identification des connaissances relatives à chaque axe :ceux-ci préfèrent souvent s'intéresser aux capacités du formalisme de modélisation adopté par la suite pour l'étape de simulation du modèle, en proposant directement des règles de correspondances des assertions sur la connaissance à modéliser vers le langage de simulation cible ((Betous-Almeida & Kanoun 2004b) vers les RdP, (Boiteau et al. 2006) vers le langage AltaRicaDF) sans se soucier de l'exhaustivité et de la cohérence de cette connaissance. D'autres travaux cependant ont ce souci d'exhaustivité en proposant une étape préliminaire de formalisation textuelle de la connaissance. Par exemple, (Zille 2009) propose des interactions génériques entre les différentes connaissances d'intérêt. Cependant, les aspects statiques de chaque connaissance sont peu ou pas traités, et chacune d'entre elle n'est pas vue de manière indépendante. (Monnin et al. 2011) et (Bernardi et al. 2008) utilisent une identification de la connaissance permettant de considérer la connaissance à la façon d'objets ou concepts de connaissance indépendants. En revanche, dans leurs travaux, ces concepts sont vus sous une forme statique (et formalisés dans le langage UML). Dans cette logique, nous cherchons à proposer une démarche basée sur la modélisation indépendante des **concepts**, à savoir le format de la représentation des éléments clés de la connaissance métier, intégrant les aspects comportementaux de ceux-ci.

Pour aboutir à cette représentation, nous proposons de nous attâcher à décrire les concepts relatifs à l'activité de maintenance, permettant de restreindre les autres axes au contexte de la CMPQ. Pour ce faire, une démarche descendante est appliquée à cet axe de connaissance, depuis les stratégies jusqu'aux actions de maintenance, en considérant ainsi l'activité de maintenance comme un sous-système à part entière. Nous proposons donc de définir dans un premier temps de manière hiérarchique

le comportement propre de ces concepts, et donc partiel puisque ne considérant pas les interactions entre concepts, et leurs caractéristiques propres dans le contexte de la CMPQ sont définis. Afin de spécifier leur comportement, des représentations génériques (et donc partielles) états-transitions sont établies, pour les concepts au comportement discret. Ensuite, chaque interaction entre concept est spécifiée. Enfin, ces interactions seront affinées selon leur nature dynamique ou paramétrique.

En appliquant la même démarche à chaque autre axe identifié (SS, SP et contexte), une connaissance exhaustive de la CMPQ cohérente avec les besoins d'une telle étude tels que définis précédemment (KPIs, requis de modélisation…) est exprimée textuellement (cf. Figure 8).

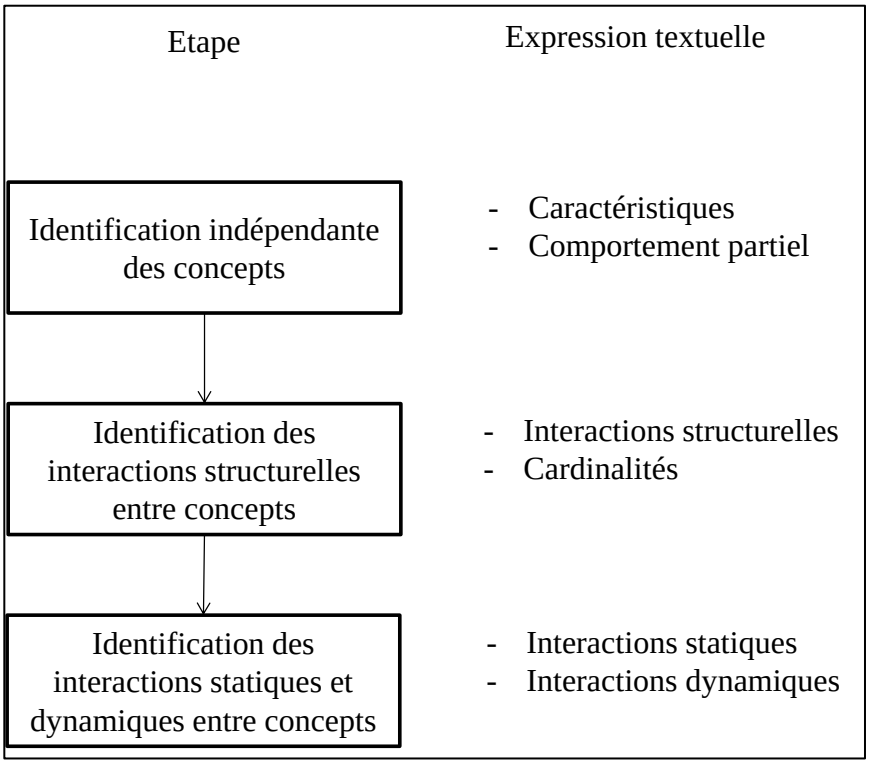


Figure 8 – Démarche d'identification de la connaissance pour la CMPQ

II.4. Identification indépendante des concepts métiers

Cette étape a pour double objectif d'identifier les caractéristiques des concepts entrant en jeu dans la CMPQ, et d'identifier leur comportement. Si les caractéristiques peuvent être identifiées de manière exhaustive, leur comportement doit être vu en interaction avec les autres concepts pour être entièrement défini. Ainsi, les **comportements définis à ce stade sont partiels et représentés des illustrations états transitions également partielles**, et seront complétés lors de la modélisation des relations entre ceux-ci (voir sections II.4, II.5 et II.6).

II.4.1 La connaissance relative à l'activité de maintenance

L'activité de maintenance permet de faire le lien entre le système de soutien et le système principal. Elle se compose de stratégies permettant d'organiser les différentes actions de maintenance agissant au niveau opérationnel selon les politiques définies. Ces politiques peuvent être de natures différentes (cf. Figure 3), et sont en lien avec le SS et le SP à travers les stratégies de maintenance et les actions de maintenance.

D'après (AFNOR 2001), un certain nombre d'actions de maintenance (AM) peuvent être distinguées, et peuvent se répertorier en deux grandes familles :

- les actions de remise en état du composant (dites actions de maintenance actives). Elles sont désignées sous le terme de réparations, définies par (AFNOR 2001) comme « action physique exécutée pour rétablir la fonction d'un bien en panne ». Nous ajoutons à cette famille les actions de maintenance routinières (graissage…). Selon leur type (réparation ou maintenance de routine), ces AM sont appliquées selon leur **durée**. Cette durée peut être modélisée de manière déterministe voir immédiate (A. Barros et al. 2006) comme stochastique (Boiteau et al. 2006). Leur **qualité** est également cruciale à prendre en compte dans le cas de la CMPQ. De nombreux travaux se sont penchés sur cette qualité puisqu'étant forcément traité par les études classiques de sûreté de fonctionnement sur des systèmes réparables. Il existe d'après (Sung, 2008) trois façons de caractériser l'efficacité d'une AM active : les AM parfaites (AGAN), ou « As Good As New », les AM minimales (ABAO) ou « As Bad As Old", et les AM imparfaites, considérées comme la modélisation plus réaliste pour la qualité des AM actives (Doyen & Gaudoin 2004), (Kijima, 1988),
- les actions d'observations. D'après (Zille 2009), les AM d'observations sont « réalisées pour connaitre le niveau de dégradation du système et ce dernier est remplacé si l'observation révèle que le niveau de dégradation est supérieur a un seuil de remplacement préventif fixe, ou si une défaillance apparait ». Elles n'ont donc aucun impact sur la dégradation ou la défaillance du composant. En maintenance conditionnelle, elles permettent de relever périodiquement l'état de dégradation d'un composant. Une action d'observation peut être de plusieurs **types** :
    - une inspection, définie par EDF comme une tâche de maintenance s'appliquant sur un composant sans engendrer d'indisponibilité de celui-ci et permettant de relever ses symptômes. Une inspection peut être caractérisée notamment par des rondes ou du condition monitoring en salle de commande,
    - Un contrôle, défini dans le contexte EDF comme une tâche de maintenance permettant d'évaluer les états de dégradation d'un composant, en engendrant l'indisponibilité de celui-ci. A noter que l'(AFNOR 2001) ne distingue pas les deux précédents types d'AM d'observations. Elles sont englobées sous l'appellation d'inspection,
    - Un essai, défini dans (AFNOR 2001) comme l'action de maintenance permettant de montrer si une caractéristique ou une propriété d'un bien est, ou non, conforme aux prérequis stipulées. Un essai est principalement pratiqué sur un composant passif ou en redondance.

Tout comme les AM actives, les AM d'observations possèdent une **durée** et une **qualité.** Concernant leur qualité, des probabilités de fausse alarme ou de non détection sont traditionnellement attribuées à ces AM (Kallen & van Noortwijk 2005).

A noter que certaines actions de maintenance liées à la gestion de l'obsolescence sont parfois prises en compte, comme dans (Clavareau & Labeau 2009a). Ce point est également lié à la problématique de gestion des stocks, et n'est donc pas requis pour la CMPQ. Il pourra cependant être considéré dans les perspectives de cette thèse.

Ainsi, ce sont ces actions de maintenance physiques qui agissent directement sur le matériel. Cependant, il est nécessaire de considérer la dimension des politiques et des stratégies de maintenance afin de permettre leur coordination avec les autres concepts relatifs à la CMPQ en interaction. Les différents concepts d'intérêt pour l'activité de maintenance afin de définir un programme de maintenance complet sont donc les applications à un niveau opérationnel des politiques de maintenance définies au chapitre 1, permettant de modéliser l'organisation d'un ensemble d'actions de maintenance de façon périodique. Le concept relatif sera par la suite noté « politique », concept reprenant donc les caractéristiques des AM qu'elles coordonnent telles que définies précédemment (durée, type, qualité), et se modélisant de manière discrète. Un ensemble de politiques appliquées formant un programme de maintenance. Le programme s'applique à un SP dans son ensemble tandis que la politique ne s'applique qu'à un composant.

Nous nous intéressons dans un premier temps aux politiques faisant intervenir uniquement des AM actives.

Les **politiques de maintenance corrective** sont cruciales pour modéliser des systèmes réparables. Elles se déclenchent au moment de la défaillance d'un composant ou lorsque l'action de maintenance d'observation visant à relever l'état d'un bien se révèle positive. Lorsqu'elles sont modélisées d'un point de vue indépendant, ces politiques sont souvent modélisées avec un état de préparation, durant lequel les ressources ne sont pas encore disponibles pour la réalisation de l'action (Boiteau et al. 2006). Nous adopterons ce modèle, conduisant ainsi à la définition de trois états pour celle-ci (un état d'attente, un état de préparation et un état d'application). La sortie de l'état d'attente est donc réalisée lorsqu'une panne se crée sur le composant auquel elle s'applique. Les **politiques de maintenance préventives systématique** sont appliquées le plus souvent à des composants sur lesquels peu d'informations sur la dégradation sont disponibles. De nombreux auteurs, répertoriés par (W. Wang 2012) se sont penchés sur des évaluations de ce type de politique, à des fins d'optimisation ou d'aide à la décision. D'un point de vue comportemental, elles se composent donc d'un même nombre d'états que les politiques de maintenance correctives, avec pour seule différence la présence d'un état planifié au lieu d'un état d'attente. La sortie de cet état planifié dépend de la période de cette politique. Concernant **les politiques de maintenance opportunes,** elles se modélisent de manière analogue aux politiques correctives. Cependant, il est nécessaire de leur définir des **conditions de déclenchement** modélisant le passage depuis l'état en attente vers l'état de préparation.

Ces trois types de politiques disposent donc des points communs suivants : elles disposent de trois états (bien qu'ils soient sémantiquement différents) et impliquent l'application uniquement d'AM actives.

Or, une des raisons de l'évolution de la méthode OMF vers l'AP-013 concerne le peu de tâches de maintenance liées à l'application de politiques **de maintenance préventives conditionnelles et prévisionnelles**. Celles-ci sont destinées à des composants mieux surveillés et/ou sur lesquels un retour d'expérience est suffisant pour obtenir des seuils de dégradation. Elles impliquent forcément la combinaison d'actions de maintenance d'observations pour observer l'état du matériel et d'actions de maintenance actives permettant la remise à niveau de ce composant. L'AM d'observation se déclenche dans ces politiques selon une **période** donnée, déclenchant, selon des **conditions** fixées à un niveau politique (Medina Oliva 2011), éventuellement la maintenance active, excepté lors de la surveillance

en continue. De par les problématiques mathématiques engendrées par ce type de modèle, (Marzio Marseguerra et al. 2002), (A. Barros et al. 2006) (Chouiki et al. 2012) cherchent à optimiser les périodes d'observations liées à ce type de politique dans le cas de petits systèmes. (Ben Salem 2008) prend en compte les couts de maintenance dans l'évaluation des politiques, par exemple par le biais de récompenses. D'un point de vue comportemental, une **politique de maintenance préventive conditionnelle** peut être vue comme l'agrégation d'une politique de maintenance systématique déclenchant des observations, puis d'une AM active ne présentant pas d'état d'attente. Ainsi, nous proposons 5 états différents pour la modéliser : l'attente de l'AM d'observation, sa préparation et son application, et la préparation et l'application de l'AM active qu'elle déclenche. Les **politiques de maintenance préventive prévisionnelle**, apparues plus récemment dans la littérature, ont pour but de projeter la dégradation, pour en évaluer l'instant de maintenance adéquat en fonction notamment du stock disponible (Ruin et al. 2009). Ainsi, la décision n'est plus réactive à au franchissement d'un seuil comme pour les politiques de maintenance conditionnelles, mais proactive. La plupart des auteurs traitant ce type de politique travaillent sur les données de dégradation afin d'en déduire les durées de vie résiduelle (Khanh Le Son, 2011) (Carnero 2005) sans envisager une évaluation des indicateurs d'intérêt relatifs à de telles politiques. En plus de la **période d'observation** et des **conditions de déclenchement** similaires aux politiques conditionnelles, il est nécessaire de leur définir un **délai** modélisant l'instant auquel l'AM active débutera. Nous proposons donc 5 états distincts pour la modéliser, avec pour variation par rapport aux politiques conditionnelles un état où l'AM active est planifiée pendant lequel le composant continue à fonctionner en lieu et place de l'état de préparation de l'AM active.

Ainsi, si chaque politique possède un comportement différent du point de vue sémantique, il est possible de caractériser des points communs entre ces concepts. Chacune d'entre elle fait intervenir des actions de maintenance active. Les politiques de maintenance préventives ont le point commun de disposer d'une période. Les politiques préventives basées sur des observations (prévisionnelles et conditionnelles) disposent quant à elles des points communs relatifs aux caractéristiques de l'AM d'observation et de conditions de déclenchement. Elles diffèrent au niveau de leur comportement, et au niveau d'un délai nécessairement défini pour modéliser les politiques prévisionnelles.

Le Tableau 1 reprend les différents concepts de connaissance liés à l'activité de maintenance.

Tableau 1 – Concepts de connaissance liés à l'activité de maintenance

Concept	Caractéristiques	Comportement/ processus
Programme de maintenance		
Politiques de maintenance	Type, durée et qualité de l'AM active	
Politiques de maintenance correctives	Type, durée et qualité de l'AM active	Discret : 3 états (en attente, en préparation, en cours)
Politiques de maintenance opportunes	Type, durée et qualité de l'AM active, condition de déclenchement	Discret : 3 états (en attente, en préparation, en cours)

Politiques de maintenance préventives	Période, Type, durée et qualité de l'AM active	
Politiques de maintenance préventive systématique	Type, durée et qualité de l'AM active, période	Discret : 3 états (planifiée, en préparation, en cours)
Politiques de maintenance basées sur des observations	Type, durée et qualité de l'AM active, type, qualité et durée de l'AM d'observation, condition de déclenchement	
Politiques de maintenance préventives conditionnelles	Type, durée et qualité de l'AM active, type, qualité et durée de l'AM d'observation, condition de déclenchement, période d'observation	Discret : 5 états (planifiée, observation en préparation, observation en cours, active en préparation, active en cours)
Politiques de maintenance prévisionnelles	Type, durée et qualité de l'AM active, type, qualité et durée de l'AM d'observation, condition de déclenchement, délai, période d'observation	Discret : 5 états (planifiée, observation en préparation, observation en cours, active planifiée, active en cours)

D'un point de vue comportemental, concernant la politique corrective et la politique opportuniste, la durée de l'action de maintenance est la seule caractéristique déclenchant une transition interne au concept. Il s'agit de la fin de l'action de maintenance, et donc le passage depuis les états « en cours » vers les états « en attente ». Ce point est communément adopté dans la littérature (Medina Oliva 2011). Nous noterons cette transition « **SC fin AM** » dans le cas d'une politique corrective et « **SO fin AM** » dans le cas d'une politique opportuniste. En effet, le passage vers l'état « en préparation » est déclenché par un évènement extérieur (par une défaillance dans le cas de la politique corrective et par le déclenchement d'une autre politique de maintenance dans le cas d'une politique opportuniste). Le passage vers l'état en préparation dépend quant à lui de la disponibilité du système de soutien. La Figure 9 représente ce comportement dans le cas de la politique de maintenance corrective.

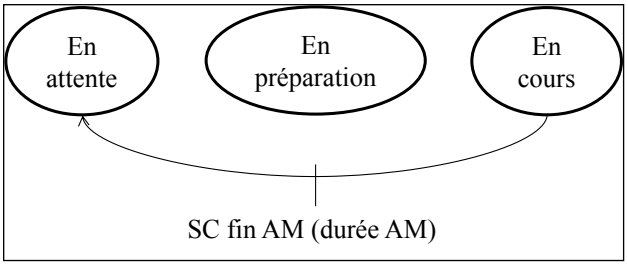


Figure 9 – Comportement partiel d'une politique de maintenance corrective

Concernant la politique préventive systématique, la préparation de l'action de maintenance est déclenchée par la période de la politique (Moghaddam & Usher 2011). Nous noterons cette transition « **SPS début maintenance** ». Ainsi, puisque la fin de la politique, notée « **SPS fin AM** » reste une transition interne de la même manière que pour les politiques précédentes, seule la transition depuis

l'état « en préparation » vers l'état « en cours » est déclenchée par un évènement extérieur. Son comportement est représenté Figure 10.

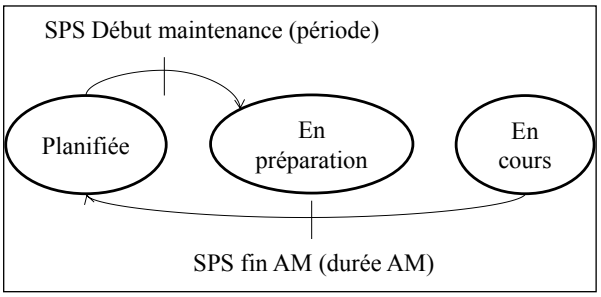


Figure 10 – Comportement partiel d'une politique de maintenance préventive systématique

En ce qui concerne les politiques préventives conditionnelles et prévisionnelles, la préparation de l'observation est déclenchée par la période d'observation. Le passage vers les états de préparation de l'AM d'observation est lié au SS. La politique conditionnelle dispose d'un état de préparation de l'AM active. Cet état n'est atteint à la fin de l'AM d'observation (dépendant de la durée de l'observation) que si celle-ci est positive. La prise en compte de la qualité des actions de maintenance d'observation se fait généralement de deux façons différentes : en supposant que cette AM se réalise parfaitement (Delia & Rafael 2008), ou de manière plus réaliste en supposant que cette AM dispose d'une simple probabilité de fausse alarme et/ou de non détection. Ce modèle est le plus communément admis (Kallen & van Noortwijk 2005) (A. Barros et al. 2006) (Berrade 2012). Cette positivité ou non dépend de plusieurs paramètres, et est définie lors de l'identification des interactions entre les concepts de connaissance. Nous adopterons donc ce point de vue, en créant ainsi une transition conditionnelle depuis l'état d'observation en cours vers soit l'état « act en préparation » ou l'état « planifiée ». Cette transition, dont la durée dépend de la durée de l'observation, est notée « **fin obs** ». Les autres transitions sont analogues à celles des politiques précédemment définies.

Pour la politique prévisionnelle, l'état « act en préparation » est remplacé par l'état « act planifiée », pour préciser que le composant est à nouveau dans son état de fonctionnement. Les ressources étant supposées disponibles pour ce type de politique, le passage depuis cet état vers l'état « act en cours » dépend alors du délai de la politique prévisionnelle (Ruin et al. 2009). Nous adopterons ce modèle, en notant cette transition « **début act** ».

Les comportements de ces types de politiques sont représentés Figure 11 (politique conditionnelle) et Figure 12 (politique prévisionnelle). Pour symboliser les transitions conditionnelles, un symbole en forme de cercle noté C a été introduit, et les conditions sont notées entre crochets. Chaque nom de transition a été précédé par les initiales de la politique à laquelle elles appartiennent.

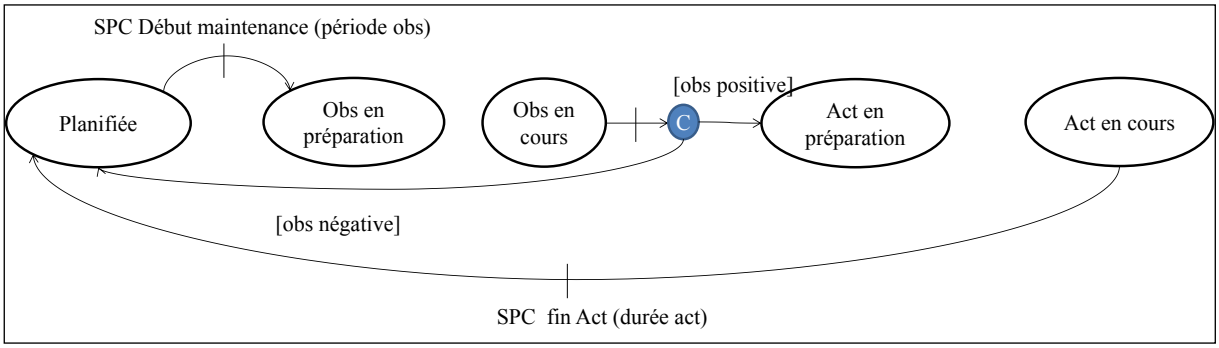


Figure 11 – Comportement partiel d'une politique de maintenance préventive conditionnelle

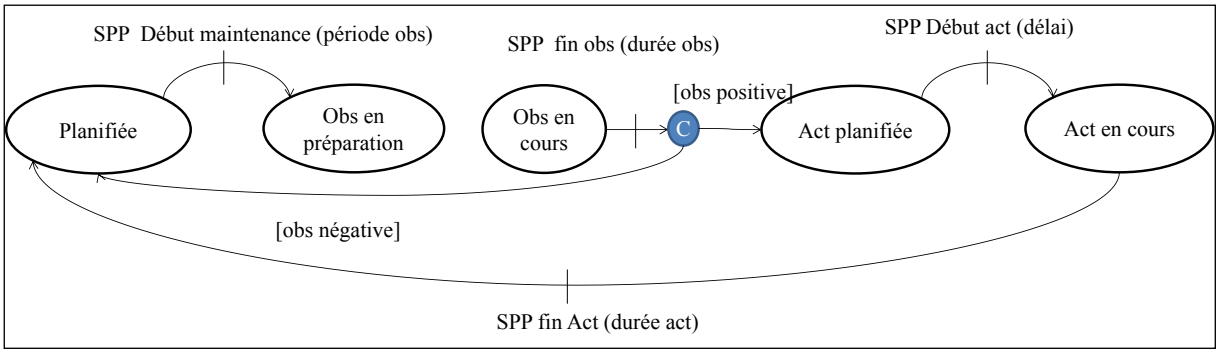


Figure 12 – Comportement partiel d'une politique de maintenance préventive prévisionnelle

A ce stade, les différents concepts de connaissance relatifs à chaque axe de connaissance ont été définis textuellement. Cependant, ils ont été construits de manière indépendante les uns des autres. La prochaine étape consiste donc à identifier, principalement à l'aide des modèles et définitions issus des normes et standards, les relations structurelles liant ces différents concepts.

II.4.2 La connaissance relative au système principal

Afin d'évaluer les KPIs liés aux équipements, il faut considérer deux points de vue d'un **SP** définis par (Meinadier, 2002) : sa vision fonctionnelle et sa vision organique. Concernant la vision fonctionnelle, chaque fonction ou sous fonction du système est à un moment supportée par un composant ou sous système faisant partie de sa vision organique. Cependant, la capacité d'un composant à remplir sa fonction, et donc de maintenir le système disponible, est dépendante de son niveau de dégradation et donc de la facette dysfonctionnelle de son point de vue organique (Nowakowski & Werbińka 2009), point particulièrement important dans le cadre de l'AP-913, afin de pouvoir adopter une granularité de modélisation aussi fine que possible selon le REX disponible.

Vision fonctionnelle du SP

Comme présenté dans les métamodèles de l'IS (AFIS, 2009), tout système peut être décomposé fonctionnellement. Cette décomposition fonctionnelle consiste, à partir de la fonction principale d'un système, à décrire ses différentes **fonctions** et sous fonctions permettant de remplir la fonction principale, jusqu'à pouvoir identifier un composant physique permettant de supporter cette fonction.

A tout niveau de granularité, les fonctions disposent d'un ou plusieurs **flux d'entrée** et d'un ou plusieurs **flux de sortie**. Le rapport entre les flux d'entrée et de sortie de la fonction se fait par l'intermédiaire d'une fonction de transfert. Celle-ci peut être représentée sous la forme d'une équation probabiliste ou déterministe, selon la connaissance de cette fonction. Cependant, dans le cas de systèmes complexes, (Hua et al. 2012) préfèrent prendre en compte de l'aléas sur cette performance. Dans le cas où celle-ci est discrète, alors cette fonction peut disposer de **lois et de paramètres d'évolution** liés notamment à l'usure ou à l'état de dégradation physique des composants la supportant, depuis un état de dégradation fonctionnel moins élevé vers un état de dégradation plus élevé.

Pour représenter l'impact de fonctions de niveau sous-système sur des fonctions de niveau système, il est, d'après (Cocheteux 2010), nécessaire de disposer d'un modèle d'agrégation des indicateurs de performances de niveau le plus bas, fonctions de la dégradation des supports ou composants de ces niveaux (cf. Figure 13), en plus du modèle d'architecture du système permettant de remonter ces indicateurs.

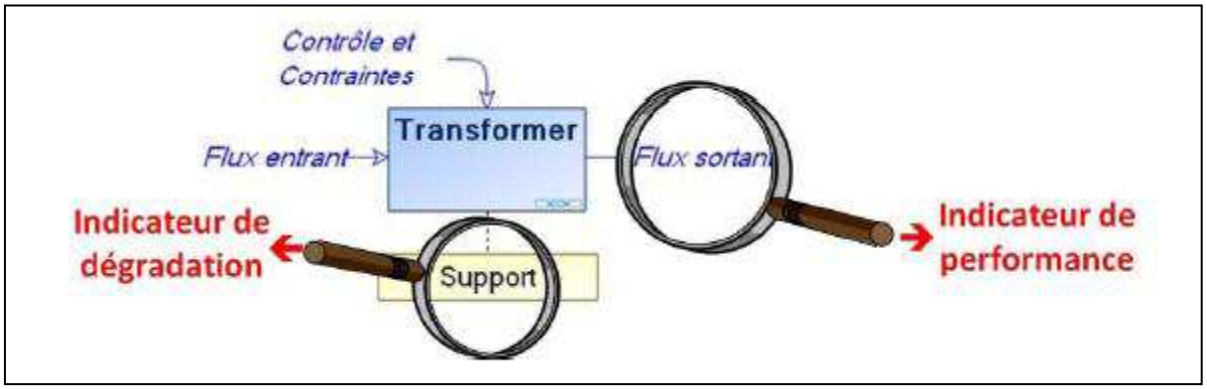


Figure 13 – Liens entre dégradation et performance pour un processus donné (Cocheteux, 2010)

La complexité de ces modèles d'agrégation peut varier. L'auteur utilise ensuite un modèle basé sur les déviations de flux définis à la manière de la méthode HAZOP. Cependant, d'autres modèles tels que des systèmes équations différentielles de dégradation peuvent modéliser cette performance fonctionnelle (Perisse 2003).

Pour évaluer la fonction de transfert d'une fonction, il est cependant nécessaire de disposer des indicateurs de dégradation physique des composants supportant ces fonctions. Ce point est d'autant plus crucial dans le contexte de la CMPQ car il constitue également le principal lien entre SS et SP par l'intermédiaire de l'activité de maintenance. Nous discutons de ces éléments dysfonctionnels dans la section suivante.

Dans le contexte de la CMPQ (les aspects liés à ses caractéristiques techniques telles que le poids sont par exemple superflus), le **composant** ne dispose pas de caractéristiques particulières excepté sa **technologie,** et les **couts** liés à son indisponibilité. En effet, le guide FIDES 2009 (UTE 2010) distingue les évaluations fiabilistes des composants selon les technologies utilisées : les composants pneumatiques, mécaniques et hydrauliques. D'autres aspects, tels que ceux liés à **maintenabilité** des composants, sont à prendre en compte de préférence au niveau dysfonctionnel. Par exemple, un mode de défaillance de celui-ci peut être plus simple à réparer qu'un autre.

D'un point de vue comportemental, les concepts de composant disposent d'un comportement discret. Ils possèdent un nombre d'états variable, dépendant des aspects fonctionnels du système ou de son architecture. Par exemple une vanne peut être ouverte ou fermée, ou dans des états d'attente suite à des redondances. Dans le cas des études de CMPQ, il est cependant possible d'isoler 3 états génériques : l'état de fonctionnement normal, l'état de panne, et l'état de maintenance.

Vision dysfonctionnelle et organique des composants

En termes de communauté scientifique, cette vision dysfonctionnelle des composants (éléments de plus bas niveau identifiés dans l'architecture du système) est plus largement traitée par la communauté de la maintenance. D'après la norme NF EN 13306 (AFNOR 2001) et l'ISO 13379 (ISO, 2003)), on peut recenser trois concepts bien distincts et interagissant entre eux impactant le dysfonctionnement d'un composant : le mode de défaillance, le mécanisme de dégradation et le symptôme.

Un mode de défaillance est la manière dont l'incapacité d'un bien à accomplir une fonction requise se produit (AFNOR 2001) ; u**n mécanisme de dégradation ou défaillance est un ensemble de** processus physiques, chimiques ou autres qui peuvent conduire ou ont conduit à une défaillance (AFNOR 2001) et **un symptôme** est une perception, par observation humaine ou par mesurages (descripteurs), pouvant indiquer avec une certaine probabilité la présence d'un ou de plusieurs défauts (ISO, 2003) .

D'après l'(AFNOR 2001), il existe des **modes de défaillance** génériques, comme par exemple la défaillance à la sollicitation. Ils disposent de caractéristiques telles que la maintenabilité (un mode de défaillance peut être plus facilement réparable qu'un autre), et de caractéristiques liées à leur **loi et leurs paramètres d'occurrence**. Ainsi, d'un point de vue comportemental, un mode de défaillance est par définition un booléen, à savoir représentant la défaillance apparue ou non. Il est déclenché par l'arrivée d'un mécanisme de dégradation à un certain seuil, que la modélisation de la dégradation soit stochastique ou continue, ou par l'occurrence d'une sollicitation ou de facteurs extérieurs.

Les **mécanismes de dégradations** sont des phénomènes traités de manière plus diverse. Ils sont le résultat de processus physiques (érosion, vieillissement…) et peuvent être traitées par des spécialistes du domaine (Jin et al. 2013), sous la forme de systèmes d'équations différentielles parfois très complexes. L'inconvénient de ces méthodes est leur lourdeur en termes de calcul, puisqu'elles font appel à de la simulation continue, ainsi que le fait qu'elles n'autorisent guère l'ajout ou la suppression d'hypothèses. Pour ces raisons, les fiabilistes préfèrent traiter ce phénomène sous la forme d'une modélisation stochastique de la dégradation. Les taux de défaillance sont alors distribués selon des lois de Weibull ou Brownienne (Kogelnik et al. 2002) (Chavanis 2008) (Huynh et al. 2012) . Ces travaux font cependant l'hypothèse d'un REX important disponible sur la dégradation/défaillance des matériels, et ne considèrent que des systèmes avec peu de composants (A. Barros et al. 2006). D'autres travaux s'adaptent en revanche à la connaissance de la dégradation en la discrétisant sous la forme de modèles états-transitions stochastiques (Monnin et al. 2011). Parfois même, l'obtention des taux de transition entre chaque état se réalise en fonction des politiques de maintenance appliquées (Lair, 2012) : si une politique de maintenance nécessitant la modélisation de seuils de décisions intermédiaires est appliquée, la dégradation disposera d'au moins 3 états. Ainsi, sous cette hypothèse discrète, le mécanisme de dégradation dispose de caractéristiques telles que **lois et de paramètres d'occurrence liés à son évolution**.

La modélisation comportementale des **symptômes** est moins traitée dans la littérature. Elle peut cependant se modéliser sous une forme analogue à celle des mécanismes de dégradation avec des modèles états transitions (Zille 2009).

Le Tableau 2 reprend les différents concepts de connaissance liés au SP, ses caractéristiques, et la ou les natures de leur comportement tels qu'identifiés précédemment.

Tableau 2 – Concepts de connaissance liés au SP

Concept	Caractéristiques	Comportement/ processus
SP		
Fonction	Flux entrants, flux sortants, loi et paramètre d'évolution	Continu/ discret : N états
Composant	Technologie, couts d'indisponibilité	Discret : N états (≥ 3 : en marche, en maintenance prev, en panne/ en maintenance corr)
Mode de défaillance	Maintenabilité, loi et paramètres d'occurrence	Discret : 2 états (apparu ou non)
Symptôme	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états
Mécanisme de dégradation	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états

A ce niveau, une représentation informelle état-transition peut être réalisée sur les concepts au comportement discret, à partir des états définis précédemment. Il reste cependant à définir les transitions propres à chaque concept, ou transitions internes, permettant le passage d'un état à un autre. Nous donnons ici la sémantique de cette transition, ainsi qu'un terme générique permettant de la nommer plus simplement.

Dans le cas d'un mode de défaillance, les états sont « apparu » ou « non apparu ». Seule la transition depuis l'état « non apparu » vers l'état « apparu » peut être définie, puisque dépendant de sa caractéristique « loi et paramètres d'occurrence ». La Figure 14 reprend cette proposition, avec la transition interne nommée « **apparition** », et entre parenthèses les caractéristiques desquelles elle dépend.

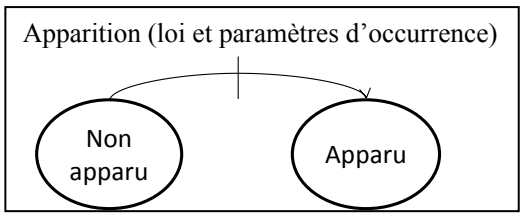


Figure 14 – Comportement partiel d'un mode de défaillance

Les fonctions et composants ne disposant pas de comportement générique (chaque transition est soit liée à un cas particulier soit liée à leur dysfonctionnement), il est inutile de représenter leur aspect état-transition. Dans le cas des mécanismes de dégradation et des symptômes, alors les transitions depuis les états de dégradation moins élevés vers les états de dégradation plus élevés peuvent être représentées à ce stade. En effet, chaque caractéristique relative à cette évolution (« loi et paramètres d'évolution ») peut être associée à une transition sous l'hypothèse d'une évolution discrète. Pour simplifier les représentations, chaque transition et les lois et paramètres d'évolution desquels elle dépend seront suivis de deux indices, le premier relatif à son état initial et le second relatif à son état final, comme représenté Figure 15 dans le cas d'un mécanisme de dégradation ou symptôme à trois états. Ainsi, la transition « évolution de la dégradation du mécanisme depuis l'état i vers l'état j » est notée « **Evolution_ij** ».

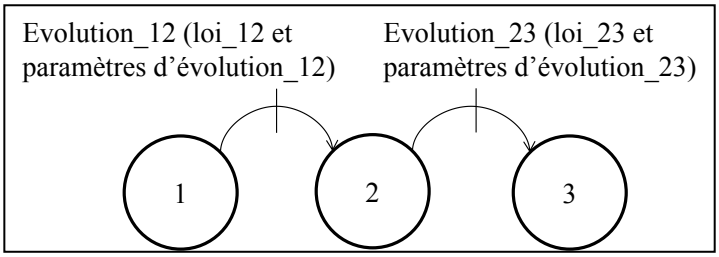


Figure 15 – Comportement partiel d'un mécanisme de dégradation à 3 états

II.4.3 La connaissance relative au système de soutien

La CMPQ nécessite la modélisation, en synergie avec le SP, du SS de celui-ci. A travers l'OSA-EAI (Open System Architecture for Entreprise Application Integration), le standard MIMOSA (www.mimosa.org) propose un modèle unifié orienté objet (formalisé sous la forme de diagrammes de classes UML) fournissant un cadre pour le développement d'application pouvant s'intégrer dans les systèmes d'information d'entreprises prenant en compte la maintenance. Il propose trois grands types de concepts :

- **Les opérateurs de maintenance,**
- **Les pièces de rechanges,**
- **Les outillages.**

D'après la classification des composants selon leur technologie proposée précédemment, il est nécessaire de distinguer des **opérateurs** selon leur spécialité. Par exemple, certains peuvent être spécialisés dans les tâches d'observation, et ceux spécialisés dans la réparation de composants mécaniques. De même, les outillages nécessaires à la réalisation de ces tâches seront parfois différents. Quel que soit leur domaine d'application, ces ressources peuvent être de **qualité** différente. Dans le cas d'un opérateur, nous parlerons à ce niveau de **compétence** liée à la formation (Jeang 1998). De même le **cout** doit être considéré à ce niveau. En effet, ce cout lié aux ressources tient une place prépondérante dans les couts globaux liés à la maintenance (Ravichandran 1993). Ces couts peuvent être liés à la possession en stock lié à l'immobilisation financière des ressources, ou à leur achat (cas des ressources non réutilisable). Concernant les opérateurs, nous proposons de considérer les charges salariales comme équivalent au cout de possession des ressources. D'autres couts peuvent être pris en

compte notamment le cout de travail d'un opérateur ou le cout d'installation d'un composant de rechange, mais nous les supposons hors du domaine de connaissance requis pour la CMPQ.

Cependant, l'application de certaines actions de maintenance peut nécessiter plusieurs ressources simultanément. Ainsi, nous ajouterons aux ressources ci-dessus les **équipes de ressources**, se constituant de plusieurs ressources.

D'un point de vue comportemental, ces ressources peuvent également être considérées de deux manières différentes : les **ressources réutilisables** et les **ressources non réutilisables**. Les comportements des ressources non réutilisables impliquent notamment la prise en compte de périodes de réapprovisionnement, et sont la plupart du temps modélisés par des files d'attentes (Sanajian & Balcıog˘lu 2009). Ces ressources disposent donc d'un nombre d'états équivalent à leur **quantité** disponible en stock. Le réapprovisionnement de ces ressources induit cependant la prise en compte de problématiques complexes connexes telles que la gestion des stocks (Pennington & Strùmme 1998), qui ne seront pas prises en compte dans le cadre de cette thèse, mais devraient faire l'objet de perspectives.

Le comportement des ressources réutilisables est lui en revanche souvent vu comme un booléen (Boiteau et al. 2006) en adéquation avec les attentes de la CMPQ. Les équipes de ressources disposent d'un comportement différent : elles doivent en effet disposer d'un état supplémentaire, modélisant l'attente potentielle provoquée par des ressources indisponibles. Nous nommerons cet état « en constitution ». De la même manière que pour les ressources, l'instant d'entrée dans cet état est lié aux évolutions des politiques de maintenance. Elles ont pour seule caractéristique leurs conditions de déclenchement, modélisant le nombre de ressources nécessaires dans l'équipe pour qu'elle puisse effectuer la maintenance.

Le Tableau 3 synthétise cette connaissance.

Tableau 3 – Concepts de connaissance liés au SS

Concept	Caractéristiques	Comportement/ processus
Ressource réutilisable	Qualité, spécialité, cout de possession/ charge salariale	Discret : 2 états (disponible, indisponible)
Ressource non réutilisable	Qualité, spécialité, nombre, cout d'achat, cout de possession	Discret : N états, N étant le nombre de ressources disponibles à l'état initial
Equipe de ressources	Conditions de déclenchement	Discret : 3 états (disponible, en constitution, en travail)

D'un point de vue comportemental, il n'est pas nécessaire de réaliser de modèle état-transition des ressources puisque leurs transitions sont déclenchées par les politiques de maintenance. Elles ne seront représentées qu'après avoir identifié les interactions dynamiques.

En revanche, dans le cas des équipes de ressources, la transition depuis l'état de constitution vers l'état de travail est une transition interne. Elle dépend en effet des conditions de déclenchement de celle-ci. Nous la nommerons « start_job_equipe » (Figure 16).

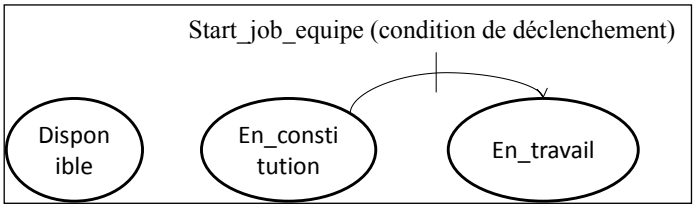


Figure 16 - Comportement partiel d'une équipe de ressources

II.4.4 La connaissance relative au contexte

Ce contexte est lié au contexte d'utilisation du système principal. En effet, pour un même système, celui-ci peut être par exemple utilisé d'une manière plus intensive, ou dans d'autres environnements (système évoluant dans un milieu plus ou moins humide par exemple). Leur modélisation dans le contexte de la CMPQ est importante puisqu'ils peuvent impacter de manière plus ou moins significative le dysfonctionnement des matériels.

Nous avons identifié trois principaux objets, parfois appelés facteurs influents, pouvant impacter sur le profil de fonctionnement du SP :

- **Les conditions opérationnelles** (CO) : utilisation importante ou non d'un matériel,
- **Les conditions environnementales** (CE), particulièrement importantes dans le contexte d'EDF, type environnement dans lequel évolue le matériel (humidité, température…), en utilisation (appartenant au SP) comme en stock (appartenant au SS).
- **Les chocs** : évènements discrets liés à l'environnement pouvant entrainer des dégradations/défaillance sur un matériel un utilisation ou en stock.

Peu de travaux liés à la maintenance considèrent les conditions environnementales de par leur nature très incertaine. Elles sont plutôt répertoriées qualitativement dans certaines études d'analyse du risque (Aven et al. 2006). Cependant, il s'agit de conditions primordiales à prendre en compte (Bottelberghs 2000). Selon la nature (hydraulique, pneumatique…) des composants du système principal, elles peuvent en effet s'avérer critiques pour leur bon fonctionnement. Si des travaux de météorologistes tentent de modéliser ces facteurs, comme par exemple l'évolution de la température, de manière déterministe, ces prévisions ne restent valables qu'à court terme. Ainsi, dans le but de prendre en compte l'impact de ces facteurs sur la dégradation, ces conditions sont discrétisées en fonction de la connaissance sur leur **intensité**. (Piero Baraldi et al. 2011) les considère sous la forme de nombre flous attribués par un expert. (Monnin et al. 2011) les discrétise sous la forme d'un Stochastic Activity Network et attribue aux transitions des taux suivant une loi d'évolution stochastique. Le passage depuis un état vers un autre se fait selon un **planning** d'évolution. (Huynh et al. 2011) ont modélisé ces conditions environnementales comme des processus de chocs ou de stress arrivant selon une loi de poisson, et pouvant entraîner l'occurrence d'un mode de défaillance sur une ressource matérielle en stock. Nous généralisons ce point en supposant que les chocs sont un cas particulier de condition environnementale impactant le SP de manière discrète.

Les conditions opérationnelles sont fixées en amont de la CMPQ. Ainsi, il est possible de prévoir de manière assez précise leur évolution, selon l'**intensité** des cadences de fonctionnement des composants sur lesquels elles sont appliquées (utilisation faible, élevée…). Ainsi, on retrouve les

modèles utilisés pour modéliser les conditions environnementales (nombres flous (Piero Baraldi et al. 2011), réseaux de Pétri (Zille 2009)…), toujours selon un **planning** fixé a priori par un responsable de maintenance. Selon les systèmes, ces conditions peuvent également être vues sous la forme de nombres de cycles, chaque départ de cycle pouvant entraîner un mode de défaillance à la sollicitation.

Le Tableau 4 reprend les différents concepts de connaissance liés au contexte, ses caractéristiques, et la ou les natures de leurs comportement tels qu'identifiés précédemment.

Tableau 4 – Concepts de connaissance liés au contexte

Concept	Caractéristiques	Comportement/ processus
Conditions opérationnelles	Intensité, planning	Discret : N états
Conditions environnementales	Intensité, planning	Continu/ Discret : N états
Chocs	Période	Discret : 2 états (actif, inactif)

Dans le cadre d'une représentation état transition, pour une condition opérationnelle comme environnementale, sous l'hypothèse d'un modèle discret, chaque état est atteignable depuis un autre, et donc une transition interne « évolution de la condition environnementale depuis son état d'intensité i vers son état d'intensité j » peut être définie. Ainsi, pour simplifier les représentations, nous procèderons de manière analogue aux mécanismes de dégradation en attribuant un indice aux transitions et au planning selon les états qu'ils relient (« **planning_*ij*** »), comme représenté Figure 17 dans le cas d'une condition environnementale avec trois niveaux d'intensité.

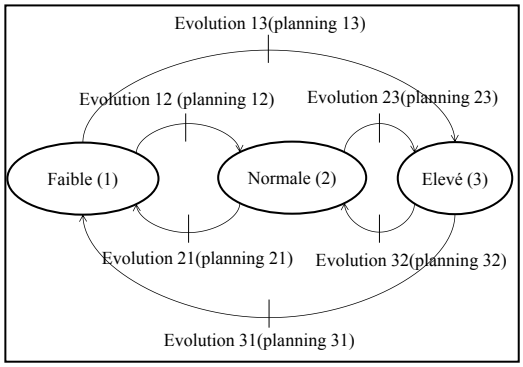


Figure 17 – Comportement d'une condition environnementale à 3 états

Concernant les chocs, le passage vers l'état actif dépend de la période des chocs. Ce passage est traduit sous la forme de la transition « **déclenchement** ». Le retour à l'état inactif est lui instantané, et se traduit sous la forme de la transition « **fin** ». La Figure 18 représente le modèle état-transition relatif aux chocs.

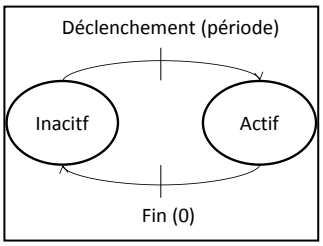


Figure 18 - Comportement d'un processus de chocs

II.5. Identification des interactions structurelles entre les différents concepts de connaissance

Il convient à présent d'identifier les relations éventuelles entre les différents concepts de manière générique. Nous parlerons ici d'interactions structurelles, et nous définirons des cardinalités entre les concepts en relation. Ces interactions vont permettre de pouvoir définir n'importe quel système sur lequel une étude de CMPQ peut se réaliser. Ces interactions structurelles se définissent principalement en reprenant les définitions des normes et standards citées précédemment (MIMOSA, (AFNOR 2001)…), et nécessitent de bien définir les cardinalités de ces relations. Nous allons procéder par dichotomie, en identifiant ces relations en partant de l'axe de connaissance relatif à l'activité de maintenance, et ce de manière hiérarchique.

Il a été établi que le programme de maintenance est l'application d'un ensemble de politiques au niveau opérationnel. Celui-ci se compose donc d'une ou plusieurs politiques de maintenance. De même, une politique de maintenance peut être hiérarchisée ou typée selon sa nature (corrective, opportuniste ou préventive). Les politiques préventives sont ensuite elle-même hiérarchisées ou typées selon leur nature (systématique ou basée sur des observations). Enfin, les politiques basées sur des observations peuvent être de types conditionnelles ou prévisionnelles.

Dans le cas où la maintenance est basée sur des AM actives, alors elle peut **agir** sur **un ou plusieurs** symptômes ou mécanismes de dégradation. En effet, elle peut entraîner une remise à niveau de ceux-ci. Si la politique est corrective, alors elle est par définition **déclenchée** par un mode de défaillance. Réciproquement, une AM active d'une politique corrective agit forcément sur un mode de défaillance.

Si les AM d'observation sont considérées, elles sont « réalisées pour connaitre le niveau de dégradation du système et ce dernier est remplacé si l'observation révèle que le niveau de dégradation est supérieur à un seuil de remplacement préventif fixe, ou si une défaillance apparait ».

Ainsi, pour chaque politique de maintenance intégrant une AM d'observation, **au moins un** élément dysfonctionnel (symptôme, mécanisme de dégradation) doit lui être rattaché afin que l'AM d'observation puisse physiquement **relever leur niveau**. A contrario, un symptôme ou un mécanisme de dégradation **peut avoir aucune ou plusieurs** AM d'observation pour relever son niveau.

De par l'encapsulation des AM au sein des politiques de maintenance, des relations de natures différentes apparaissent parfois entre les concepts (par exemple, une politique de maintenance conditionnelle peut à la fois, de par son AM active, remettre à niveau un mécanisme de dégradation, et de par son AM d'observation, relever le niveau d'un mécanisme de dégradation).

Si la politique est opportuniste, elle est **déclenchée** par une autre politique de maintenance. Enfin, **toutes** les politiques de maintenance **peuvent** être **synchronisées** entre elles, afin par exemple d'adapter les périodes de l'ensemble des politiques préventives à l'occurrence d'une panne.

Le Tableau 5 reprend les interactions structurelles liées à l'activité de maintenance. Lorsqu'un concept A est forcément en relation avec au moins un concept B, alors la cardinalité du concept B est notée 1…N. Lorsqu'un concept A peut être en relation avec au moins un concept B, alors la cardinalité du concept B est notée 0…N. Enfin, lorsque qu'un concept B est lié à un et un seul concept A, la cardinalité du concept A est notée 1. A noter que les relations hiérarchiques ne nécessitent pas de définir de cardinalité.

Tableau 5 – Interactions structurelles liées à l'activité de maintenance

Concept A	Cardina-lité A	Relation de A vers B	Concept B	Cardinalité B
Programme de maintenance	1	Se compose	Politique de maintenance	1…N
Programme de maintenance	1…N	S'applique à	SP	1…N
Politique de maintenance		Peut être	Politique préventive	
Politique de maintenance		Peut être	Politique opportuniste	
Politique de maintenance		Peut être	Politique préventive	
Politique préventive		Peut être	Politique basée sur des observations	
Politique préventive		Peut être	Politique préventive systématique	
Politique basée sur des observations		Peut être	Politique prévisionnelle	
Politique basée sur des observations		Peut être	Politique conditionnelle	
Politique basée sur des observations	1…N	Relève niveau	Symptôme	0…N
Politique basée sur des observations	1…N	Agit sur	Mécanisme de dégradation	0…N

Politique de maintenance (en général)	0…N	S'applique	Composant	1…N
Politique de maintenance (sauf opportuniste)	1…N	Déclenche	Politique de maintenance opportuniste	0…N
Politique de maintenance	0…N	Synchronise	Politique de maintenance	0…N
Politique de maintenance corrective	1…N	Est déclenchée	Mode de défaillance	1…N

Concernant les concepts du SP définis Tableau 2, en reprenant les méta-modèles de l'IS (AFIS, 2009), un SP comporte au moins une fonction et au moins un composant. Les fonctions **peuvent être associées à une ou plusieurs autres** fonctions à un niveau de granularité donné, et **se composent** elles-mêmes de fonctions (cf. Figure 13). A leur plus bas niveau de granularité, elles sont supportées **par un ou plusieurs** composants physiques selon la connaissance disponible. Le Tableau 6 reprend les interactions structurelles liées à la décomposition fonctionnelle des systèmes.

Tableau 6 – Interactions structurelles liées à la décomposition fonctionnelle des systèmes

Concept A	Cardinalité A	Relation de A vers B	Concept B	Cardinalité B
SP	1	Comporte	Fonction	1…N
SP	1	Comporte	Composants	1…N
Fonction	1	Se compose	Fonction	1...N
Fonction	0…N	Est associée	Fonction	0…N
Fonction	1…N	Est supportée	Composant	1...N

Dans cette logique, chaque composant **peut-être** lui-même associé à **d'autres** composants pour supporter une fonction. Concernant les aspects dysfonctionnels du composant définis Tableau 2, en reprenant les définitions relatives aux mécanismes de dégradation, aux symptômes et aux modes de défaillance, alors ceux-ci **se composent** dans le contexte de la CMPQ **d'au moins un** mode de défaillance (si aucun mode de défaillance n'est attribué à un composant, alors il n'est pas nécessaire de le modéliser), et **éventuellement d'un ou plusieurs** mécanismes de dégradation ou symptômes. Les symptômes sont révélateurs d'un mécanisme de dégradation. Ces mécanismes de dégradation **impactent** quant à eux non seulement la cinétique **d'un ou plusieurs** modes de défaillance, mais aussi **éventuellement** d'autres mécanismes de dégradation (E. Zio & P. Baraldi 2003), des éventuels symptômes associés, ou des fonctions du composant auxquelles ils appartiennent. De même, un mode de défaillance **peut entraîner l'apparition** d'autres modes de défaillance (Hoepfer et al. 2009). Le Tableau 7 reprend les interactions structurelles liées aux aspects dysfonctionnels des composants.

Tableau 7 – Interactions structurelles liées aux aspects dysfonctionnels des composants

Concept A	Cardinalité A	Relation de A vers B	Concept B	Cardinalité B
Composant	0…N	Associé à	Composant	0…N
Composant	1…N	Dispose de	Mécanisme de dégradation	0…N
Composant	1…N	Dispose de	Mode défaillance	1…N
Composant	1…N	Dispose de	Symptôme	0…N
Mécanisme de dégradation	0…N	Impacte la cinétique	Symptôme	0…N
Mécanisme de dégradation	0…N	Impacte la cinétique	Mode de défaillance	1…N
Mécanisme de dégradation	0…N	Impacte la cinétique	Mécanisme de dégradation	0…N
Mécanisme de dégradation	0…N	Impacte la cinétique	Fonction	1…N
Mode de défaillance	0…N	Entraîne apparition	Mode de défaillance	0…N

Concernant le contexte, les Conditions Opérationnelles et Environnementales (resp. CO et CE) **peuvent impacter la dynamique d'évolution** des mécanismes de dégradation et des modes de défaillance. Les chocs, eux, ne **peuvent** entraîner que l'apparition des modes de dégradation. De manière réciproque, ces concepts liés au contexte peuvent tout à fait ne pas être connus, ou a contrario **plusieurs** CO ou CE peuvent impacter **un unique** concept dysfonctionnel d'un composant. Le Tableau 8 reprend les interactions structurelles liées au contexte.

Tableau 8 – Interactions structurelles liées au contexte

Concept A	Cardinalité A	Relation de A vers B	Concept B	Cardinalité B
CO	0…N	Impacte le cinétique	Mécanisme de dégradation	0…N
CE	0…N	Impacte la cinétique	Mécanisme de dégradation	0…N
CO	0…N	Impacte la cinétique	Mode de défaillance	0…N
CE	0…N	Impacte la	Mode de	0…N

		cinétique	défaillance	
Chocs	0…N	Entraîne apparition	Mode de défaillance	0…N

Concernant le SS, les ressources réutilisables comme les ressources non réutilisables sont gérées à travers les politiques de maintenance quelles qu'elles soient. Selon les modèles, il est envisageable de ne pas prendre en compte ce SS (dans le cas ou peu ou aucune information ne serait disponible à ce sujet). Ainsi, une **cardinalité minimale de 0** doit être affectée aux ressources. De la même manière, **plusieurs** ressources **peuvent être consommées** par au moins une politique de maintenance. Une équipe de ressources se compose quant à elle d'au moins une ressource, et est également consommée par une politique de maintenance. Le Tableau 9 reprend les interactions structurelles liées au SS.

Tableau 9 – Interactions structurelles liées au SS

Concept A	Cardinalité A	Relation de A vers B	Concept B	Cardinalité B
Ressource (en général)	0…N	Consommée par	Politique de maintenance	1…N
Equipe de ressources	1	Se compose	Ressources réutilisables / ressources non réutilisables	1…N
Equipe de ressources	0…N	Consommée par	Politique de maintenance	1…N

Ces différentes relations structurelles entraînent ou non des impacts sur les concepts associés. Ces impacts peuvent également être multiples : par exemple, le comportement d'une politique de maintenance conditionnelle dépend de l'état d'un mécanisme de dégradation associé. Mais aussi, elle va impacter l'état de ce mécanisme à la suite de la réalisation de l'action de maintenance active à travers la remise à niveau du mécanisme de dégradation. Il est donc nécessaire de spécifier les différents impacts possibles entre les différents concepts. Ces impacts peuvent être de deux types :

- **Paramétriques**: ils concernent les relations entre caractéristiques de concepts. Par exemple un mécanisme de dégradation, selon son état, peut impacter le taux de défaillance ou l'équation modélisant l'occurrence du mode de défaillance d'un composant,
- **Dynamiques** : ils concernent les relations entre transitions des concepts. Par exemple, la fin d'une action de maintenance entraînera le redémarrage du composant. Cette modélisation permet d'identifier les évènements synchronisés du modèle, et va contribuer à préciser les données déclenchant les transitions forcées.

II.6. Identification des interactions dynamiques entre les différents concepts de connaissance

Pour identifier ce type d'interactions, il est nécessaire de dérouler, à partir des transitions internes aux concepts identifiées précédemment, les interactions dynamiques possibles entre chaque axe de

modélisation. A partir de là, chaque transition non renseignée dans l'étape de formalisation indépendante des concepts doit pouvoir être complétée. Nous allons poser des hypothèses afin de simplifier l'identification de ces interactions.

II.6.1 Cas des interactions dynamiques entre le système principal et l'activité de maintenance

Dans ce cas de figure nous ne considérons le SP que sous la forme d'un simple composant. En effet, excepté dans le cas de la maintenance opportuniste, les relations entre maintenance et chaque composant d'un système est analogue. Ce composant est vu dans un état de dégradation initial nominal (au niveau à la fois des symptômes, des mécanismes de dégradation et des modes de défaillance), avec un ensemble de politiques de maintenances différentes lui étant appliquées.

Les symptômes et mécanismes de dégradations sont par nature un phénomène continu. Cette interaction est donc paramétrique (voir section II.6).

L'occurrence de modes de défaillance peut engendrer plusieurs évènements forcés. Le composant passe en effet, par définition, en état de panne. Nous notons cette transition forcée « **panne** ». De même, d'autres modes de défaillance liés peuvent arriver à ce même moment, comme modélisé dans les travaux sur les défaillances de cause commune (Chouiki et al. 2012). Cette transition est notée « **début maintenance** ». La politique de maintenance corrective peut également être automatiquement déclenchée à l'apparition de ce mode, dans les cas où ce mode est supposé immédiatement visible (sans le besoin d'AM d'observation pour le détecter) (Clavareau & Labeau 2009a) (Chouiki et al. 2012). Cette transition est notée « **occurrence forcée** », pour la différencier de la transition « occurrence » identifiée précédemment.

Lorsque des politiques de maintenance préventives s'appliquent au composant, alors celles-ci peuvent déclencher des évènements sur celui-ci et son dysfonctionnement. En effet, le déclenchement de ces politiques est périodique. Ceci entraîne donc le changement d'état du composant depuis l'état « en marche » vers l'état « en maintenance préventive », que nous notons « **début maintenance préventive** ». En effet, bien que seuls les contrôles entraînent théoriquement une indisponibilité du composant (AFNOR 2001), les autres AM d'observations peuvent être supposées comme entraînant une indisponibilité nulle.

Dans le cas d'une politique corrective uniquement, la fin de l'AM active entraîne la disparition du mode de défaillance sur lequel elles agissent. En effet, nous les avons définies comme pouvant se déclencher automatiquement à l'occurrence de celui-ci. De plus, le composant étant alors dans l'état de panne, il est nécessaire de le faire basculer à ce moment dans son état de fonctionnement. Ainsi, nous proposons la définition de deux transitions forcées allant en ce sens, à savoir « **fin panne** » depuis l'état « panne » vers l'état « en marche » du composant, et « **disparition** », depuis l'état « apparu » vers l'état « non apparu » du mode de défaillance.

Dans le cas d'une politique de maintenance préventive, trivialement, le composant passe quant à lui de l'état « en maintenance préventive » vers l'état « en marche » lors de son application (transition « **début maintenance préventive** »). Dans le cas des politiques de maintenance conditionnelles, la fin de l'observation, selon sa positivité ou sa négativité, entraîne respectivement le début de l'AM active

(cf. Figure 11) ou le retour du composant à son état de fonctionnement, symbolisant le fait qu'aucune dégradation critique n'ait été détectée. Le composant ayant préalablement été mis dans son état de maintenance préventive, nous supposons que le début de l'AM active dans ce contexte n'entraîne aucune transition forcée pour l'observation positive. Dans le cas d'une observation négative, alors une transition forcée « **fin maintenance préventive** » peut être déclenchée sur le composant.

Si la politique est prévisionnelle, la différence se situe dans le fait que l'observation positive entraîne un retour du composant à l'état de marche. En effet, l'AM active débutera seulement après un délai (Ruin et al. 2009). De ce fait, quelle que soit l'issue de l'observation, la transition « **fin maintenance préventive** » est franchie sur le composant. En revanche, la transition « début act » implique un retour du composant dans son état de maintenance préventive par la transition forcée « **début maintenance préventive** ».

D'une manière générale, dans le cas de toute politique de maintenance, la fin de leur AM active entraîne la remise à niveau du mécanisme de dégradation sur lequel elles agissent, mais aussi éventuellement du symptôme associé. Pour modéliser ce point, il est nécessaire de s'intéresser aux travaux de modélisation de la maintenance. De nombreux travaux (Doyen & Gaudoin 2004) (Jin et al. 2013) ont abordé la qualité de l'action de maintenance puisqu'étant forcément traité par les études classiques de sûreté de fonctionnement sur des systèmes réparables. Au niveau des actions de maintenance actives, les AM imparfaites ont été identifiées précédemment comme la modélisation la plus réaliste des AM. Ainsi, depuis chaque état, une transition forcée doit être définie depuis l'état dans lequel a débuté la maintenance vers un état de dégradation moindre. Nous notons cette transition « **remise à niveau *ij*** », l'indice *i* modélisant l'état de dégradation actuel et l'indice *j* l'état de dégradation atteint. La manière de calculer cette qualité est discutée section II.6.

Le Tableau 10 résume les différentes interactions dynamiques entre le SP et l'activité de maintenance.

Tableau 10 – Interactions dynamiques entre le SP et l'activité de maintenance

Transition	Concept en interaction	Transition forcée engendrée
Apparition mode défaillance	Politique corrective	Début maintenance (état « en attente » vers état « en préparation »)
	Composant	Panne (état « en marche » vers état « en panne »)
	Mode de défaillance	Occurrence forcée (état « non apparu » vers état apparu)
Début politique préventive	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »)
Fin AM active (quelle que soit la politique préventive)	Composant	Fin maintenance préventive (état « en maintenance

		préventive » vers état « en marche »
Fin AM active (politique corrective)	Composant	Fin panne (état « en panne» vers état « en marche »
Fin AM active (quelle que soit la politique)	Mécanisme de dégradation/ Symptôme	Remise à niveau ij (depuis un état de dégradation élevé i vers un état de dégradation équivalent ou moins élevé j)
Fin AM active (politique corrective)	Mode de défaillance	Disparition (état « apparu » vers état « non apparu »
Fin observation négative (quelle que soit la politique basée sur des observations)	Composant	Fin maintenance préventive (état « en maintenance préventive » vers état « en marche »
Début act (politique prévisionnelle)	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »

II.6.2 Cas des interactions dynamiques entre l'activité de maintenance et le système de soutien

Quelle que soit l'AM déclenchée, des ressources soit réutilisables ou non réutilisables ont préalablement été définies comme nécessaires à sa réalisation. Si cette ressource est réutilisable, elle a été définie Tableau 3 comme possédant 2 états. La sollicitation de chacune de ces ressources entraîne son passage depuis l'état indisponible vers l'état disponible (Boiteau et al. 2006). Si la ressource n'est pas disponible à ce moment, alors la transition est réservée pour participer, à son retour, à la réalisation de l'AM en attente. Lorsque l'ensemble des ressources est disponible pour réaliser l'AM, alors celle-ci peut se réaliser. Ainsi, puisque nous avons défini précédemment l'équipe de ressources comme l'entité gérant les ressources à sa disposition, lorsque la transition « début maintenance » est franchie, et ce quelle que soit la politique ou le type d'AM, la transition forcée « **constitution** » doit être déclenchée chez l'équipe de ressource. L'interaction dynamique est donc ici conditionnée par la disponibilité ou non de la ressource.

Lorsque l'équipe de ressource est constituée (ce qui intervient selon les conditions de déclenchement de cette équipe, et la transition « start_job_equipe »), alors l'action de maintenance peut commencer et la politique de maintenance doit passer de l'état « en préparation » vers l'état « en cours » par la transition « **début AM** », que celle-ci soit active ou d'observation. De même, cette transition entraîne la transition forcée « **start job i** » modélisant le passage depuis l'état supérieur *i* vers l'état inférieur tant que cet état est supérieur à 0. Lorsque la ressource est réutilisable, la transition forcée « **start job** » est tirée sur chacune d'entre elle.

Lorsqu'une AM se termine, alors l'équipe de ressources peut se libérer et la transition forcée « **end job équipe** » franchie. Toutes ses ressources réutilisables peuvent alors retourner dans leur stock à

l'état « disponible » (Zille 2009) (nous notons cette transition « **end job** »), tandis que, par définition, les ressources non réutilisable sont consommées.

Le Tableau 11 résume les différentes interactions dynamiques entre le SS et l'activité de maintenance.

Tableau 11 – Interactions dynamiques entre l'activité de maintenance et le SS

Transition	Concept en interaction	Transition forcée engendrée
Début maintenance (politique de maintenance)	Equipe de ressources	Constitution (état « disponible » vers état « en constitution »)
Start job équipe	Ressources non réutilisables	Start job *i* (état *i* vers état *i-1*)
Start job équipe	Ressources réutilisables	Start job (état "disponible" vers état"indisponible"
Start job équipe	Politique de maintenance non basée sur des observations	Début AM (état « en préparation » vers état « en cours »)
Start job équipe	Politique de maintenance basée sur des observations	Début obs (état « en préparation » vers état « en cours »)
Start job équipe	Politique de maintenance basée sur des observations	Début act (état « en préparation » vers état « en cours »)
Fin AM (active ou observation)	Ressource réutilisable	End job (état « indisponible » vers état « disponible »
Fin AM (active ou observation)	Equipe de ressource	End job équipe (état « en_travail » vers état « disponible »)

II.6.3 Cas des interactions dynamiques entre le SP et le contexte

Sur ce point, nous considérons uniquement un SP formé d'un seul composant. En effet, les relations entre contexte et chaque composant d'un système sont analogues.

Les conditions opérationnelles et environnementales ont pour effet de ralentir ou d'accélérer la cinétique de dégradation. Leur évolution va donc impacter un ou plusieurs mécanismes d'un composant. Concernant les modèles physiques de dégradation, (Yu et al. 2013) intègrent directement ces facteurs dans l'équation de dégradation, ce qui nécessite d'avoir une idée précise des conditions futures et/ou un REX important et suffisamment précis. Concernant les modèles stochastiques de dégradation, (Brissaud et al. 2010) prennent en compte cet impact en considérant cette évolution dans le taux de dégradation ou de défaillance du matériel, ce qui nécessite également d'avoir une idée

précise des conditions futures et/ou un REX important et suffisamment précis. Ainsi, cette interaction est paramétrique (section II.7.).

A contrario, les chocs vont impacter directement la défaillance du composant. D'après (Huynh et al. 2011) ils entraînent obligatoirement la défaillance de leurs modes de défaillance associés. Ainsi, la transition forcée « **occurrence forcée** » du mode de défaillance associée doit être déclenchée lors de l'occurrence de la transition « déclenchement » (cf. Figure 18) du processus de chocs.

Le Tableau 12 résume les différentes interactions dynamiques entre le SP et le contexte.

Tableau 12 – Interaction dynamique entre le SP et le contexte

Transition	Concept en interaction	Transition forcée engendrée
Déclenchement (choc)	Mode de défaillance	Occurrence forcée (état « non apparu » vers état « apparu »)

II.6.4 Cas des interactions dynamiques internes au SP

Ces interactions sont relatives aux constituants d'un SP et peuvent être de natures très différentes. Il est également nécessaire à ce niveau d'identifier les interactions dynamiques liées à la politique de maintenance opportuniste.

Ces interactions peuvent être fonctionnelles. Sous ce terme nous désignons les interactions fonctionnelles transversales et liées à un niveau de granularité donné. Par exemple, l'énergie fournie par un composant peut être fonction de l'énergie fournie par le composant en amont dans la chaine fonctionnelle. Ces interactions peuvent également être hiérarchiques, c'est-à-dire que les performances d'un composant de niveau supérieur est fonction de l'ensemble des fonctions du niveau inférieur. Les études HAZOP peuvent être exploitées pour modéliser qualitativement ces liens. (Cocheteux 2010) utilise un ANFIS, ou réseau de neurones flou, pour projeter la performance de niveau système en fonction des performances de niveau sous système. Il peut également exister des systèmes bouclés, bien que ceux-ci soient principalement de nature électrique, qui sont par définition « un système avec au moins un composant dont l'état dépend d'un autre et vice versa ». Ce type de système peut entraîner une complexification importante des modèles (Bouissou & Seguin 2006). L'état de la fonction dépend donc de l'état du composant qui la supporte. Cette interaction est paramétrique lorsqu'il s'agit d'un mécanisme de dégradation ou d'un symptôme (le taux d'occurrence d'une dégradation fonctionnelle est impactée par l'état du mécanisme) et dynamique lorsqu'il s'agit de l'état du composant : l'occurrence d'une panne ou d'une maintenance sur celui-ci provoquera l'arrêt de la fonction et l'occurrence d'une transition forcée sur celle-ci, depuis l'état courant vers l'état « non remplie ». De même, le retour à l'état marche entraînera le retour de la fonction dans un état de fonctionnement. Ainsi, les transitions forcées « **arrêt fonction i** » et « **reprise fonction i** », l'indice i modélisant respectivement l'état de dégradation fonctionnel depuis lequel la fonction s'arrête et depuis lequel la fonction repart, peuvent être définis selon les transitions du composant définies Tableau 10.

Des composants peuvent être dits en dépendance physique. Ainsi, la dégradation ou la défaillance d'un composant peut influer sur un composant voisin. C'est le cas notamment des travaux relatifs aux défaillances de causes communes (Hoepfer et al. 2009), pour lesquelles des composants différents

vont tomber en panne en même temps. Ainsi un mécanisme de dégradation ou un mode de défaillance peut impacter son homologue sur un composant voisin. Ces interactions sont donc similaires aux interactions présentes dans un unique composant, et ces transitions forcées ont déjà été définies Tableau 10. Il est également nécessaire à ce niveau de prendre en compte les défaillances à la sollicitation qui vont impacter directement la défaillance du composant. En général, il est difficile de modéliser physiquement les mécanismes d'une défaillance à la sollicitation. Ainsi, une façon réaliste de modéliser des défaillances à la sollicitation est d'attribuer une probabilité de défaillance à la sollicitation à chaque composant (Betous-Almeida & Kanoun 2004b). Ainsi, chaque fois qu'un composant redémarre (suite à la fin de sa maintenance par exemple), alors il existe une probabilité pour qu'il défaille immédiatement. Cette interaction est donc à la fois dynamique et conditionnée. Une transition forcée du mode de défaillance peut ainsi être conditionnée à l'occurrence des évènements « fin panne » et « fin maintenance préventive » du composant, que nous notons « **occurrence forcée conditionnée** ».

Des composants peuvent être dépendants d'un point de vue architectural. Ce sont le cas de composants dont le mode de fonctionnement peut être dépendant du mode de fonctionnement d'autres composants. Nombreux sont les systèmes utilisant des redondances passives ou actives par exemple (Kehren 2005). Cette réalité implique, en cohérence avec nos travaux, la nécessité d'une définition d'états supplémentaires pour le composant (états d'attente notamment) et de caractéristiques également (conditions de redondances, de déclenchement…). L'arrivée dans les états d'attente provoque l'arrêt de la fonction et l'occurrence d'une transition forcée sur celle-ci, depuis l'état courant vers l'état « non remplie ». De même, la sortie d'un état d'attente provoquera le retour de la fonction vers un état de marche. Cependant, ces cas sont liés à des systèmes particuliers, et ne doivent être définis que pour un système spécifique.

Au niveau de la politique de maintenance opportuniste, celle-ci se déclenche d'après (E. Thomas 2009) lorsqu'une politique de maintenance liée, quelle qu'elle soit, débute sa préparation. Ainsi, chaque transition « début maintenance » entraînera, au niveau de la politique opportuniste, l'occurrence de la transition forcée « **début maintenance** ».

Le Tableau 13 résume les différentes interactions dynamiques du SP.

Tableau 13 – Interactions dynamiques internes au SP

Transition	Concept en interaction	Transition forcée engendrée
Début maintenance préventive (composant)	Fonction	Arrêt fonction *i* (état de fonctionnement i vers état « non remplie »)
Début panne (composant)	Fonction	Arrêt fonction *i* (état de fonctionnement i vers état « non remplie »)
Fin maintenance préventive (composant)	Fonction	Reprise fonction *i* (état « non remplie » vers état de fonctionnement *i*)

Fin panne (composant)	Fonction	Reprise fonction *i* (état « non remplie » vers état de fonctionnement *i*)
Début maintenance (politique de maintenance en général)	Politique de maintenance opportuniste	Début maintenance (état « en attente » vers état « en préparation »
Fin maintenance préventive (composant)	Mode de défaillance (défaillance à la sollicitation)	Occurrence forcée conditionnée (état « non apparu » vers état « apparu »)
Fin panne (composant)	Mode de défaillance (défaillance à la sollicitation)	Occurrence forcée conditionnée (état « non apparu » vers état « apparu »)

Les différentes transitions forcées relatives aux différents concepts ont à présent été définies de manière cohérente avec leurs interactions dynamiques. Ces interactions peuvent être instantanées, et leur franchissement peut parfois être conditionné. Cette section a permis aussi de dégager des interactions paramétriques entre les caractéristiques des différents concepts.

II.7. Identification des interactions paramétriques entre les différents concepts de connaissance

Chaque caractéristique peut être fonction ou non d'autres caractéristiques des concepts associés. Nous en avons identifié dans le paragraphe précédent, notamment concernant l'aspect dysfonctionnel des composants.

Les symptômes et mécanismes de dégradations sont par nature un phénomène continu. Par exemple, (Monnin et al. 2011) ou (M. Marseguerra et al. 2003) prennent en compte leur impact sur leur mode de défaillance associé sous la forme d'un impact sur l'occurrence de celui-ci. Ainsi, chaque changement d'état (« évolution 12 », Figure 15 par exemple) n'entraîne pas de transition forcée. Cette interaction est donc paramétrique, et les **lois et paramètres d'évolution** des symptômes et mécanismes de dégradation sont en effet dépendants des états de dégradations des mécanismes de dégradation associés, ainsi que de l'intensité (et donc des états) des conditions environnementales et opérationnelles éventuellement associées.

De même, les **lois et paramètres d'occurrence** des modes de défaillance dépendent des états de dégradations des mécanismes de dégradation associés, ainsi que de l'intensité des conditions environnementales et opérationnelles éventuellement associées.

Au niveau des **probabilités de dégradation fonctionnelle**, ces facteurs sont également fonction des mécanismes de dégradation du composant les supportant.

Les **flux sortants** d'une fonction sont quant à eux fonctions de l'état fonctionnel, et donc celui de la fonction de transfert du composant, et des flux entrants de celle-ci (Cocheteux, P., Voisin, A., Levrat, E., Iung 2009). Or, ces flux entrants sont équivalents aux flux sortants de la fonction en amont. Ces

grandeurs sont donc dépendantes de caractéristiques d'un objet extérieur et sont donc sujets à une interaction paramétrique.

Il existe également de nombreux modèles permettant le calcul des **qualités des AM** imparfaites :

- (Nakagawa, 1979) a proposé la règle (p,q). De la même manière que pour les AM d'observations, cette règle consiste à attribuer deux probabilités complémentaires à chaque AM d'être ABAO ou AGAN. Ce modèle est par ailleurs établi comme étant le seul permettant de modéliser la qualité des AM d'observation, à travers une probabilité de non détection et une probabilité de fausse alarme,
- (Kijima, 1988) propose un facteur d'amélioration prenant en compte l'âge virtuel du composant comme impactant la qualité des AM lui étant affectées,
- (Doyen & Gaudoin 2004) vont plus loin dans cette voie, puisqu'ils proposent non seulement une réduction de l'intensité des AM en fonction de l'âge, mais aussi une réduction de l'âge virtuel du composant en fonction de la maintenance.

Les méthodes de calcul de ces différents paramètres sont principalement fonction de la qualité ou de la compétence du SS (Léger, 2008) ainsi que bien sûr du type d'AM appliquée (par exemple, pour (Cha & Kim 2001) les réparations sont supposées imparfaites et tandis que pour (Biswas & Sarkar 2000) les remplacements sont supposés parfaits).

La **durée de l'AM**, dépend principalement de la capacité du composant à être maintenu et de l'apport du système logistique du système principal. Concernant les actions de maintenance actives, cette notion est traditionnellement estimée dans les travaux relatifs à la sûreté de fonctionnement à travers l'une des propriétés d'un composant : sa maintenabilité, souvent modélisée par un taux de réparation exponentiel μ (Dutuit et al. 1997) (Betous-Almeida & Kanoun 2004b). Ce taux, sans toutefois considérer les éventuels délais logistiques (ressources indisponibles…), peut également intégrer, à la manière de taux de défaillance opérationnel, la qualité des ressources. Concernant les actions de maintenance d'observation, par définition, seuls les contrôles entraînent un arrêt du matériel. Cette durée peut être modélisée à travers un taux constant analogue à la maintenabilité. Les travaux proposant une modélisation plus fine de cette durée prennent en compte l'impact du système de soutien, importante dans le contexte de la CMPQ.

Dans le domaine de l'évaluation des politiques de maintenance, peu de travaux ont considéré le temps de réparation, de maintenance préventive ou les ressources de la maintenance. Un grand nombre fait l'hypothèse que le temps de réparation est négligeable ou que les ressources sont toujours disponibles (Wang, 1999). De plus, le REX disponible sur ces points est souvent très limité. Cependant, la spécificité de la CMPQ par rapport aux études de sûreté de fonctionnement classiques est la prise en compte du système de soutien. Ainsi, cette durée est également fonction de la disponibilité des ressources, réutilisable ou non (Monnin et al. 2011) (Medina Oliva 2011). Ce point est pris en compte pour la CMPQ par l'intermédiaire de l'état de préparation (interactions paramétriques).

Concernant les politiques de maintenance préventives, leurs **périodes** peuvent être variables selon la synchronisation ou non des politiques. En effet, ces périodes doivent pouvoir se réactualiser si par exemple un composant est sujet à une maintenance corrective, et ce selon certaines règles. Ces politiques ou activités peuvent également parfois être priorisées (si un seul opérateur est disponible, et

que deux possibilités d'AM s'offrent à lui, quelle AM va-t-il effectuer en priorité ?) (Moore & Starr 2006).

Les **conditions de déclenchement** d'une politique de maintenance conditionnelle ou prévisionnelle sont également fonction des états de l'élément dysfonctionnel du composant sur lequel leur AM d'observation s'applique. Par exemple, si la politique de la politique est conservative, alors l'AM active associée s'appliquera si un élément dysfonctionnel est observé dans un faible état de dégradation.

Enfin, les conditions de déclenchement du travail d'une équipe de ressources sont liées à la disponibilité des ressources qui la compose. Si l'ensemble des ressources nécessaires est disponible, l'équipe peut commencer son travail.

Le Tableau 14 résume les différentes interactions paramétriques et les caractéristiques des concepts qu'elles impliquent.

Tableau 14 – Interactions paramétriques : relations entre caractéristiques

Caractéristique (concept associé)	Caractéristique impactant (concept associé)
Loi et paramètre d'évolution (mécanisme de dégradation/ symptôme)	Etat (mécanisme de dégradation) Etat (condition environnementale) Etat (condition environnementale)
Loi et paramètre d'occurrence (mode de défaillance)	Etat (mécanisme de dégradation) Etat (condition environnementale) Etat (condition environnementale)
Loi et paramètre d'évolution (fonction)	Etat (mécanisme de dégradation)
Flux sortants (fonction)	Etat (fonction) Flux sortants (fonction amont)
Durée AM active ou d'observation (politique de maintenance)	Compétence/ qualité (ressources réutilisable ou non) Maintenabilité (mode de défaillance/ mécanisme de dégradation) Type AM
Qualité AM active ou d'observation (politique de maintenance)	Compétence/ qualité (ressources réutilisable ou non) Type AM
Périodes (politique de maintenance préventive)	Période (politique de maintenance)
Conditions de déclenchement (politique de maintenance conditionnelle/ prévisionnelle)	Etat (mécanisme de dégradation/symptôme)

Conditions de déclenchement (équipe d'opérateurs)	Etat (ressources réutilisables/ ressources non réutilisables)

II.8. Synthèse des liens entre la connaissance génériques et les KPIs

Il convient à présent de faire le lien entre les indicateurs répertoriés au chapitre 1 et les éléments de connaissance génériques répertoriés ci-dessus. En effet, l'objectif est bien de générer des règles de calcul des KPIs à partir des concepts. Ces règles permettent ainsi, lors de la phase de simulation du modèle exécutable de calculer les KPIs et ainsi de conclure quant à la CMPQ. Pour rappel, ces indicateurs d'intérêt peuvent se répartir en trois catégories : les indicateurs relatifs aux équipements (disponibilité, taux de production, qualité et nombre d'arrêts), les indicateurs relatifs aux couts (cout de maintenance, cout de production) et les indicateurs relatifs à la maintenance (tâches de maintenance planifiées, tâches de maintenance non planifiées).

La disponibilité moyenne A d'un composant est donnée par l'équation $A = 1 - \bar{A}$, $\overline{A}$ étant l'indisponibilité moyenne d'un composant. $\overline{A}$ est donc la probabilité du composant/système d'être en incapacité remplir leur fonction et est donc calculable à partir du temps d'attente du concept de fonction dans l'état « non_assurée ». Cette indisponibilité peut être décomposée en une indisponibilité non fortuite (programmée) et une indisponibilité fortuite. L'indisponibilité fortuite est donnée par le rapport entre le temps pendant lequel le concept de composant/système se trouve dans l'état « en_panne » et le temps total de fonctionnement, tandis que l'indisponibilité non fortuite est donnée par le rapport entre temps durant lequel le concept de composant/système se trouve dans l'état « en_maintenance_prev » et le temps total de fonctionnement. A un niveau système, il est nécessaire de prendre en compte la criticité des composants. L'évaluation de l'indisponibilité de niveau système est donc liée à chaque système particulier.

Le taux de production s'évalue quant à lui à un niveau composant et nécessite, par définition, de prendre en compte les temps de réglages, les changements de séries des outillages, ainsi que d'autres données relatives à la production. Notre démarche de modélisation étant orientée CMPQ, ces données n'ont pas été attachées à des concepts Cependant, en ajoutant l'état « en_réglage » au composant, il suffit de faire le rapport entre le temps que celui-ci passe dans cet état et son temps total de fonctionnement hors maintenance et pannes. Les transitions vers cet état seraient données par un planning de production fixé a priori.

La qualité de la production dépend directement de la fonction des composants/systèmes. Si elle est dans son état « nominal », alors la qualité du produit est parfaite. Sinon, elle se dégrade proportionnellement à l'état de dégradation dans lequel elle se trouve. Cette qualité dépend donc de la manière donc le concept de fonction a été défini (nombre d'états, signification de chaque état…) et est donc propre à un système particulier.

Si les couts relatifs à la production dépendent de données relatives à la production, les couts globaux de maintenance sont évaluables par les caractéristiques des concepts identifiés et dépendent :

- Du coup de possession C_i des ressources i en interactions avec le SP,
- Du coup des ressources C'_j non réutilisables j consommées (leur nombre est donné par le nombre de fois qu'une transition déclenchant le lancement de toute politique utilisant une ressource non réutilisable j est franchie),

- Du cout d'indisponibilité du système C_{ind}.

Les indicateurs relatifs à la maintenance, et plus spécifiquement le nombre de tâches de maintenance planifiées ou non, peuvent être détaillés et spécifiques à chaque politique de maintenance définie. Ces indicateurs deviennent alors le nombre de fois que la transition modélisant le début d'une politique de maintenance est déclenchée.

Le Tableau 15 synthétise ces règles de calcul. Nous utilisons la notation *T(x,A)* pour modéliser un temps d'attente du concept *A* dans l'état *x* et la notation *N(x,A)* pour modéliser le nombre de transition *x* tirées dans le concept *A*.

Tableau 15 – Synthèse des règles de calcul des KPIs

Indice	KPI	Règle
R1	Disponibilité composant	*1-(R2+R3)*
R2	Indisponibilité fortuite composant	$\frac{T(en\_panne,\ composant)}{T_{tot}}$
R3	Indisponibilité programmée composant	$\frac{T(en\_maintenance\_prev,\ composant)}{T_{tot}}$
R4	Taux de production composant	$\frac{T(en\_réglage,\ composant)}{T(en\_réglage,\ composant) + T(en\_marche,\ composant)}$
R4	Cout global de maintenance	$\sum_i C_i + C_{ind} + \sum_j N(debut\_AM, stratégie\ de\ maintenance_j) * C'_j)$
R5	Nombre de tâches de maintenance planifiées	$N(debut\_AM, strategie\ de\ maintenance\ préventive)$
R6	Nombre de tâches de maintenance non planifiées	$N(debut\_AM, strategie\ de\ maintenance\ corrective)$

II.9. Conclusion

Ce chapitre a mis en évidence l'ensemble des connaissances métiers génériques et leurs interactions à prendre en compte pour réaliser une étude de CMPQ. Ils sont schématisés Figure 19, en compagnie des KPIs quantifiables par les différents axes de connaissance (SP, SS, contexte et activité de maintenance). Cette connaissance est représentée sous la forme de concepts génériques en regard de leur point de vue :

- Statique : des caractéristiques fidèles à la connaissance de la réalité ont été identifiées pour chaque concept. Chacune de ces caractéristiques prend en compte les caractéristiques des autres concepts dont elles peuvent être fonction,
- Comportementales : des représentations états-transitions fidèles à la connaissance de la réalité ont été identifiées pour chaque concept. Chacune de ces représentations prend en compte les transitions forcées déclenchées par l'occurrence de transitions dans d'autres concepts par l'intermédiaire de la définition des interactions dynamiques et paramétriques entre les concepts.

Les besoins liés à la formalisation de ces vues statiques et comportementales permettent de préciser le prérequis de modélisation de la connaissance générique relative à la CMPQ défini au chapitre 1. En effet, trois types de transitions doivent pouvoir être modélisés afin de modéliser les phénomènes comportementaux relatifs à un problème de CMPQ :

- la possibilité de modéliser des transitions déclenchées par d'autres concepts, et donc les transitions forcées ou simultanées,
- la possibilité de modéliser des transitions pouvant faire l'objet d'une condition (cf. Figure 11 et Figure 12), ou transitions conditionnées,
- la possibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul.

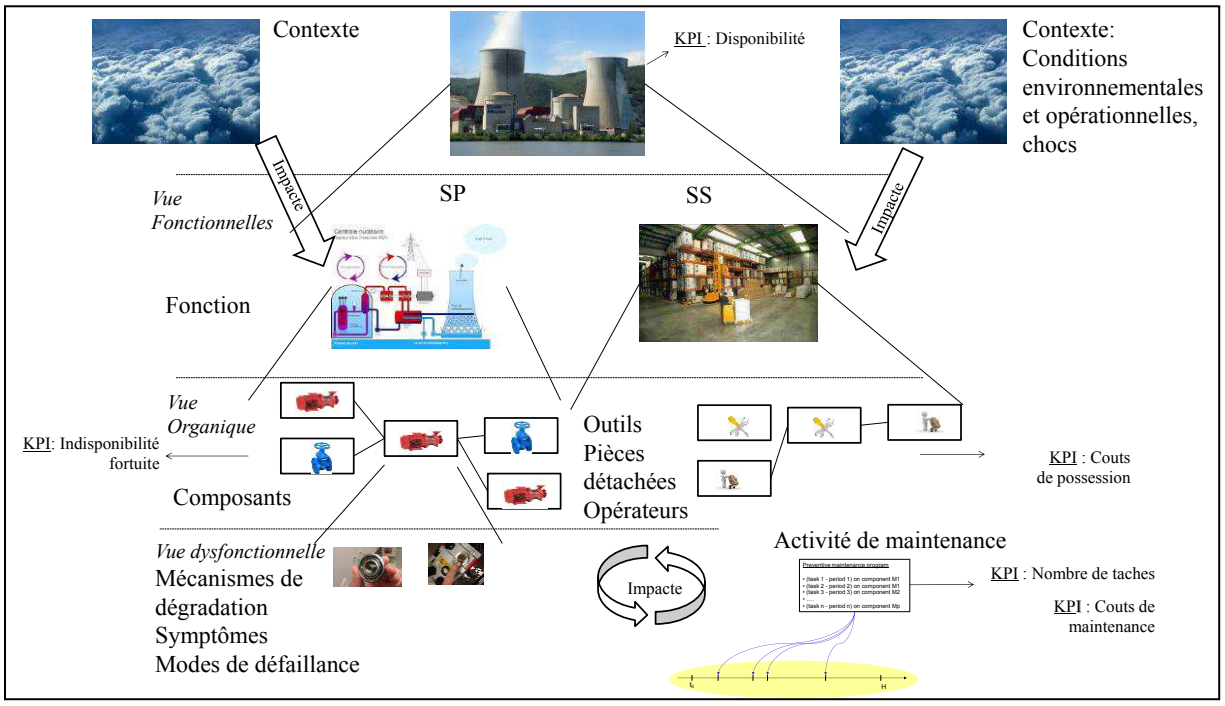


Figure 19 – Connaissance, interactions et KPIs relatifs à la CMPQ

En ajoutant ces prérequis aux prérequis définis au chapitre 1 liés à l'utilisabilité de l'outil un ensemble de critères pour le choix d'un ou plusieurs langages permettant de formaliser la connaissance générique peut être défini. Ce choix est discuté au chapitre 3.

Chapitre III. Etat de l'art sur les langages et méthodes permettant de supporter les critères de la CMPQ

III.1. Introduction

Le chapitre 2 a permis de répondre partiellement à l'étape (1') de la démarche pour la simulation des KPIs de la CMPQ, par l'identification de la connaissance générique métier nécessaire à leur évaluation. Or, il est à présent nécessaire de capitaliser cette connaissance pour permettre sa réutilisation. Pour ce faire, il est nécessaire de s'intéresser aux langages permettant de supporter une telle capitalisation, et de les comparer en regard des prérequis définis aux chapitres 1 et 2. Ces prérequis sont les suivants :

- la possibilité de modéliser des transitions forcées ou simultanées,
- la possibilité de modéliser des transitions conditionnées,
- la possibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul,
- la capacité à obtenir un modèle vérifiable et validable,
- la capacité à réduire l'effort de modélisation pour chaque étude, et donc la modularité du langage,
- la capacité du langage à être exploité par un utilisateur sans connaissance de ce langage,

Ces critères sont inclus dans ceux définis par (Bouissou, 2007), qui définit 7 critères permettant d'identifier la capacité de ces langages à gérer la complexité des systèmes: la puissance de modélisation, l'aptitude à l'abstraction, l'aptitude à la factorisation et la généralisation, l'aptitude à la projection, la composabilité, l'aptitude à la transformation, l'intérêt de la représentation graphique et leur aptitude aux traitements automatisés.

Par rapport à ces différents critères, l'objectif de ce chapitre est d'identifier le ou les différents langages les plus adéquats pour à la fois permettre la réalisation de l'étape (1') de la démarche, à savoir modéliser la connaissance générique, la réalisation de l'étape (2), mais aussi la réalisation de l'étape (3), à savoir l'évaluation des KPIs. Pour ce faire, deux grandes familles d'approches pour la CMPQ seront identifiées, et les avantages et inconvénients de chacune sont étudiés. Nous argumentons la solution d'utiliser deux langages différents. Ensuite, une étude comparative des langages permettant la simulation ou la capitalisation de connaissances similaires est réalisée. Nous détaillons alors le choix des langages SysML et AltaRicaDF (ADF) pour la modélisation générique de la connaissance (étape 1'). Le langage SysML permet de modéliser les concepts de connaissance de manière fidèle à la connaissance identifiée chapitre 2, et le langage ADF permet à la fois d'instancier ces concepts à un cas d'étude particulier (étape (2)) et d'effectuer des simulations stochastiques pour la réalisation de l'étape (3).

III.2. Les approches de la quantification des programmes de maintenance complexes

Conformément aux différentes étapes identifiées au chapitre 1, le choix du ou des langages les plus adaptés doit être fait selon deux critères. Tout d'abord, il doit permettre la modélisation de la connaissance identifiée au chapitre 2 en évitant les pertes sémantiques afin de compléter l'étape (1') de

création d'un modèle générique. De plus, il doit permettre d'effectuer des simulations afin d'effectuer la quantification à proprement parler des KPIs sur la base des règles définies Tableau 15, et donc de réaliser l'étape (3). Ces critères sont-ils antagonistes ?

Afin de répondre à cette question, deux types d'approches existantes ont été identifiées. Le premier type d'approche dispose d'un modèle générique de la connaissance préalable, avant de transformer ce modèle générique en un modèle de simulation plus spécifique souvent modélisé dans un langage formel différent de celui utilisé pour capitaliser le premier modèle (Zille 2009), (Monnin et al. 2011), (Bernardi et al. 2002). Les langages utilisés pour modéliser la connaissance générique sont en effet des langages décorrélés de tout formalisme de simulation stochastique, et donc n'ayant pas vocation à effectuer des simulations de KPIs.

Ces travaux sont dits guidés par une démarche d'ingénierie basée sur les modèles (MDBI), fidèles aux enjeux de la CMPQ, puisque possédant une étape d'identification de la connaissance préalable à construction du modèle.

Le second type d'approches se caractérise par l'utilisation d'un seul langage pour modéliser la connaissance générique et la simuler. Des hypothèses sont ainsi réalisées sur la connaissance à simuler, afin d'adapter la connaissance aux capacités sémantiques du langage ((Betous-Almeida & Kanoun 2004a), (Clavareau & Labeau 2009a), (Boiteau et al. 2006)).

III.2.1 Les approches utilisant un unique langage pour les deux types de modélisation

Ces travaux sont souvent relatifs aux problématiques de gestion de la complexité des systèmes. En effet, un langage permettant la simulation stochastique est choisi, et des simulations sont effectuées sur un cas d'étude. Ce type d'approche est construit sur le principe d'avoir un modèle qui permet à la fois de modéliser la connaissance et de l'exécuter pour évaluer les KPIs.

Par exemple, (Betous-Almeida & Kanoun 2004b) propose une démarche de construction générique basée sur l'utilisation de Réseaux de Pétri. Cependant, son cas d'étude étant la disponibilité d'un système d'information, la maintenance est supposée parfaite et instantanée. De plus, seules les politiques correctives sont considérées, sans prise en compte du système de soutien.

(Boiteau et al. 2006) va plus loin puisqu'il considère un système de soutien agissant sur le système principal. Il utilise ensuite le langage AltaRicaData Flow (ADF) pour formaliser ses atomes de modélisations représentant les éléments constitutifs de son système.

(Clavareau & Labeau 2009b) crée un ensemble d'atomes de modélisation dans le formalisme des Réseaux de Pétri (RdP). Pour ce faire, il effectue des hypothèses sur la connaissance générique relative à son problème, à savoir la maitrise de l'obsolescence des constituants d'un système de production. Cependant, des hypothèses non fidèles à la sémantique résident malgré tout (non prise en compte du contexte, taux de défaillance constants…). Elles permettent cependant de transformer la connaissance simplement en RdP.

(Piero Baraldi et al. 2011) utilise un modèle flou pour modéliser l'évolution d'un composant sujet à des facteurs influents (maintenance, conditions environnementales…). De la même manière, des hypothèses sont effectuées sur cette connaissance afin de la rendre compatible avec le modèle flou.

Ainsi, ce type d'approches est efficace pour :

- Créer un modèle simulable d'un système particulier,
- Comparer l'effort de modélisation selon langage utilisé.

Cependant, ces approches présentent un inconvénient majeur. Elles ne permettent pas de modéliser la connaissance générique sans pertes sémantiques par rapport à la connaissance métier identifiée au chapitre 2, et donc de compléter l'étape (1') de création d'un modèle générique, puisque le formalisme choisi n'est généralement pas adapté pour totalement modéliser la connaissance métier. En effet, de par leur capacité sémantique, les langages dédiés à la simulation stochastique ne permettent pas de retranscrire de manière fidèle la connaissance métier.

III.2.2 Les approches utilisant un langage par type de modélisation

Sur ce type d'approche, (Monnin, 2007) propose un modèle statique formalisé par le biais des diagrammes de classes UML. Il transforme ensuite, par l'intermédiaire de règles de correspondances systématiques, ce modèle vers des Stochastic Activity Network (SAN) atomiques. Cependant, le champ de modélisation proposé parait trop orienté sur les systèmes de combat, cas d'étude de ce travail, pour être réutilisé. De plus, il ne propose pas de vue dynamique générique rendant ainsi opaque la cohérence des atomes de modélisations formalisés.

Dans le même souci de formalisation amont (avant l'utilisation d'un langage permettant la simulation stochastique) de la connaissance (Medina Oliva, 2011) propose également un modèle statique formalisé par des diagrammes de classes UML. Cependant, l'auteur ne choisit pas un formalisme à état discret pour la simulation stochastique puisqu'il préfère utiliser les Réseaux Bayésiens, ce qui conduit à une transformation de langage peu évidente et informelle depuis le modèle initial UML.

Concernant les deux travaux, il est intéressant de noter que la formalisation d'une connaissance générique permet ensuite de réutiliser cette connaissance par instanciation pour chaque étude. Cependant, les auteurs préfèrent se concentrer sur la modélisation statique (et non dynamique du problème).

En complément, (Zille 2009) propose des modèles atomiques dynamiques génériques cohérents avec les normes et standards relatifs à la maintenance. Ces modèles sont représentés par l'intermédiaire de représentations graphiques astucieuses mais non formelles. Ils sont ensuite transformés de manière informelle vers des RdPS. La cohérence du modèle final est alors remise en cause par l'absence de cadre sémantique permettant d'assurer la cohérence entre les représentations génériques et le langage permettant la simulation stochastique.

L'ensemble de ces travaux se caractérisent donc par deux niveaux d'abstraction bien distincts :

- Un niveau modélisation de la connaissance, non destiné à la simulation, évitant les pertes sémantiques par rapport à la connaissance métier identifié chapitre 2. Aucun modèle normé n'étant

assez fin pour remplir ces prérequis, chaque auteur s'attâche à proposer le sien. Cependant, les modèles proposés sont soit incomplets dans le contexte de la CMPQ, soit rédigés en langage naturel,

- Un niveau de simulation de la connaissance à l'aide d'un langage formel adapté à la simulation stochastique. Il permet ainsi l'obtention des KPIs d'intérêt, même s'il peut entraîner des pertes sémantiques, au moment d'établir les correspondances entre les deux langages, par rapport au modèle initial de la connaissance. Pour cela, de nombreux langages, identiques à ceux des approches utilisant un même langage pour chaque niveau d'abstraction, existent.

La difficulté de ce type d'approche réside donc dans le passage d'un langage à l'autre. Les travaux proposent parfois des règles informelles de transformation (Zille 2009) rendant délicate la vérification de la correspondance sémantique entre les deux langages. Cependant, (Bernardi et al. 2002), par exemple, établi des correspondances précises entre deux langages, fournissant ainsi un support à une éventuelle implantation de la transformation de langages. Ce type d'approche pose donc ouvertement la problématique scientifique de transformation de modèles, induisant la considération d'un nouveau prérequis sur les langages : leur aptitude à avoir leur connaissance être exploité par un autre langage.

Afin de déterminer la meilleure adéquation d'une approche par rapport à l'autre, ou l'intérêt de passer par une formalisation de la connaissance décorrélée de tout formalisme de simulation stochastique, il est d'aborder nécessaire de décrire les langages supports à la simulation stochastique, et analyser dans quelles mesures ils permettent de satisfaire les prérequis de la CMPQ.

III.3. Les langages pour la simulation stochastique

La simulation stochastique a été identifiée comme le type de simulation adéquat pour la quantification des KPIs requis pour la CMPQ. En effet, les langages permettant de supporter ce type de simulation sont particulièrement adaptés lorsque l'on souhaite traiter les aspects de dépendances entre composants et de système réparable, quitte à conclure sur un calcul lourd en termes de simulation. Ces langages sont traditionnellement utilisés par les travaux relatifs aux études de SdF.

III.3.1 Les langages classiques de la SdF

Ces langages peuvent se regrouper en deux familles : les langages à espace d'état discret, et les autres. (Muppala, Fricks, & Trivedi, 2000) se sont attachés (cf. Figure 20) à établir les liens entre ces différents formalismes. Les flêches d'implication mettent en évidence la prise en compte d'hypothèses supplémentaires par le langage ciblé, telles que l'hypothèse Markovienne par exemple. Cependant, si perfectionné soient-ils, l'utilisabilité du langage (environnement de calcul existants, généricité…) doit être également un critère clé à prendre en compte.

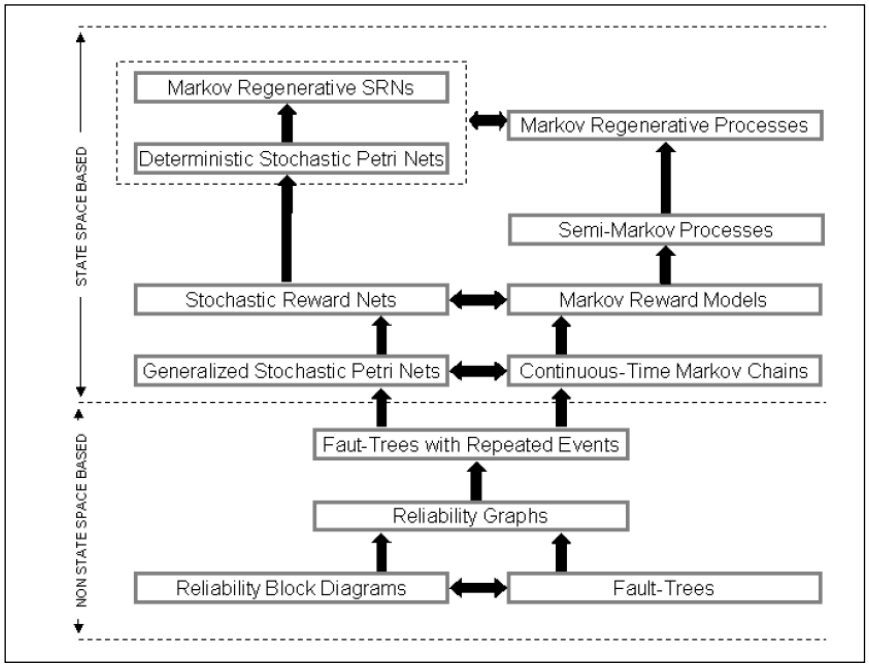


Figure 20 – Equivalence entre les techniques de modélisation (Muppala, Fricks, & Trivedi, 2000)

Parmi les langages à espaces d'états discrets, les chaines de Markov sont le langage de base et elles permettent de représenter un grand nombre de phénomènes sous l'hypothèse markovienne, et permettent une résolution analytique de problème modélisé. Certaines de leurs extensions permettent de dépasser cette propriété markovienne (processus semi markoviens). Cependant, ces langages restent malgré tout très sensibles à l'explosion combinatoire. C'est pourquoi ils sont peu utilisés aujourd'hui dans la représentation des systèmes complexes, hormis sous l'affirmation d'hypothèses entraînant un fort effort de modélisation pour chaque étude (Nowakowski & Werbinka 2009).

En effet, Les réseaux de Pétri (RdP) et ses extensions dédiées à la SdF (Stochastic Activity Network, SAN, RdP Stochastiques Généralisés, RdPSG) sont plus appréciés pour obtenir des KPIs à partir de modèles interactionnels complexes. Ils peuvent représenter n'importe quel processus à état discret, autorisant ainsi une puissance de modélisation élevée, et disposent de nombreux algorithmes efficaces.

Par exemple, (Betous-Almeida & Kanoun 2004b) utilise une méthode de construction de modèles basée sur les RdPSG afin de calculer la disponibilité d'un système d'information. (Monnin et al. 2011) utilise les SAN pour calculer la disponibilité opérationnel d'un système de combat. (Zille 2009) et (Clavareau & Labeau 2009b) utilisent les RdPSG pour calculer des indicateurs sur un système de production manufacturier. Cependant, si celle-ci reste possible, la réutilisabilité de concepts dynamiques formalisés en réseau de Pétri peut être fastidieuse pour l'utilisateur. En effet, il est nécessaire de connaitre le langage pour paramétrer ce genre d'interactions.

D'autres langages, à espaces d'états non discrets, sont classiquement utilisés pour les applications liées à la sûreté de fonctionnement, parmi lesquels les arbres de défaillances, les diagrammes de fiabilité, les réseaux bayésiens ou les réseaux bayésiens dynamiques. Parmi ceux-ci, seuls les réseaux bayésiens et les réseaux bayésiens dynamiques permettent de prendre en compte la dimension temporelle. Par exemple, (Medina Oliva, 2011) utilise le formalisme PRM (Probabilist Relational Model) issu des réseaux bayésiens et cette dimension temporelle pour calculer la disponibilité d'un

système manufacturier de production de ferment. Ceux-ci sont particulièrement utiles pour composer les systèmes, étant donné la possibilité de définir des lois de probabilités conditionnelles paramétrables.

Cependant, seuls de petits sous modèles peuvent être réutilisés (Bouissou, 2007), induisant ainsi une contrainte en termes d'utilisabilité du langage, et notamment concernant la réduction de l'effort de modélisation.

Les réseaux bayésiens dynamiques (Weber & Jouffe 2006) permettent de calculer les états du système en fonction de ces états précédents, en plus de ses évolutions. Cependant, ce mode de calcul augmente considérablement le combinatoire et impacte le temps de calcul de manière conséquente. Mais ils présentent cependant une supériorité sur les réseaux bayésiens classiques en termes de puissance de modélisation, bien que possédant le même problème par rapport à la méthode d'identification de la connaissance proposée.

En conclusion, ces langages, si utiles et utilisés soient-ils, font globalement preuve d'un manque d'aptitude à la généralisation et à la composabilité. La construction de modèles de niveau système reste fastidieuse, et ils sont souvent décorrélés de méthodes de construction rigoureuse et adaptée. Ainsi, des limites sont apparues afin de répondre aux nouveaux enjeux des grands systèmes tels que ceux sur lesquels la CMPQ se focalise. Pour cela, des langages dédiés à la sûreté de fonctionnement sont apparus plus récemment (DEVS (Zeigler, 1998), SDM (Ramesh et al. 1999), AltaRicaDF (A. Rauzy 2002), FIGARO (Bouissou et al. 2002)…) afin de répondre à ces nouveaux besoins en permettant la création automatique de modèles de sûreté tels que les RdP ou les arbres de défaillance par exemple, par le biais d'algorithmes de transformation transparents pour un utilisateur quelconque. Ces langages textuels, ou non graphiques, permettent donc de modéliser des concepts à un niveau d'abstraction plus élevée qu'avec toute représentation graphique, permettant ainsi à des utilisateurs de modéliser des systèmes sans connaissances du langage ni du paradigme de modélisation (Bouissou, 2007).

III.3.2 Les langages de modélisation dédiés à la SdF

Ces langages, bien que non directement simulables, sont assimilés à des langages permettant la simulation stochastique. En effet, comme ils disposent d'un niveau d'abstraction supérieur à ceux des langages classiques, leur simulation est accessible de manière transparente pour l'utilisateur, puisqu'ils sont supportés par des environnements de calculs permettant leur transformation en réseau de Pétri ou en arbre de défaillance équivalents. De par ce niveau d'abstraction élevé, ils sont également plus facilement réutilisables pour un utilisateur sans connaissance du langage. Cependant, ils souffrent également d'absence de méthode de construction cohérente, ce qui est d'autant plus important les concernant puisqu'ils ne disposent que rarement de vues graphiques.

Parmi ces langages, le langage DEVS (Zeigler, 1998) (discrete event system specification) fournit une démarche conceptuelle pour la spécification de simulations à évènements discrets d'une manière modulaire et hiérarchique. Ce formalisme est de plus supporté par de nombreux environnements de calculs dédiés.

Le langage SDM (Ramesh et al. 1999) (System Description Method) a pour but d'étendre le formalisme des RdP afin de faciliter leur création à un niveau système.

Le langage FIGARO (A. Rauzy 2002) possède les mêmes objectifs que le langage précédent en généralisant également le formalisme des BDMP (Boolean logic driven Markov Process) notamment, et dispose d'un environnement dédié : l'environnement KB3.

Le langage AltaRicaDF, extension du langage AltaRica, est à l'heure actuelle, le langage dédié à la SdF le plus étudié dans la littérature. De nombreuses publications tant industrielles (E. Arbaretier, Z. Brik, V. Brindejonc, B. Desmarquest 2010), (Brik & Secher 2010) qu'universitaires (Adeline 2011) , (Kehren 2005) lui ont été dédiés. Le langage est en évolution actuellement, et le langage STG (guarded transition system) possède quelques amélioration par rapport à ADF (prise en compte de systèmes bouclés).

Parmi ces langages, deux d'entre eux (DEVS, SDM) proposent des méthodes graphiques de construction des modèles. Ces méthodes, si efficaces soient-elles, ne permettent cependant pas la capitalisation d'une connaissance générique décorrélée de tout aspect lié à la simulation.

III.3.3 Langages dédiés à la simulation stochastique : avantages et inconvénients

En termes de capacité à modéliser avec précision la connaissance, ces langages sont à peu près équivalents, et leur efficacité à effectuer des simulations sur des systèmes particuliers sujets à des interactions complexes proches de celles de la CMPQ a été démontrée (Bouissou, 2007). La différence réside surtout dans leurs capacités à gérer la complexité. Au niveau de la réduction de l'effort de modélisation, aucun d'entre eux n'est pleinement satisfaisant. En effet, ils ne permettent que peu la réutilisation de sous modèles. Les réseaux bayésiens ont par ailleurs été pénalisés au niveau de l'aptitude à une exploitation de la connaissance modélisée par un autre modèle en raison de leur vue non états transition différente de la philosophie de représentation des concepts de connaissance du chapitre 2. En outre, si la vue graphique présente dans ces modèles aide à la validation des modèles avec un expert métier, ces ne sont que très difficilement vérifiables par model-checking (Katsaros 2009).

Nous avons décidé d'attribuer une note sur 2 à chacun des critères afin d'évaluer dans quelles mesures ces langages satisfont les précédemment définies. La note de 0 signifie que le prérequis n'est pas du tout satisfait, la note de 1 signifie que le prérequis est en partie satisfait, et la note de 2 signifie que le prérequis est complètement satisfait. Le Tableau 16 résume les conclusions de l'état de l'art.

Tableau 16 –Langages de simulation stochastique classiques et prérequis du modèle

Prérequis	Chaines de Markov	Réseaux de Pétri stochastiques	Réseaux bayésiens	Réseaux bayésiens dynamiques
possibilité de modéliser des transitions forcées	2	2	1	1

possibilité de modéliser des transitions conditionnées	2	2	2	2
possibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul	1	1	1	1
La capacité à effectuer des simulations stochastiques	2	2	2	2
La capacité à réduire l'effort de modélisation pour chaque étude	0	1	1	1
La capacité d'être exploité par un utilisateur sans connaissance de ce langage	1	1	1	1
Aptitude à une exploitation de la connaissance modélisée par un autre langage	1	2	1	1
Capacité à être vérifié et validé	1	1	1	1

Concernant les langages de modélisation dédiés à la sûreté de fonctionnement, ils paraissent nettement plus adaptés, notamment en termes de réduction de l'effort de modélisation, pour les aspects de simulation de la CMPQ. Ils permettent en effet de modéliser par éclatement de transitions et par créations d'états fictifs des lois de transitions variables et complexes, bien que seule la simulation stochastique leur soit accessible. Les aspects de fiabilité des langages FIGARO et SDM ont été pénalisés de par leur absence de sémantique bien définie. En revanche, et ce pour l'ensemble de ces langages excepté les GTS, les modèles bouclés ne sont pas simulables. Cette contrainte sera donc à fortement prendre en compte lors de la construction du modèle. Le langage ADF présente l'intérêt de pouvoir être vérifié par model-checking, mais il n'est que difficilement validable par un expert de par son absence de vue graphique exhaustive.

Ces constats sont résumés Tableau 17.

Tableau 17 – Langage de simulation stochastique et prérequis du modèle

Prérequis	ADF	DEVS	FIGARO	SDM	STG
possibilité de modéliser des transitions forcées	2	2	2	2	2

possibilité de modéliser des transitions conditionnées	2	2	2	2	2
possibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul	1	1	1	1	1
La capacité à effectuer des simulations stochastiques	2	1	2	1	2
La capacité à réduire l'effort de modélisation pour chaque étude	2	2	2	2	2
La capacité d'être exploité par un utilisateur sans connaissance de ce langage	1	1	1	1	1
Aptitude à une exploitation de la connaissance modélisée par un autre langage	2	2	1	1	2
Capacité à être vérifié et validé	1	0	0	0	1

Ainsi, les langages ADF, STG et DEVS satisfont le mieux les prérequis vitaux établis sur le modèle. Si seuls les STG permettent la simulation des systèmes bouclés, aucun environnement dédié n'est disponible. Cependant, étant donné l'extension du langage ADF, les liens entre les deux langages peuvent être plus simplement définis. En outre, un environnement dédié au langage ADF (SIMFIA) et permettant la simulation stochastique d'un tel code est disponible. Le compilateur/simulateur présent dans SIMFIA permet de donner les résultats de simulations de Monte Carlo concernant les temps d'attentes dans chaque état du modèle, ainsi que le nombre de transitions tirées, ce qui permet de supporter en partie l'évaluation des KPIs tels que définie par les règles du Tableau 15. Les KPIs liés au cout devront cependant faire l'objet d'une étape de post traitement. De même, le rayonnement actuel tant au niveau industriel (partenariat avec Dassault, Cassidian…) que scientifique (une douzaine de thèses ont utilisé ce langage (tant à un niveau fondamental (Point, 1999) (Kehren 2005), qu'à un niveau technique (Adeline 2011) (David et al. 2010)). Cette mouvance tant industrielle qu'académique du langage nous conduit à privilégier le choix d'ADF.

Cependant, l'absence de moyens simples permettant sa vérification et validation exhaustive ainsi que l'impossibilité de modéliser la connaissance métier dans son exhaustivité (impossibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul) conduit à nous intéresser aux langages décorrélés de tout formalisme de simulation stochastique, parmi lesquels certains sont plus aptes à aborder simplement cette procédure. De plus, ceux-ci peuvent permettre de modéliser des transitions complexes.

III.4. Les langages décorrélés de tout formalisme de simulation stochastique

Il est nécessaire de faire un choix, parmi les langages et techniques de modélisation décorrélés de tout formalisme de simulation stochastique existants, sur l'un d'entre eux afin de modéliser de manière générique des aspects statiques, comportementaux et interactionnels des concepts de connaissance du chapitre 2. La vue statique est principalement traitée par les langages dits de modélisation architecturale des systèmes, et la vue comportementale est traitée par les langages dits de modélisation comportementale. Enfin, la vue interactionnelle est souvent liée à des langages multi vues, pouvant présenter des méthodes permettant d'assurer une cohérence entre ces vues. Cette vue interactionnelle est d'autant plus cruciale que le chapitre 2 a montré la complexité des interactions paramétriques et dynamiques de la connaissance générique pour la CMPQ.

Nous allons tout d'abord nous intéresser aux langages de modélisation architecturale, puis aux langages permettant la modélisation comportementale.

III.4.1 Les langages de modélisation architecturale et interactionnelle

(R. Wieringa 1999) réalise un état de l'art sur une vingtaine de différents langages de modélisation architecturale (ADL). Il identifie un certain nombre de critères afin d'évaluer les différences entre ceux-ci. A l'instant de cette publication (1999) il identifie clairement le langage UML (Unified Modeling Language, (OMG, 2011)) comme le plus complet, notamment par son aspect multi-vues pour ce type d'application. Initialement développé dans le domaine informatique pour décrire l'architecture de systèmes d'informations, ce langage s'est par la suite étendu à l'IS et aux études numériques grâce à l'ajout d'annotations sémantiques. En ce sens, les projets SPT (OMG, 2005) ou MARTE (OMG, 2007) proposent des profils standards UML destinés à l'évaluation de performances de production. (Bernardi et al., 2007) propose un profil destiné à la SdF. A la différence des ontologies, ce langage propose des vues de modélisation orientées traitement. En effet, UML dispose de différents diagrammes en relation les uns avec les autres, permettant de représenter les aspects comportementaux d'objets (diagrammes de séquences, diagrammes d'états…). OCL, une extension au langage UML, permet par ailleurs de représenter des contraintes quantifiables entre les différents objets. Cette extension ne dispose cependant pas d'une vue graphique nécessaire pour comprendre et animer des modèles.

Cependant, dans le contexte de l'IS visant à modéliser conjointement SS et SP, ce langage a été étendu au langage SysML (OMG, 2010) pour couvrir naturellement le scope des systèmes physiques. SysML est né du standard AP-212 faisant état du besoin d'un langage pivot entre les différents domaines de l'ingénierie. Il reprend les principes du langage UML, dont il est une extension, et présente ainsi un aspect multi vues intéressant pour l'obtention de modèles complets, pouvant par exemple prendre en compte des contraintes sur des valeurs et donc supporter l'idée de quantification. En effet, par rapport aux 4 axes de modélisation identifiés, les différents diagrammes, disposant de liens sémantiques entre eux, permettent de faciliter la modélisation des interactions entre les différents concepts.

Plusieurs travaux relatifs à ces modélisations ont en effet été déjà réalisés dans le cadre d'analyses quantitatives. A travers les seuls diagrammes de blocs (aspects statiques), paramétriques (aspects

interactionnels) modélisant les concepts de connaissance, des diagrammes d'états (aspects dynamique) peuvent être construits de manière cohérente (David et al. 2010).

De plus, d'après (Dobre, 2010), SysML est le langage au niveau d'abstraction le plus élevé et décorrélé de tout aspect métier. En outre, il est aujourd'hui le langage de référence pour la modélisation des systèmes (Branscomb et al. 2013). L'utilisation de ce langage est donc appuyée par la compatibilité avec d'autres utilisations du modèle, comme par exemple la simulation de systèmes continus régis par des équations différentielles (Modelica (C. J. J. Paredis et al. 2010)) ou pour le model checking.

Or, ce langage SysML souffre aujourd'hui d'un manque des méthodes pour la modélisation des systèmes : en effet, l'objectif premier du langage reste l'aide à la conception des systèmes à travers un langage commun aux différents domaines d'ingénierie (Estefan 2008). De plus, peu d'outils de validation des exigences initiales existent.

Le langage AADL, permet également la modélisation des systèmes à travers des vues statiques et dynamiques. Le nom d'AADL signifie *Architecture Analysis & Design Language (langage d'analyse et de conception d'architectures)*. En pratique AADL s'adresse à tous les systèmes embarqués temps-réel, et pas seulement aux systèmes avioniques. La description d'une architecture en AADL consiste donc en la description de ses composants et leur composition sous forme d'une arborescence. Cette description pouvant être contenue dans des fichiers, une base de données…

Cependant, ce langage reste dédié aux systèmes d'informations en temps réels, et est assez peu adapté aux systèmes physiques tels que ceux concernés par la CMPQ. En effet, (Rolland 2008) souligne la nécessité d'accompagner de tels langages d'un cadre permettant d'obtenir le comportement des systèmes décrits. En effet, ils ne disposent initialement pas d'une vue comportementale.

III.4.2 Les langages de modélisation comportementale

Parmi les différents langages de modélisation architecturale ci-dessus, les plus performants disposent d'une vue comportementale. La plus intéressante et répandue est celle des automates à états ou statemachine. Il s'agit d'un modèle état transition simple permettant notamment la simulation pas à pas, model-checking, ou l'exécution de modèle. C'est une vue à part entière intégrée dans les langages UML et SysML.

Cependant, parmi ces « statemachine », initialement appelées les « statemachine » de Harel, de légères différences existent principalement au niveau de leur exécution dans les langages et outils dédiés (Crane 2006) (IBM Rhapsody, Statemate, SCADE…), et donc leur sémantique n'est pas clairement définie (Reggio & R. J. Wieringa 1999). En outre, leur formalisme ne permet pas pour autant directement la simulation stochastique, et ce quel que soit l'outil support utilisé. Cependant, la possibilité de pouvoir faire de l'exécution de modèle à partir d'un statemachine est très intéressante dans notre contexte puisqu'elle permet de créer, en exécutant le diagramme, un ensemble de scénarios devant être en cohérence avec la connaissance modélisée, rendant ainsi possible une vérification complète du modèle. De même, d'un point de vue de la validation, les diagrammes présentent une vue graphique et sont donc facilement explicables à un expert à même de les valider.

III.4.3 Les langages décorrélés de tout formalisme de simulation stochastique : avantages et désavantages

Les principaux langages de haut niveau ou pour la simulation stochastique permettant de modéliser des aspects statiques et dynamiques ont été présentés. Il est à présent nécessaire de synthétiser leurs différents avantages/inconvénients dans le contexte de la CMPQ.

En conclusion, au niveau des langages de haut niveau, seuls les langages SysML et UML disposent, par l'intermédiaire des statecharts, d'une vue dynamique, pouvant être vérifiée par exécution de modèles. Les diagrammes paramétriques SysML ou la vue OCL d'UML peuvent modéliser des équations complexes, déterministes, stochastiques ou possibilistes, au sein du langage. De nombreux travaux (Bernardi et al. 2002) (Nottage & Corns 2011) ont traité du passage entre ces langages, par l'intermédiaire de profils, par la définition de règles algébriques ou par la définition d'algorithmes vers des langages plus formels. Cependant, SysML dispose d'une vue graphique (les diagrammes paramétriques) plus facilement validable qu'OCL par un expert. Ce point est en outre d'autant plus important que dans le contexte de la CMPQ, la connaissance générique présente des interactions complexes et de natures différentes. Ces statechart permettent en plus de fournir un format de sortie facilitant l'exploitation ultérieure de cette connaissance, en établissant des relations entre les éléments sémantiques des différents langages, dans le formalisme du modèle exécutable. En effet, dans le cas d'une exploitation transparente de la connaissance modélisée, il parait envisageable pour un utilisateur de l'outil supportant notre démarche de travailler uniquement sur ce modèle de haut niveau (le passage d'un modèle à un autre s'opérant de façon automatique).

Enfin, la démarche d'identification de la connaissance proposée au chapitre 2 est basée sur une vision conjointe SS et SP. Le langage SysML est utilisé dans le cadre d'études d'IS pour la construction des modèles de SS et de SP. De fait, l'utilisation de ce langage pour modéliser les concepts de connaissance relatifs à l'activité de maintenance notamment permettrait de spécifier plus simplement les interactions entre les modèles existants.

Le Tableau 18 reprend ces constats. Les notes ont été attribuées de la même manière que pour les tableaux précédents.

Tableau 18 – Langages décorrélés de tout formalisme de simulation stochastique et prérequis du modèle

Prérequis	Langage SysML	Langage UML	Langage AADL
possibilité de modéliser des transitions forcées	2	2	1
possibilité de modéliser des transitions conditionnées	2	2	1
possibilité de modéliser des transitions aux lois variables et complexes et ce dans plusieurs formalismes de calcul	2	1	1

La capacité à effectuer des simulations stochastiques	0	0	0
La capacité à réduire l'effort de modélisation pour chaque étude	2	2	2
La capacité d'être exploité par un utilisateur sans connaissance de ce langage	1	1	1
Aptitude à une exploitation de la connaissance modélisée par un autre langage	2	2	2
Capacité à être vérifié et validé	2	2	1

III.4.4 Conclusion sur le choix de l'approche pour la CMPQ : vers la proposition de deux modèles génériques

Les approches utilisant un langage par niveau d'abstraction paraissent donc les plus adaptées à notre problème pour trois raisons majeures :

- Le chapitre 2 ayant proposé une identification textuelle de la connaissance nécessaire à la CMPQ, il serait plus judicieux de chercher à modéliser, de manière exhaustive cette connaissance en restant cohérent sur les aspects métiers, ce qu'empêchent les langages dédiés à la simulation stochastique,
- Les modèles comportementaux créés sont vérifiables et validables,
- Une telle démarche permet de capitaliser la connaissance générique sous la forme de concepts de modélisation métier standardisés. A partir de ce modèle générique capitalisé, il est possible de créer des occurrences des concepts de modélisation métier, et donc de clairement réduire l'effort de modélisation pour chaque étude. Physiquement, cette procédure d'instanciation peut se matérialiser à travers, par exemple, une base de données relationnelle.

Ces trois raisons permettent de compléter l'étape (1') de construction du modèle générique amorcée par le chapitre 2. La résultante de cette étape est alors l'obtention de deux modèles génériques : l'un décorrélé de tout formalisme de simulation stochastique pour capitaliser la connaissance générique en évitant les pertes sémantiques par rapport à la connaissance brute ; et l'autre permettant la simulation stochastique, quitte à présenter quelques pertes sémantiques par rapport à la connaissance initiale. Il est cependant nécessaire d'aborder à présent la problématique de transformation de langages. L'étape (2) de création du modèle relatif à un système particulier se réalise directement par instanciation d'un des modèles génériques.

Il existe une dernière raison appuyant le choix de cette approche. En effet, l'un des besoins actuels de l'ingénierie basée sur les modèles (C.J.J. Paredis et al. 2010), est de disposer d'un modèle décorrélé de tout formalisme de simulation stochastique permettant d'englober la connaissance nécessaire à la fois à la simulation stochastique, mais aussi à d'autres applications connexes (cf. Figure 21) à l'évaluation des KPIs de la CMPQ, telles que par exemple la simulation continue des dégradations du système ou

la simulation du contrôle commande des systèmes. D'autant plus que ces applications peuvent être non seulement connexes mais constituer des données d'entrée à une étude de CMPQ (une simulation continue de l'évolution d'un mécanisme de dégradation peut entrainer la définition de seuils d'alarmes requis pour la CMPQ).

Le modèle générique du système reste le même, et seule la transformation de langage change (langage adapté à la simulation continue par exemple).

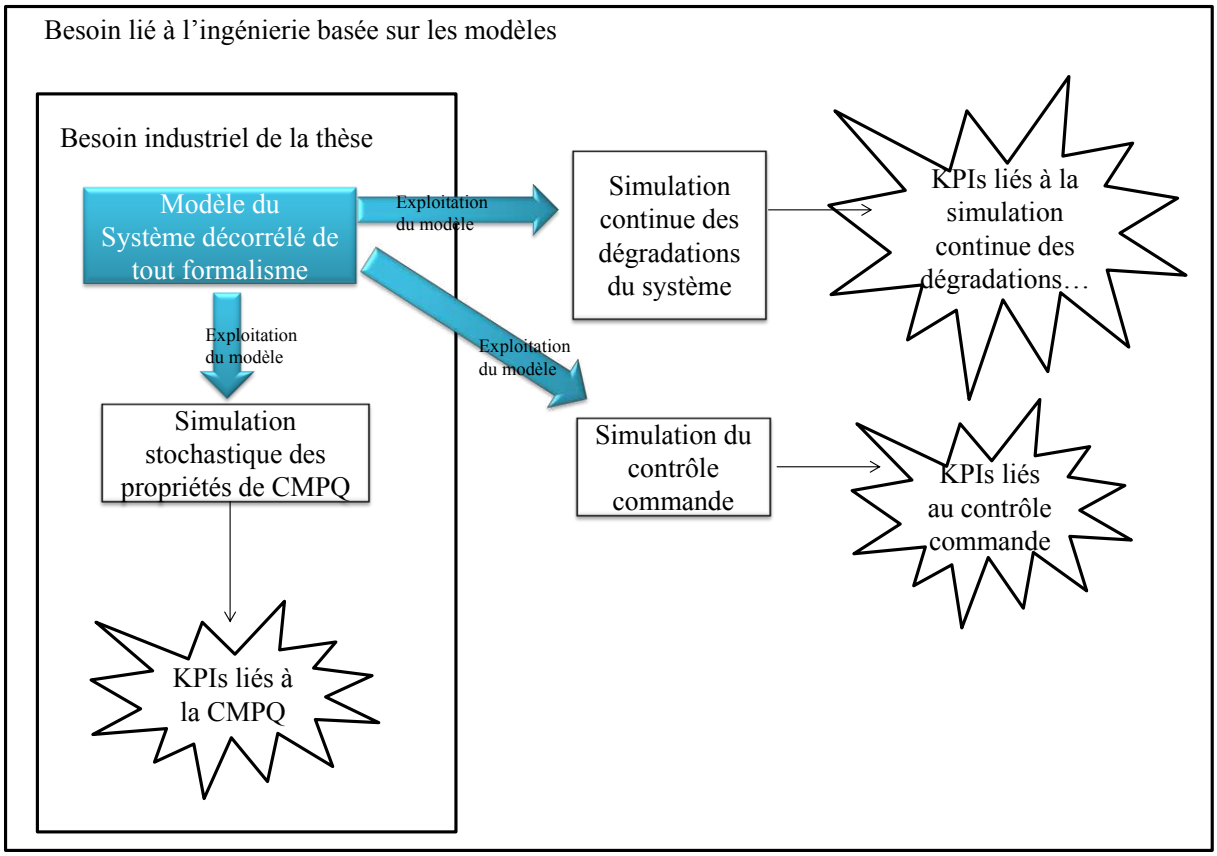


Figure 21 – Le besoin d'un modèle décorrélé de tout formalisme de simulation stochastique

III.5. Le besoin d'assurer une cohérence sémantique entre les modèles

Afin de décider du choix d'un langage par rapport à l'autre, il est également nécessaire de considérer le prérequis sous-jacent aux points précédemment abordés, à savoir garantir la cohérence dans la chaine de modélisation entre les langages, afin d'obtenir un modèle générique simulable cohérent avec le modèle générique décorrélé de tout formalisme de simulation stochastique. Pour ce faire, il faut :

- Identifier l'essence d'une transformation de modèle,
- évaluer l'aptitude à la transformation,
    - Au sein des vues du langage décorrélé de tout formalisme de simulation stochastique,
    - Entre le langage décorrélé de tout formalisme de simulation et le langage pour la simulation.

III.5.1 Contexte d'une ingénierie basée sur les modèles

Avant de rentrer dans la comparaison des langages d'intérêts du point de vue de leur aptitude à avoir une connaissance exploitée par un autre langage, il est nécessaire de s'intéresser aux fondements des transformations de modèles. En ingénierie basée sur les modèles (MDE) est un moyen automatique pour s'assurer de la cohérence d'une famille de modèles. Elle permet de réduire l'effort de modélisation et éviter les erreurs lors de la conception d'un modèle cible à partir d'un modèle source (Czarnecki, 2006), en établissant des relations entre les méta-modèles des deux langages. Ces transformations peuvent être endogènes ou exogènes, selon que le modèle est exprimé ou non dans le même langage. Enfin, ces transformations sont implémentées physiquement à l'aide de langages dédiés (XML...).

Dans notre choix de deux langages différents, la transformation de langage est exogène. Cependant, étant donné l'envergure d'un travail de cette nature (Mens & Van Gorp, 2006), notre travail de thèse va surtout chercher à exploiter la connaissance issue d'un modèle décorrélé de tout formalisme de simulation stochastique pour créer le modèle permettant à la simulation stochastique, plutôt que de chercher à supporter tous les mécanismes d'une transformation de langage en cohérence avec ses fondements scientifiques. Cette exploitation se matérialise par un mapping unidirectionnel depuis les différents éléments sémantiques d'intérêt du langage source vers les éléments sémantiques d'intérêt du langage cible. Ainsi, les aspects de transformation exhaustive d'un langage vers un autre à partir de leurs méta-modèles respectifs, ainsi que les aspects liés au langage dédiés aux transformations de modèles ne seront pas traités, et doivent faire l'objet de perspectives à ce travail.

III.5.2 Au sein des langages décorrélés de tout formalisme de simulation stochastique

Le langage SysML, étant donné les travaux actuels liés à l'IS et le besoin d'avoir un langage capable de supporter les différentes phases de conception d'un système, a été l'objet de plusieurs travaux fournissant des méthodes permettant d'assurer la cohérence entre les diagrammes statiques et dynamiques. Le langage UML permet également d'aborder ce point.

Par exemple, le logiciel Rational Rhapsody (*www.ibm.fr*) supportant ces langages permet la formation de diagrammes à partir d'éléments sémantiques d'autres diagrammes. Cela permet d'une part une d'assurer la cohérence sémantique dans le processus, interne au langage, de modélisation de la connaissance, et d'autre part un support considérable à la vérification des modèles.

III.5.3 Depuis les langages décorrélés de tout formalisme de simulation stochastique vers les langages pour la simulation stochastique

La problématique du passage depuis le modèle décorrélé de tout formalisme de simulation stochastique vers le langage pour la simulation stochastique choisi se réfère plus à la communauté de l'interopérabilité et de la transformation de langages. Cependant, dans le cas des travaux liés à la maintenance, les auteurs s'attâchent principalement à définir des règles de correspondances entre les éléments sémantiques d'intérêt des deux langages plutôt qu'à une transformation de langage exhaustive. (Bernardi et al. 2002) (Monnin et al. 2011) par exemple se sont penchés sur ces approches, notamment à partir des vues statiques du langage UML. En effet, ces modèles, tous comme les

modèles SysML, peuvent être « profilés », construits pour être directement compatibles avec les langages cibles. Il est en effet essentiel pour cette étape de partir d'un langage de décorrélés de tout formalisme de simulation stochastique vers un langage plus applicatif. Une autre approche consiste à faire les liens sémantiques entre les différents langages grâce à des définitions algébriques. Ces définitions peuvent ensuite être exploitées pour l'implémentation de passerelles XML entre des outils supports utilisant les langages en question.

Le Tableau 19 reprend certains de ces travaux, établissant des correspondances entre un langage de ce type et un langage dédié à la SdF.

Tableau 19 – Correspondances de langages vers la simulation stochastique

Auteurs	Langage décorrélé de tout formalisme de simulation stochastique	Langage pour la simulation stochastique
(David et al. 2010)	SysML/ADF	Pas de simulation stochastique
(Nikolaidou et al., 2008)	SysML (profilé)	DEVS
(Bernardi et al., 2008)	UML (profilé)	RdP
(Monnin et al. 2011)	UML (diagramme statique)	SAN
(Dumas et al., 2012)	AADL	ADF

La plupart de ces travaux utilisent uniquement un modèle statique de la connaissance pour créer les modèles comportementaux simulables, laissant ainsi la place à des erreurs liés au modélisateur.

Certains travaux se basant sur SysML comme par exemple (David, 2009) dans le contexte de MEDISIS, vont plus loin puisqu'ils proposent une création de modèles dynamiques à partir non seulement de modèles statiques mais aussi d'une vue comportementale SysML. Cependant, seules des interactions simples sont considérées, et le langage ADF n'est pas utilisé pour la simulation stochastique. Des travaux depuis SysML vers ADF ou les STG sont en plein essor, et disponibles sur *www.lix.polytechnique.fr*.

III.6. Bilan de l'état de l'art

Ainsi, en accord avec l'état de l'art présenté ci-dessus, nous proposons dans cette thèse pour répondre aux besoins de la CMPQ l'utilisation dans un premier temps du langage SysML, par lequel les vues statiques et comportementales vont permettre de représenter et capitaliser l'ensemble de la connaissance générique nécessaire pour compléter l'étape (1') de la démarche sans se borner aux applications liées à la simulation stochastique. Par rapport aux langages concurrents, il permet en effet de modéliser l'ensemble de la connaissance identifiée au chapitre 2 sans pertes sémantiques. Ainsi, l'ensemble du modèle fourni pourra être réutilisé à travers différents domaines d'ingénierie et servir pour quantifier les prérequis de SdF sur un système particulier dans le cas de la conception des

systèmes. En effet, ce langage est le langage de base de l'IS, et il parait intéressant dans ce contexte de fournir un modèle des interactions génériques entre SS et SP dans ce langage commun.

Le modèle simulable permettant de réaliser l'étape (3) sera formalisé ensuite dans un langage dédié à la sûreté de fonctionnement, puisque ceux-ci présentent de nets avantages en termes d'utilisabilité en regard des langages tels que les réseaux de Pétri. Parmi ces langages, nous choisissons le langage ADF puisque celui-ci dispose d'algorithmes d'évaluations performants et présents dans des environnements disponibles tels que SIMFIA, permettant sa compilation transparente en modèles simulables. En outre, il présente une visibilité nationale importante, tant au niveau industriel (partenariat avec Dassault, Cassidian…) que scientifique (une douzaine de thèses ont utilisé ce langage tant à un niveau fondamental (Point, 1999) (Kehren 2005), qu'à un niveau industriel (Adeline 2011) ou technique (David et al. 2010), (Cressent et al. 2013)). De plus, de par la philosophie de construction du langage (Bouissou et al., 2006), les modèles ADF sont faciles à obtenir depuis les modèles d'états SysML notamment grâce à la possibilité de synchronisation des évènements, constituant ainsi un véritable avantage en support du mapping demandé.

Cette démarche doit cependant être développée de manière à répondre au problème de cohérence sémantique entre les différents langages et vues. En effet, l'obtention des modèles à états SysML doit être guidée par une déclinaison des connaissances statiques issues des normes et standards et capitalisée statiquement grâce aux diagrammes statiques. Ce point peut impliquer l'utilisation de diagrammes intermédiaires dans la modélisation.

De même, les aspects liés à la simulation stochastique et aux limites du langage ADF doivent être, lors du passage à ce langage, identifiés et modélisés dans le langage pivot (SysML) afin de pouvoir réaliser le mapping voulu entre les deux langages. En effet, ADF ne permet pas de modéliser les systèmes bouclés, et a été conçu pour la définition de flux uniquement fonctionnels. Ainsi, les transitions à paramètres variables ne peuvent être modélisées. De même, seules des équations booléennes peuvent être modélisées. Nous prendrons en compte ces limites au chapitre 5, traitant du mapping depuis le modèle SysML vers le modèle ADF, de la façon suivante :

- En s'adaptant : définition des systèmes bouclés, définition d'un cadre booléen. Une perte sémantique est alors entraînée par rapport à la connaissance d'intérêt, mais celle-ci est jugée non critique par rapport aux besoins d'EDF ;
- En les contournant : définition de flux d'informations non seulement fonctionnels mais aussi « informationnels », éclatement des transitions.

III.7. Vérification et validation potentielles des modèles issus de la démarche proposée

Les différents modèles issus de la démarche proposée doivent être validés et vérifiés. En effet, il est logique à ce stade d'évaluer si le modèle créé est bien celui attendu (le bon modèle) pour supporter la CMPQ.

Dans notre démarche proposée, la démarche de vérification et de validation doit intervenir à plusieurs niveaux. Il s'agirait tout d'abord de vérifier le modèle SysML par rapport à la connaissance initiale métier. Pour répondre à ce point, une possibilité est d'effectuer une démarche de rétro ingénierie depuis le modèle SysML pour reconstruire la connaissance métier, puis de comparer les

résultats obtenus avec la connaissance brute initiale. Ceci peut être facilité par les liens sémantiques entre les différents diagrammes, et par un outillage adapté (par exemple, l'outil Rhapsody permet de supporter certaines correspondances entre diagrammes). Concernant la validation de ce modèle, elle peut se faire par entretien avec un expert du domaine capable de crédibiliser la connaissance modélisée (dans notre cas, un expert EDF en CMPQ).

De la même manière, il faut vérifier et valider le modèle ADF pour la CMPQ. En termes de vérification, une possibilité est d'effectuer une vérification formelle par model checking partielle du modèle ADF. Pour valider ce modèle, il est nécessaire de passer par des phases d'exploitation du modèle et d'utilisation des résultats par un client (EDF dans notre cas) pour en dégager un intérêt, et donc de comparer les résultats de simulation obtenus avec le modèle ADF avec d'autres langages/méthodes (benchmark). Une autre solution consiste à poser certaines hypothèses (markovienne par exemple) permettant de créer des modèles analytiques équivalents, et donc d'évaluer partiellement si le modèle donne des résultats proches ou non des résultats théoriques.

Ces aspects de vérification et de validation seront concrètement abordés dans les chapitres 4, 5 et 6. Le Tableau 20 synthétise ces points.

Tableau 20 – Vérification et validation de la démarche

Etapes de la démarche	Vérification	Validation
Modèle SysML pour la CMPQ	Démarche de rétro ingénierie vers la connaissance métier	Expert métier
Modèle ADF pour la CMPQ	Model checking	Utilisation de la démarche par le client Vérification analytique par comparaison entre des résultats théoriques et les résultats donnés par le modèle

III.8. Conclusion

A partir des prérequis sur l'utilisabilité d'un modèle et sur la puissance de modélisation, nous avons présenté dans ce chapitre notre raisonnement pour le choix de deux langages de modélisation pour permettre de répondre le plus précisément aux besoins des étapes (1'), (2) et (3) de la démarche de modélisation choisie. Ce raisonnement conduit au choix à la fois du langage SysML et du langage ADF pour la modélisation générique du problème de la CMPQ et donc supporter les étapes (1'). L'étape (2) d'instanciation du modèle générique doit être réalisée avec ces deux modèles, en sachant que la réalisation de l'étape (3) de simulation ne peut être supportée que par le langage formel ADF. Cependant, la construction du modèle ADF d'un système particulier est guidée par le modèle de la connaissance décorrélé de tout formalisme de simulation stochastique.

La Figure 22 projette la Figure 1 décrivant la méthode proposée pour la quantification des programmes de maintenance dans le référentiel des langages choisis.

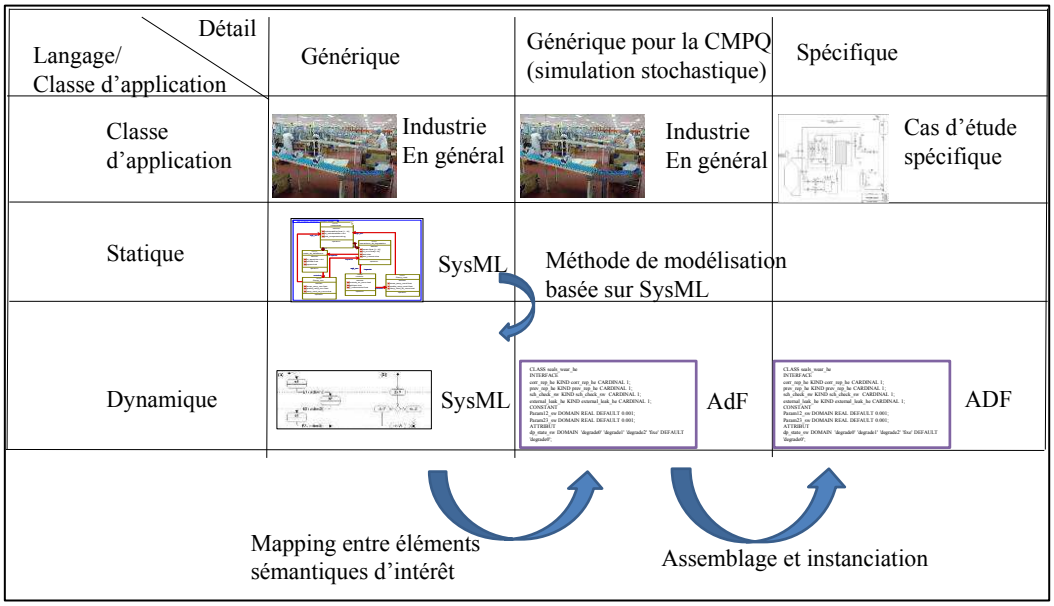


Figure 22 – Méthodologie proposée

Les originalités de cette proposition sont donc les suivantes :

- L'utilisation d'un langage semi formel pour à la fois:
  - capitaliser la connaissance statique et comportementale de manière cohérente avec les différentes normes et standards,
  - assurer une meilleure correspondance sémantique entre les aspects statiques et comportementaux,
  - permettre une validation et une vérification du modèle.
- La formalisation de règles de correspondance entre deux formalismes états transition : les statemachine SysML (dédiés à la simulation pas à pas) et les automates de mode (dédiés à la simulation stochastique sous ADF),
- La proposition de deux modèles génériques chacun à leur niveau pour respectivement permettre une réutilisation des modèles : pour la simulation des systèmes en général et pour la simulation stochastique des systèmes.

L'objectif est donc à présent de créer un modèle SysML4CMPQ conforme à la connaissance identifiée dans le chapitre 2 en utilisant les liens sémantiques entre les vues du langage. Le chapitre 4 va détailler cette démarche.

Chapitre IV. Méthodologie de construction du modèle SysML4CMPQ

IV.1. Introduction

Le chapitre 2 a permis de définir l'ensemble des concepts de modélisation nécessaire à la proposition d'un modèle générique de la connaissance support à l'étape (1') de la démarche. Pour capitaliser cette connaissance en évitant les pertes sémantiques, le chapitre 3 a conclu sur le choix du langage SysML.

Or, dans ce même chapitre, des outils ont été identifiés comme permettant d'exploiter de manière cohérente l'utilisation successive de ses différentes vues et diagrammes, et aussi permettant de faire de la vérification de modèle. L'objectif de ce chapitre est donc, en appliquant cette démarche au problème de la CMPQ, de construire un modèle SysML constitué de différents diagrammes permettant de modéliser la connaissance générique métier du chapitre 2 en se servant des éléments sémantiques SysML pour garantir la cohérence entre les différents diagrammes utilisés. Ce modèle est par la suite noté modèle SysML4CMPQ, par concaténation des termes SysML, « for » et CMPQ.

Pour ce faire, nous allons dans un premier temps introduire le langage SysML de manière générale afin d'en identifier les diagrammes intéressants dans notre contexte. Nous décrirons ensuite les éléments sémantiques spécifiques à ces diagrammes permettant de réaliser la démarche de construction du modèle SysML4CMPQ. Ensuite, nous décrirons ces étapes en établissant des correspondances entre les éléments sémantiques SysML et les éléments de connaissance identifiés (transitions forcées ou simultanées, états, transitions simples…) au chapitre 2. Chaque étape sera illustrée par un exemple. Ce chapitre conduit à l'obtention du modèle SysML4CMPQ, et les diagrammes sont réalisés à l'aide du logiciel Rational Rhapsody permettant de guider l'établissement des correspondances sémantiques entre les différents diagrammes. SysML est donc ici utilisé comme langage pour modéliser les connaissances métier de la CMPQ, en cohérence avec les pratiques de l'IS mais sans contribution à ce langage, ni à l'IS proprement dite. Chaque élément sémantique SysML sera par la suite noté en italique.

IV.2. Présentation du langage SysML

SysML s'articule autour de neuf types de diagrammes, chacun d'eux étant dédié à la représentation des vues particulières d'un système. Ces diagrammes sont répartis par l'OMG en trois groupes (cf. Figure 23): le diagramme transverse, les diagrammes structurels et les diagrammes comportementaux.

Le diagramme transverse est :

- le diagramme d'exigence (montre les exigences du système et leurs relations).

Les diagrammes structurels correspondent à :

- diagramme de packages (montre l'organisation logique du modèle et les relations entre packages).
- diagramme de définition de blocs (montre les briques de base statiques : blocs, compositions, associations, attributs, opérations, généralisations, etc.),
- diagramme de bloc interne (montre l'organisation interne d'un élément statique complexe),
- diagramme paramétrique (représente les contraintes du système, les équations qui le régissent),

Les diagrammes comportementaux correspondent à :

- diagramme de cas d'utilisation (montre les interactions fonctionnelles entre les acteurs et le système à l'étude).
- diagramme d'activité (montre l'enchainement des actions et décisions au sein d'une activité complexe),
- diagramme de séquence (montre la séquence verticale des messages passés entre blocs au sein d'une interaction),
- diagramme d'états (montre les différents états et transitions possibles des blocs dynamiques),

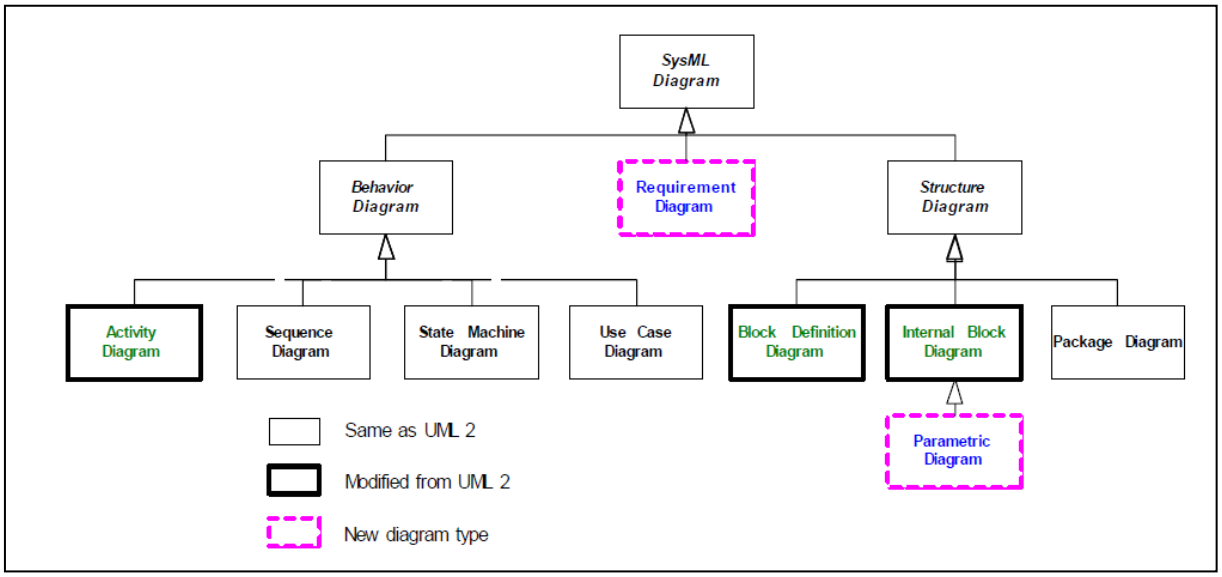


Figure 23 – Taxonomie des diagrammes SysML (OMG, 2010)

Pour décrire l'ensemble des concepts identifiés dans le chapitre 2, les diagrammes de définition de blocs, les diagrammes paramétriques et les diagrammes d'états sont suffisants (David 2009). Cependant, d'autres diagrammes peuvent venir faciliter la construction des diagrammes d'états par rapport à la connaissance. C'est par exemple le cas des diagrammes de séquences (SD) (Hoffmann 2011), qui viendront former une étape intermédiaire dans la création de ces diagrammes et permettront une validation éventuelle des diagrammes d'états sous l'outil Rational Rhapsody. A présent, nous allons étudier, en reprenant la démarche d'identification de la connaissance Figure 8, les équivalences et les relations entre les éléments SysML des différents diagrammes d'intérêt et les étapes de la démarche d'identification.

IV.3. Description de la démarche de construction du modèle SysML4CMPQ

IV.3.1 Modélisation indépendante des différents concepts de connaissance

La première étape de la démarche d'identification de la connaissance consiste à identifier la connaissance relative aux différents concepts de manière physiquement indépendante. Par exemple, sans l'intervention de facteurs extérieurs liés à d'autres concepts (politique de maintenance), un mode de défaillance ne peut pas disparaitre. Cependant, ce même mode de défaillance peut se voir attribuer

une probabilité propre d'occurrence. Cette étape fait donc apparaitre des caractéristiques comportementales et statiques partielles de chaque concept, en attendant de les compléter avec la prise en compte de leurs interactions.

Les *blocs* SysML sont particulièrement adaptés à représenter ces concepts. D'après (David et al. 2010), « le bloc est la brique de base dans la déclaration des concepts constitutifs d'un système ». Ils sont utilisés pour représenter des entités concrètes (système, composant) comme abstraites. L'assemblage de ces différents *blocs* forme un diagramme de définition de blocs (BDD), et chaque *bloc* le constituant peut être instancié (par exemple, un mode de défaillance est un *bloc* générique, et la défaillance à la sollicitation une instance de ce *bloc*). Les éléments SysML constitutifs des blocs sont les suivants :

- Les *properties* : elles définissent les caractéristiques structurelles des blocs. Elles peuvent être de plusieurs types :
    - Les *value properties* : elles décrivent les caractéristiques quantifiables des blocs,
    - Les *parts* : elles sont issues d'un lien de compostion entre blocs,
    - Les *port properties* : elles décrivent les caractéristiques structurelles indiquant les points d'interaction offerts par le bloc. Ils peuvent être de deux types : les *flow ports*, modélisant les flux transformés, et les *standard ports*, modélisant les interfaces de demande ou de proposition de service,
    - Les *constraint properties* expriment les contraintes vérifiées par *les value properties* du bloc,
- Les *opérations* : elles définissent les traitements réalisables par les instances des blocs.

Ces différents éléments sémantiques permettent de représenter l'ensemble des caractéristiques et opérations internes propres à chaque concept identifié dans le chapitre 2. Cependant, dans ce chapitre 2 des comportements propres aux différents concepts sont identifiés. Ces visions état transition peuvent être modélisées par l'intermédiaire des statemachine (STM) diagrams ou diagrammes d'états. Ces diagrammes seront toutefois partiels à ce niveau (non considération des interactions entre les diagrammes), et leur sémantique ne sera présentée qu'après avoir défini celle des diagrammes de séquences permettant d'obtenir les STM complets.

IV.3.2 Modélisation des interactions structurelles entre concepts de connaissance

La démarche d'identification de la connaissance propose ensuite d'établir les différentes relations structurelles entre les concepts. Ces différentes relations permettent de construire le Block Definition Diagram (BDD) partiel en associant les différents *blocs* précédemment créés. Le BDD SysML propose plusieurs types de relations permettant de maitriser la complexité du modèle :

- Les *associations* : elles expriment de façon générale une relation entre blocs. Elles nécessitent la définition de *cardinalités* entre les *blocs* en interaction,
- Les *généralisations* : elles expriment une relation d'héritage entre des *blocs*. Ce type de relation permet notamment de gérer la complexité du BDD. Par exemple, le *bloc* politique de

maintenance corrective est un type de politique de maintenance. Il aura donc les mêmes associations et propriétés que le bloc politique de maintenance,

- Les *compositions* indiquant la déclaration d'un *part*.

Le BDD ainsi formé est complet au niveau des interactions structurelles. Cependant, les *blocs* ne sont pas encore complétés un à un. A ce niveau, le BDD n'est donc que partiel.

IV.3.3 Modélisation des interactions dynamiques et paramétriques entre concepts de connaissance

La troisième étape d'identification de la connaissance précise les relations structurelles précédentes selon leur nature paramétrique ou dynamique.

Concernant les interactions paramétriques, les diagrammes paramétriques (PD) sont particulièrement adaptés à ce type de représentation. Ils se composent de :

- *Blocs* : ces blocs sont équivalents aux blocs précédemment définis,
- *Constraintparameter* : ces éléments sont rattachés aux blocs, et permettent de définir les grandeurs rentrant en jeu dans une contrainte paramétrique,
- *Constraintblock* : ces blocs permettent de spécifier la nature de la relation entre les différents *constraintparameter.*

Concernant les interactions dynamiques, si elles sont directement représentables par les STM (par l'intermédiaire notamment des messages envoyés), il est plus judicieux d'utiliser les digrammes de séquence (SD) pour les modéliser dans un premier temps.

En effet, les SD sont particulièrement adaptés pour représenter l'ensemble de ces interactions de manière méthodique et intuitive. Les SD se composent d'un certain nombre d'éléments sémantiques :

- Les *lifelines :* ce sont les éléments de base de ces diagrammes, ils modélisent des instances de blocs équivalentes aux blocs définis précédemment,
- Les *messages* : ils peuvent être synchrones ou asynchrones. Ils peuvent également être réflexifs, c'est-à-dire bouclés sur une même *lifeline*,
- Les opérateurs : ils permettent de représenter des boucles temporelles, des chemins alternatifs, ou encore des chemins optionnels.

Les messages, lorsqu'ils sont fonction d'une grandeur, à savoir un attribut du bloc duquel la *lifeline* est une instance, sont suivis d'une parenthèse dans laquelle le ou les paramètres sont notés. Ces diagrammes peuvent être enrichis par des opérateurs particuliers pouvant représenter des boucles temporelles, des chemins alternatifs, ou encore des chemins optionnels idéaux pour modéliser des transitions conditionnées. Il existe d'autres opérateurs ou éléments dans les SD, mais ils sont superflus dans le cadre de la formalisation du modèle pour la CMPQ (OMG, 2010).

IV.3.4 Modélisation des comportements globaux des concepts de connaissance

A chaque *bloc* peut être associé un STM. Ils sont utilisés pour modéliser comment les *blocs* se comportent face à l'arrivée d'événements. Le formalisme utilisé est basé sur les statecharts de David Harel (Harel 1987). Ils se composent de plusieurs éléments sémantiques, parmi lesquels :

- Les *états* : ils permettent de définir les différents états des blocs,
- Les *transitions* : elles permettent de relier les états entre eux,
- Les *messages envoyés* : ils permettent de spécifier les transitions forcées sur d'autres blocs liées à l'occurrence d'une transition,
- Les *états d'attente* : ils permettent de spécifier les états accessibles au moment de l'occurrence d'une transition forcée. Pour ce faire, il est nécessaire de définir des macros-états, englobant notamment les états d'attente.

Chaque opération interne étant liée à un attribut de bloc dans notre modélisation, ces diagrammes seront annotés de sorte à faire apparaitre clairement le ou les paramètres déclencheurs de chaque *transition.*

En replaçant ces diagrammes d'états dans les étapes de notre démarche d'identification de la connaissance, ceux-ci peuvent être utilisés dans la première étape pour modéliser le comportement partiel des concepts. Cependant, ils s'avèrent particulièrement intéressant lors de la modélisation des comportements globaux des concepts. En effet, des liens entre les éléments sémantiques des concepts des STM et les éléments sémantiques des SD existent, ce qui permet de guider la construction des STM globaux à travers l'outil Rhapsody.

Un ensemble de diagrammes SysML est maintenant disponible. Cependant, les STM et BDD initiaux ne sont que partiels puisqu'ils ont été formés de manière analogue à l'identification de la connaissance. Il est cependant nécessaire à ce stade, pour compléter la vue comportementale du modèle donnée par les STM, de créer les transitions simultanées créées par les interactions entre les concepts de connaissance. Pour ce faire, il est nécessaire de prendre en compte l'information des SD précédemment définie. Une méthode d'agrégation des STM basée sur Harmony (Hoffmann 2011) est donc proposée, pour permettre d'assurer la cohérence entre les différents diagrammes précédents. Ainsi, les éléments liés notamment aux différents messages envoyés sont créés, et les STM et BDD sont complétés.

Cette section a présenté la méthodologie de construction du modèle SysML4CMPQ à partir de la démarche d'identification de la connaissance. Nous allons à présent appliquer cette méthodologie en prenant soin de faire le lien entre les éléments de connaissance textuelle du chapitre 2 et les éléments sémantiques du langage SysML définis précédemment, afin de faciliter l'application de la démarche à un autre langage que SysML le cas échéant. La démarche est synthétisée et mise en relation avec la démarche d'identification de la connaissance générique Figure 24.

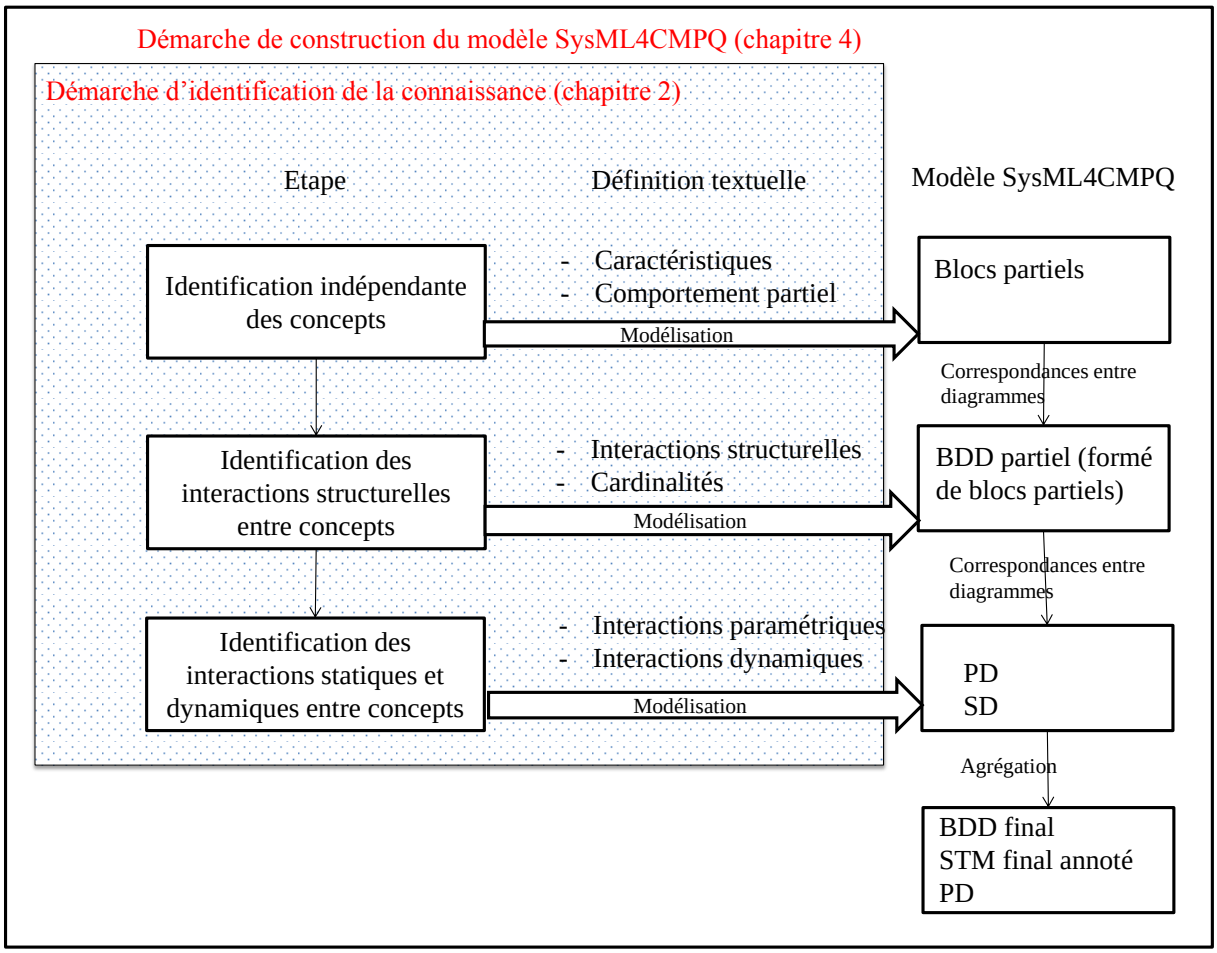


Figure 24 – Démarche de construction du modèle SysML4CMPQ

IV.4. Modélisation indépendante des concepts de connaissance

IV.4.1 Création des blocs partiels

La première étape de la démarche Figure 24 propose d'utiliser les *blocs* SysML pour modéliser les concepts de connaissance de manière indépendante, tels qu'ils ont été identifiés au chapitre 2. Ces *blocs* ne sont à ce niveau que partiels puisque la CMPQ requiert également la définition des *opérations* du *bloc* et la définition des *constraint properties* du bloc. Ces *blocs* seront ainsi complétés au fur et à mesure de la démarche de modélisation.

Ce sont ces objets qui plus tard seront manipulés par l'utilisateur de l'outil, pour la réalisation des études de quantification de programmes de maintenance complexes. Chaque concept dispose de caractéristiques (cf. Tableau 2 à Tableau 1). Ces caractéristiques sont formalisables sous la forme de *value properties* SysML propres à chaque concept. Ils sont attachés à une multiplicité et à un type de variable (chaine de caractère, réel…) ou peuvent répondre à une énumération induisant un choix multiple. Par défaut, les multiplicités des *value type* sont de 1. Sous Rhapsody, les *value type* relatifs à des chaines de caractères sont notées « char », les *value type* relatifs à des lois d'évolution sont notées « void », et les autres seront modélisés par des réels, notés « float ». Concernant les énumérations, elles reprennent le nom de l'énumération définie, comme par exemple dans le cas de la définition d'une technologie particulière pouvant être : électrique, mécanique…

Chaque *opération* est pour l'instant équivalente à une transition interne identifiée au chapitre 2.

Le Tableau 21 reprend les correspondances entre la connaissance identifiée et les éléments sémantiques des blocs SysML équivalents.

Tableau 21 – Correspondances entre les éléments de connaissance et les éléments sémantiques des blocs SysML

Connaissance	Elément SysML équivalent
Concept	Bloc
Caractéristique	Value type
Transition interne	Opération

Les Figure 25 (a) et (b) montrent les diagrammes de blocs partiels relatifs à un mécanisme de dégradation et à une politique de maintenance corrective. Tout au long de leur modélisation, ces exemples sont représentatifs des différentes interactions d'intérêt. Elles sont cohérentes avec les caractéristiques et opérations des concepts équivalents repris respectivement dans le Tableau 2 et le Tableau 1.

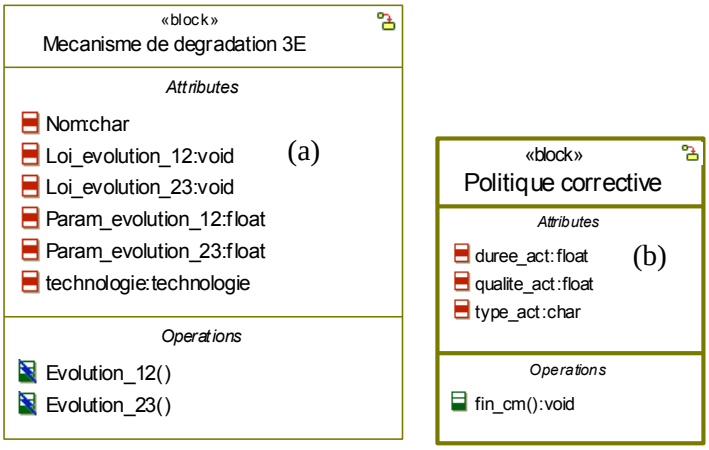


Figure 25 – (a) Bloc partiel d'un mécanisme de dégradation à 3 états. (b) Bloc partiel d'une politique corrective

A ce niveau d'identification de la connaissance dans la démarche de construction du modèle SysML4CMPQ, des diagrammes d'états partiels peuvent également être construits, de manière cohérente avec les blocs ci-dessus.

IV.4.2 Création du diagramme d'état partiel associé à chaque bloc partiel

Les différents états et transitions internes représentés par les schémas état-transitions du chapitre 2 sont strictement équivalents aux *états* et *transitions* des STM SysML. En plus, il est nécessaire d'identifier un *état initial* dans les STM. Ces représentations états-transitions sont cependant pourvues

des paramètres déclencheurs de la transition modélisant le temps passé dans l'état la précédant, notés entre parenthèses. Or, dans les STM SysML, il y a une signification différente à une telle représentation. Nous annoterons donc les transitions des STM concernées avec un lien et un encadré reprenant le ou les paramètres déclencheurs. Le Tableau 22 reprend les correspondances entre la connaissance identifiée et les éléments sémantiques des STM SysML équivalents.

Tableau 22 - Correspondances entre les éléments de connaissance et les éléments sémantiques des STM SysML

Connaissance	Elément SysML équivalent
Transition interne	Transition
Etat	Etat
Paramètre déclencheurs	Annotation sémantique : encadré jaune

Par exemple, dans le cas d'un mécanisme de dégradation à 3 états, seules les transitions liées à la dégradation physique de celui-ci lui sont internes (cf. représentation Figure 15). Le STM relatif au *bloc* partiel de la Figure 25 (a) est noté Figure 26. Les *transitions* ainsi créées sont équivalentes aux *opérations* du *bloc* partiel associé. Pour rester cohérent avec les termes définis chapitre 2, la notation « deg_1 » est utilisée pour nommer l'état nominal de dégradation.

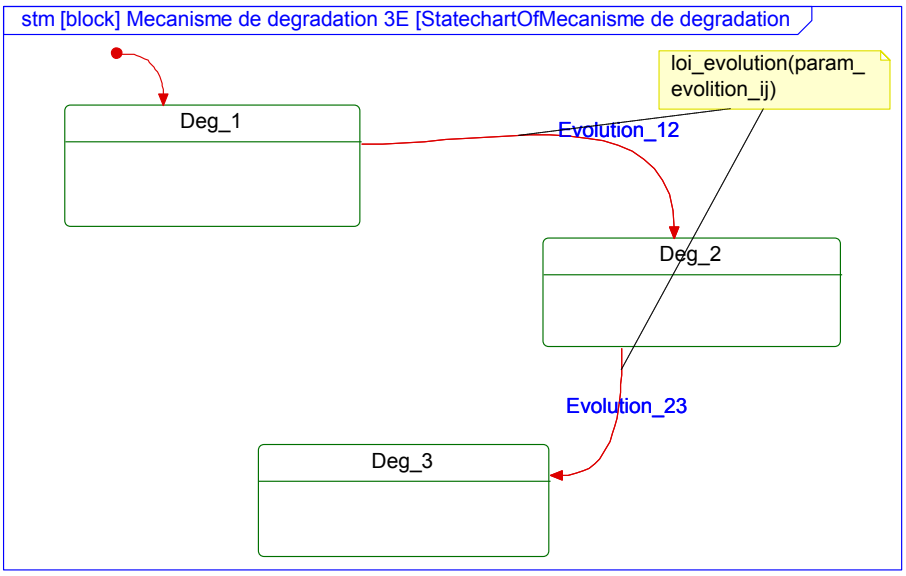


Figure 26 – STM partiel d'un mécanisme de dégradation à 3 états

Dans le cas de la politique de maintenance corrective, en reprenant la représentation états-transitions Figure 9, seule l'opération « fin_AM », dépendant de la durée de l'action de maintenance, est interne. Son diagramme d'états partiel est modélisé Figure 27.

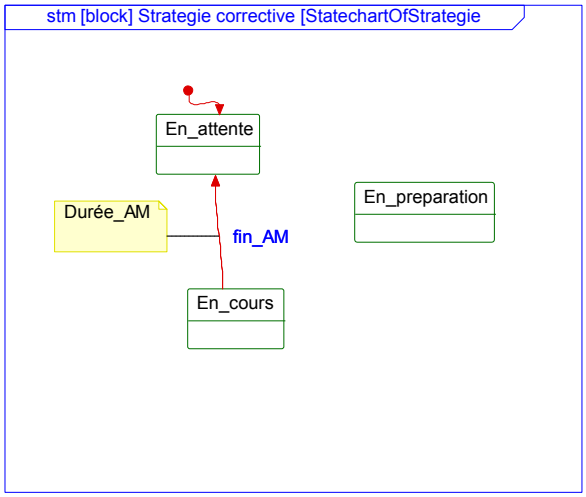


Figure 27 – STM partiel d'une politique corrective

En appliquant ces principes de modélisation à l'ensemble des concepts identifié au chapitre 2, l'ensemble des blocs et des STM partiels peut être construit.

IV.5. Modélisation des interactions structurelles entre les concepts de connaissance

Il est à présent nécessaire d'associer les différents blocs partiels précédemment obtenus. Les interactions structurelles du modèle métier pour la CMPQ ont été identifiées au chapitre 2 (du Tableau 6 au Tableau 9). Chaque relation est équivalente à une association SysML, exceptées celles possédant une relation de cardinalité 0,1 ou 1pour lesquelles l'élément SysML équivalent est la composition, et les relations hiérarchiques (relations de type « peut être »), pour lesquelles l'élément SysML équivalent est la généralisation. Celles-ci permettent de disposer d'un modèle lisible à condition d'employer une méthode adaptée, partant du plus haut niveau hiérarchique vers le plus bas. En effet les attributs et associations de chaque bloc « de haut niveau » sont systématiquement reprises pour ses blocs particuliers.

Ces relations hiérarchiques sont dans notre modèle liées aux concepts :

- Des politiques de maintenance : toutes les politiques de maintenance possèdent des attributs en commun. En rentrant dans le détail de ces politiques, certaines agissent sur différents concepts du SP notamment,
- Des concepts au comportement non générique : un mécanisme de dégradation peut être à deux ou trois niveaux de dégradations par exemple.

Le Tableau 23 reprend les correspondances entre la connaissance identifiée et les éléments sémantiques des BDD SysML équivalents.

Tableau 23 - Correspondances entre les éléments de connaissance et les éléments sémantiques des BDD SysML

Connaissance	Elément SysML équivalent
Relation hiérarchique	Généralisation
Relation de cardinalité 0,1 ou 1	Composition vers le bloc de cardinalité 0 ou 1
Autre et cardinalités	Association et cardinalités

Toutefois, la construction d'un BDD doit être soumise à une démarche descendante pour sa formalisation propre. Etant donné la connaissance identifiée et le fait d'avoir modélisé le sous-système lié à l'activité de maintenance au centre du SS et du SP, il convient donc de partir du niveau des programmes de maintenance (le plus haut niveau conceptuel du sous-système de l'activité de maintenance) pour identifier les relations génériques entre les blocs de haut niveau, puis de les préciser en fonction de la descente dans la hiérarchie de la connaissance. Il faut procéder de la même manière pour l'identification des attributs des différents blocs. En effet, les généralisations SysML permettent de modéliser les attributs et relations génériques aux blocs généralisés.

La Figure 28 reprend une partie de ce BDD relative aux interactions internes à l'axe de l'activité de maintenance, sous la forme d'une taxinomie, mettant ainsi en valeur la « factorisation » des attributs des blocs, en utilisant les notions de généralisation. Les blocs créés généralisés n'ont cependant pas pour but d'être manipulés pour la CMPQ. Ils servent simplement à améliorer la lisibilité du modèle.

En procédant de la même manière pour l'ensemble des interactions (tout d'abord à partir du programme de maintenance), alors le BDD des blocs partiels peut être entièrement construit sous Rhapsody. Le BDD Figure 29 modélise les relations entre l'activité de maintenance et les autres axes dans le cas d'une politique de maintenance basé sur des observations. Ces associations sont modélisées de manière conforme avec les connaissances identifiées au chapitre 2. Ce BDD se lit de « gauche à droite », et on y retrouve tant les concepts de connaissance du SS que ceux du SP.

Cependant, les BDD ne permettent pas de préciser les différentes relations dynamiques et paramétriques (définies au chapitre 2) entre les différents blocs.

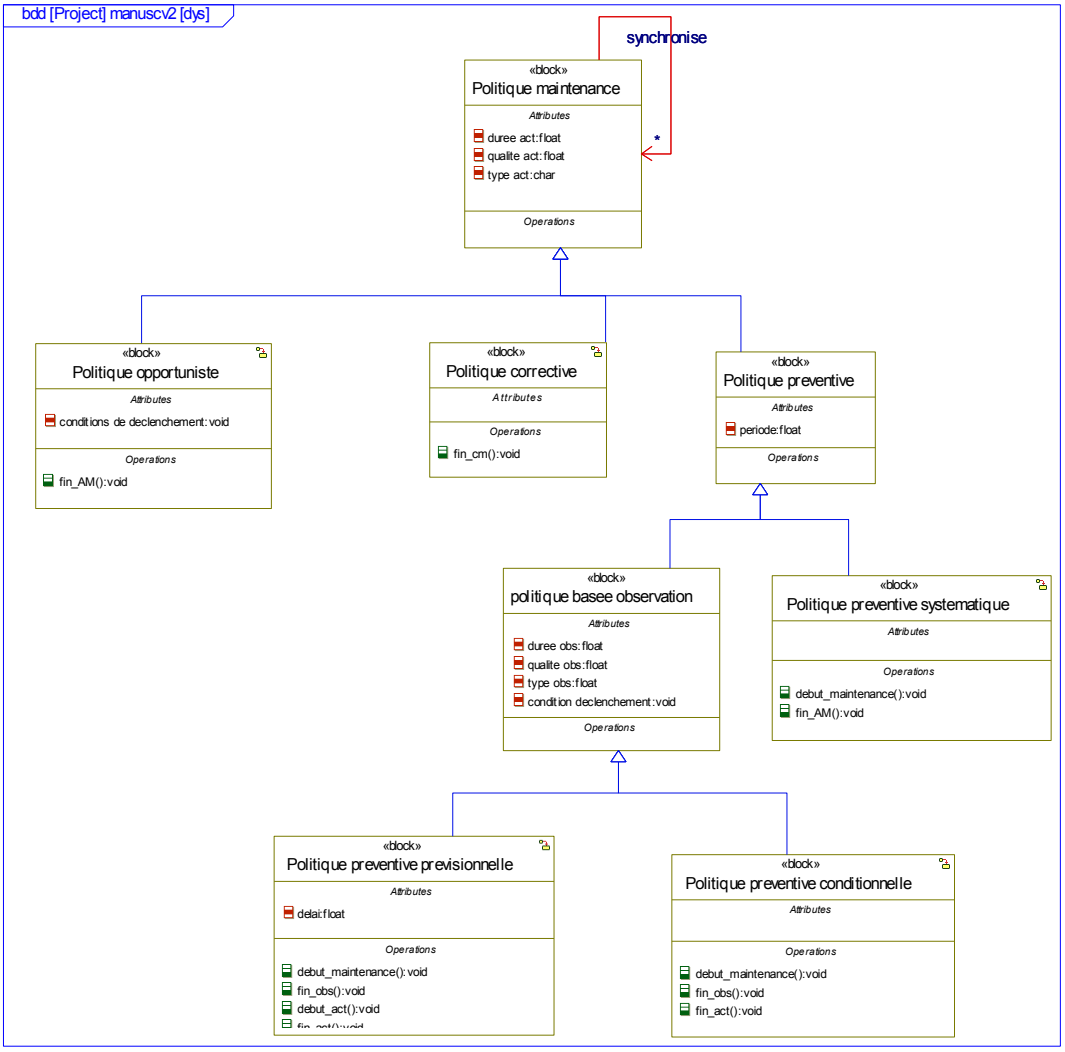


Figure 28 – BDD relatif aux politiques de maintenance

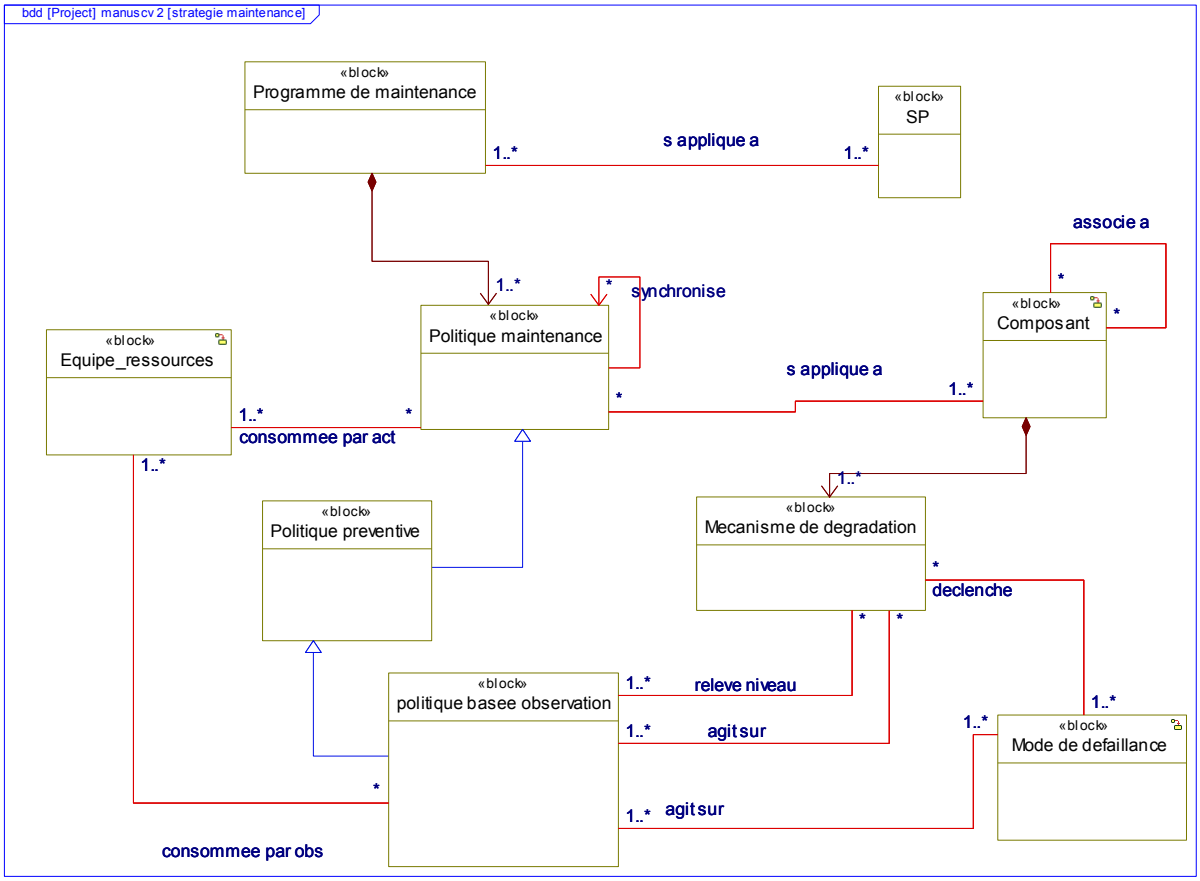


Figure 29 - Extrait du BDD reliant les blocs partiels des différents axes à travers une politique basée sur des observations

IV.6. Modélisation des interactions dynamiques entre les concepts de connaissance

Les diagrammes de séquences permettent de modéliser les interactions dynamiques identifiées du Tableau 10 au Tableau 13. Sur chaque *lifeline* modélisant un bloc, une opération interne peut se modéliser par un *message réflexif*. Ces *messages* déclenchent alors, par l'intermédiaire d'un *message entre lifeline*, l'occurrence d'une évolution forcée sur l'autre bloc, modélisée par un *message réflexif* sur la *lifeline* cible. Dans le cas où ces transitions sont conditionnées, alors un *opérateur* modélisant un chemin optionnel et sa condition peuvent être ajouté. Ces diagrammes sont utilisés pour modéliser des instances de *blocs*. Dans notre cas, l'intérêt de ces diagrammes est de modéliser des aspects génériques. Il est donc nécessaire de modéliser, pour l'ensemble des scénarios d'interactions entre les blocs, les SD associés.

Le Tableau 24 reprend les correspondances entre la connaissance identifiée et les éléments sémantiques des SD SysML équivalents.

Tableau 24 - Correspondances entre les éléments de connaissance et les éléments sémantiques des SD SysML

Connaissance	Elément SysML équivalent
Concept	*Lifeline*
Transition déclencheur	Message bouclé
Transition forcée engendrée	Message vers la *lifeline* en interaction puis message bouclé sur la *lifeline* en interaction
Condition	Opérateur « optionnel »

De par leur nature de concept en interaction entre le SS et le SP, les politiques de maintenance fournissent un exemple intéressant pour illustrer l'utilisation des SD car elles font intervenir des relations dynamiques à la fois simples et conditionnées. Supposons la politique de maintenance corrective soumise au scénario suivant : elle est associée à un mode de défaillance déclenchant le début de sa préparation. Une équipe de ressource constituée d'une seule ressource réutilisable est nécessaire à la réalisation de l'AM, et l'AM est imparfaite.

Ainsi, d'après les relations du Tableau 11, le début de l'AM est conditionnée par l'occurrence d'un mode de défaillance. A l'arrivée de celui-ci, la politique rentre dans son état de préparation par l'intermédiaire du *message bouclé* « SC_debut_maintenance », comme modélisé Figure 30. Nous nommons les *messages* entre *lifelines*, dont les noms n'ont pas été spécifiés au chapitre 2, par le même nom que le *message* qu'ils déclenchent.

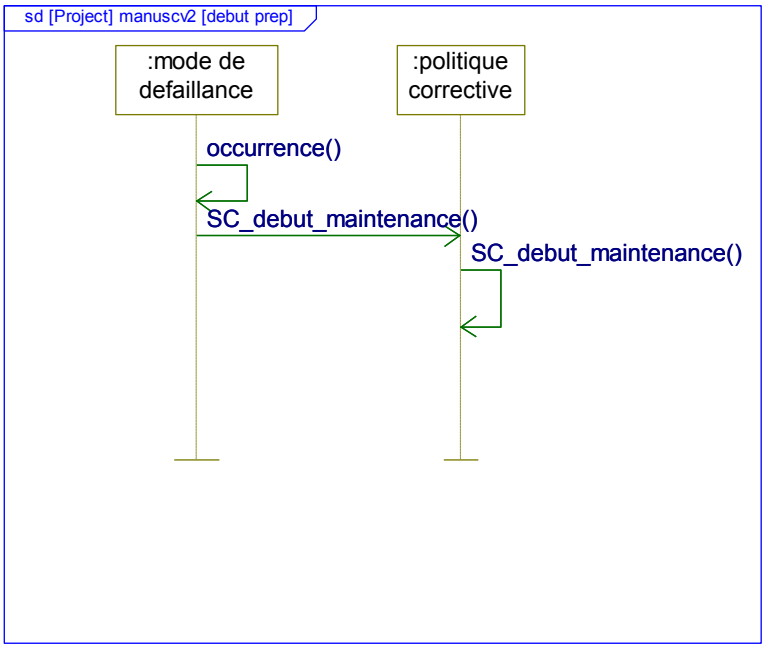


Figure 30 – SD relatif au déclenchement d'une politique corrective

Le début de cette préparation entraîne à la fois la constitution de l'équipe de ressources et la requête d'une ressource, notée « start_job ». Cette requête est cependant dépendante de la disponibilité de la ressource, comme modélisé Figure 31 avec un opérateur modélisant la disponibilité de la ressource. L'opérateur optionnel SysML modélisé par un rectangle nommé « opt » permet de modéliser cette dépendance.

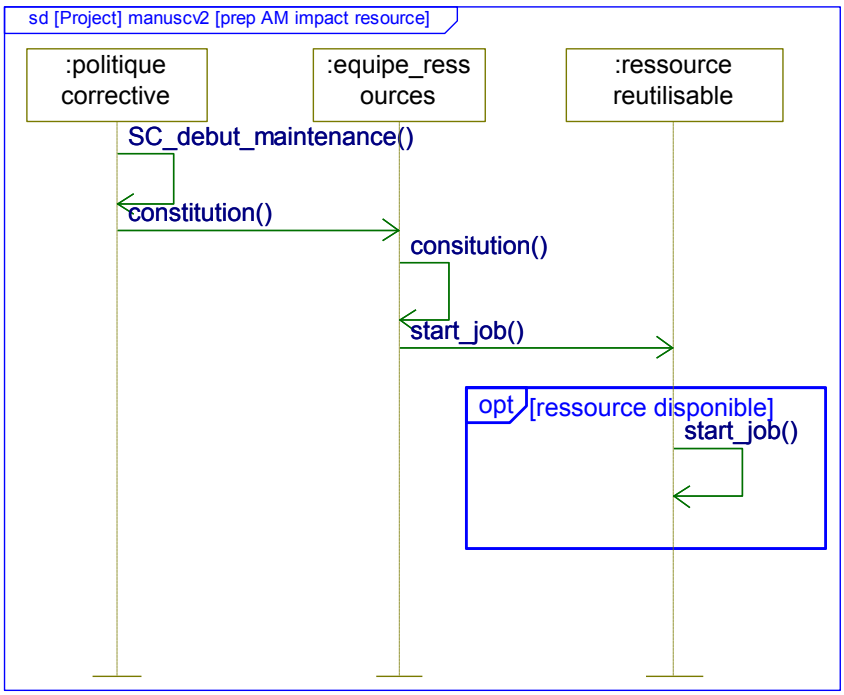


Figure 31 – SD relatif à la préparation d'une politique corrective

Si l'ensemble des ressources constitutives de l'équipe sont disponibles, alors la politique de maintenance corrective peut être déclenchée et atteindre son état « en_cours », modélisant le fait que l'action de maintenance est en cours d'exécution. Cependant, cette transition est fonction d'une interaction paramétrique puisque son déclenchement dépend des conditions de déclenchement propres à l'équipe de ressources. Ainsi, elle est détaillée lors de la définition des interactions paramétriques entre les concepts.

Enfin, à la fin de l'AM, la ressource est renvoyée en stock, le mode de défaillance disparait, le composant retourne dans son état de fonctionnement et potentiellement le mécanisme de dégradation associé peut revenir dans un état de fonctionnement. Cette dernière interaction est cependant conditionnelle à la qualité de l'AM, comme modélisé Figure 32.

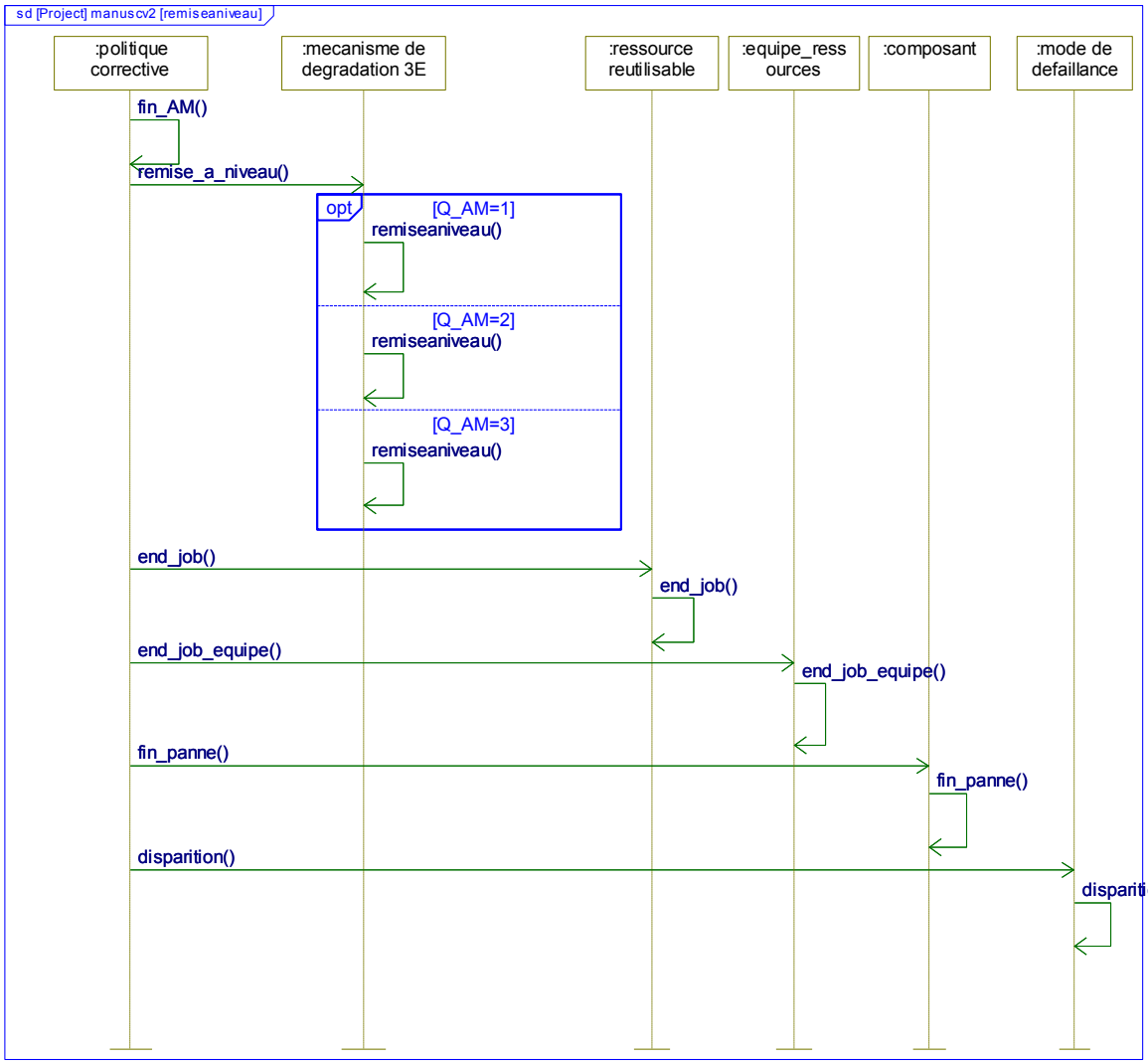


Figure 32 – SD relatif à la fin d'une politique corrective

En procédant de la même manière pour l'ensemble relations dynamiques répertoriées dans les Tableau 10 à Tableau 13, alors l'ensemble des SD a été modélisé. L'étape suivante consiste donc à modéliser les interactions paramétriques.

IV.7. Modélisation des interactions paramétriques entre les concepts de connaissance

Chaque interaction paramétrique identifiée Tableau 14 est donc notée dans les PD sous la forme d'une équation de contrainte Les caractéristiques et états des concepts rentrant en jeu sont modélisés sous la forme de *constraintparameter*, attachés aux blocs en question repris dans les PD, et reliés au *constraintblock* modélisant l'équation de contrainte. Les *constraintparameter* ne permettent cependant pas de leur attribuer un sens, et de savoir s'ils sont fonctions ou résultats d'une équation de contrainte (les PD sont des graphes non orientés). Ainsi, nous supposons leur sens donné par l'équation de contrainte. Pour étayer ce point, nous avons suivi la convention de lecture classique des entrées/

sorties d'un processus, c'est-à-dire que les entrées sont modélisées à gauche du *constraintblock* et les sorties à droite.

Ces diagrammes sont traditionnellement utilisés pour modéliser des instances de *blocs*. Dans notre cas, l'intérêt de ces diagrammes est de modéliser des aspects génériques. Cette hypothèse est faite, en supposant que chaque bloc est générique et déclinable selon les *cardinalités* des *associations* entre *blocs*. Par exemple, si la probabilité d'occurrence d'un mode de défaillance est fonction des états de trois mécanismes de dégradation, alors ce paramètre rentrera trois fois en compte dans l'équation de la contrainte.

Le Tableau 25 reprend les correspondances entre la connaissance identifiée et les éléments sémantiques des PD SysML équivalents.

Tableau 25 - Correspondances entre les éléments de connaissance et les éléments sémantiques des PD SysML

Connaissance	Elément SysML équivalent
Concept	Bloc
Caractéristique ou état rentrant en jeu dans l'équation	Constraintparameter
Caractéristique résultante de l'équation	Constraintparameter
Equation	Constraint et sens de la constraint d'un constraintblock

Dans l'exemple de la qualité de la maintenance, celle-ci dépend du type d'AM, ainsi que de la compétence/ qualité des ressources destinées à accomplir cette action. Chaque *constraintparameter* est donc équivalent à la value type du bloc équivalent, et la relation de contrainte est notée dans le *constraintblock* (cf. Figure 33). Afin de rester décorrélé de tout formalisme de simulation stochastique, une forme générique a été donnée à l'équation et donc ne contraint pas le langage utilisé pour la simulation stochastique à être d'un type ou d'un autre. Le choix de l'équation sera précisé par la suite afin d'être cohérent avec la sémantique du langage de simulation retenu (par exemple, les équations continues de dégradation ne pourront être modélisées avec le langage ADF).

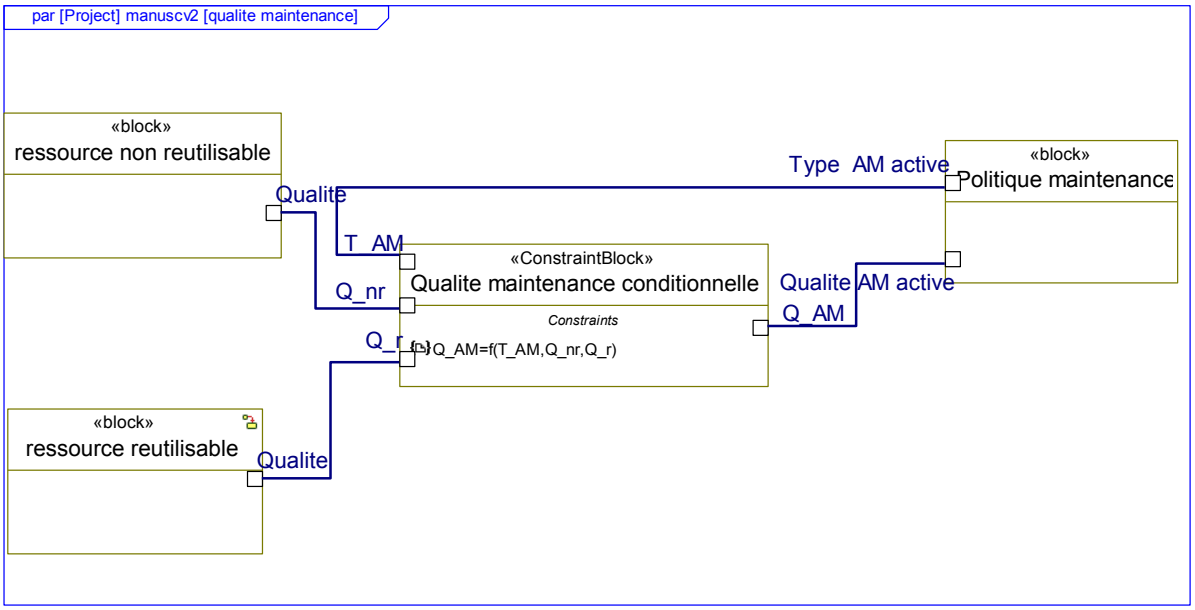


Figure 33 – PD relatif à la qualité de l'AM

Dans le cas du profil de défaillance, le PD est modélisé Figure 34. Ce profil peut être fonction des états d'autres mécanismes de dégradation, ainsi que des états des conditions environnementales et opérationnelles. Afin de conserver une vision générique, un seul bloc de chaque type avec des value type entrant en jeu dans l'équation a été noté. Lorsque l'on passera à la modélisation d'un système particulier, alors nous serons capables de préciser exactement le nombre de blocs de chaque type entrant en jeu.

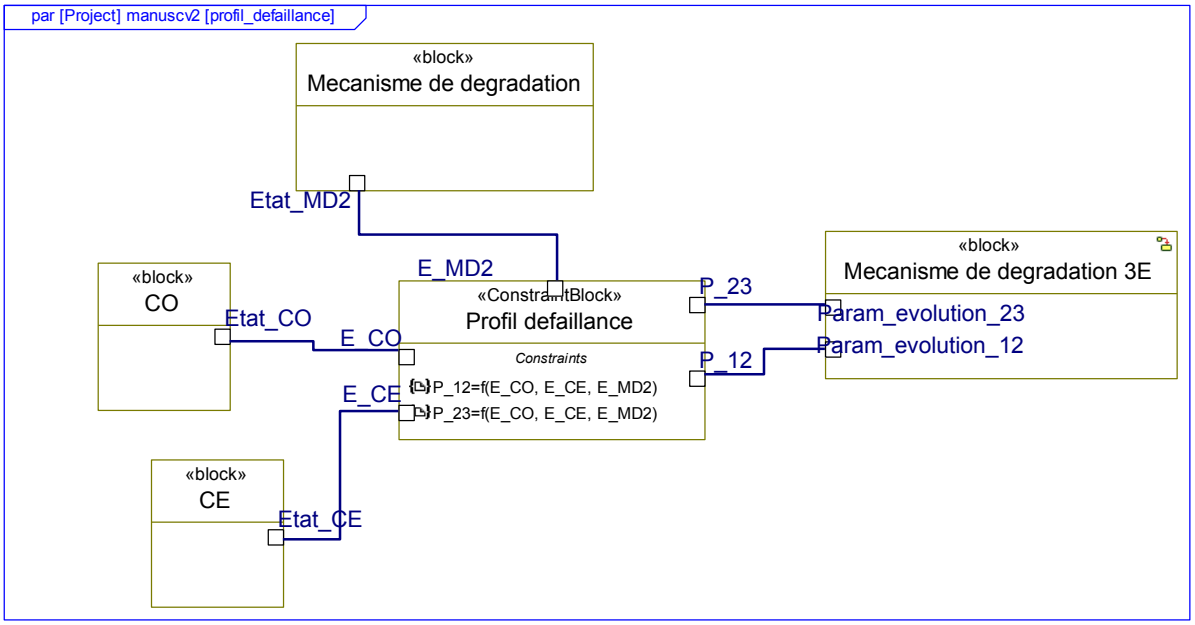


Figure 34 – PD relatif au profil de défaillance

Pour les PD, nous allons également modéliser l'exemple du profil fonctionnel. Les *constraintparameter* modélisant ces flux sont habituellement pris en compte dans les diagrammes internes de blocs dans le langage SysML, par l'intermédiaire des flow port. Comme nous avons choisi

de ne pas représenter ces diagrammes, il est nécessaire de préciser la fonction de transfert modélisant la façon, à partir d'un concept « fonction », d'évaluer la fonction de transfert de celle-ci.

Ces flux font par ailleurs l'objet d'un traitement spécial lors de l'étape de mapping entre les éléments d'intérêts des deux langages, puisque le langage ADF permet une modélisation intuitive de ce type d'équation de contrainte faisant intervenir des flux. En reprenant les interactions paramétriques du Tableau 15, la fonction possède deux caractéristiques pouvant varier : ses lois et paramètres d'évolution, et son flux sortant. Nous appelons l'équation de contrainte permettant modéliser la fonction de transfert depuis des flux d'entrées vers un flux de sortie, « profil fonctionnel ». Cette contrainte est particulière puisqu'elle est interne au *block* « fonction ». Par la suite nous parlerons d'équations de contrainte informationnelles dans les cas précédents et d'équations de contraintes fonctionnelles dans ce cas.

L'équation de contrainte peut alors être représentée dans un PD, comme modélisé Figure 35.

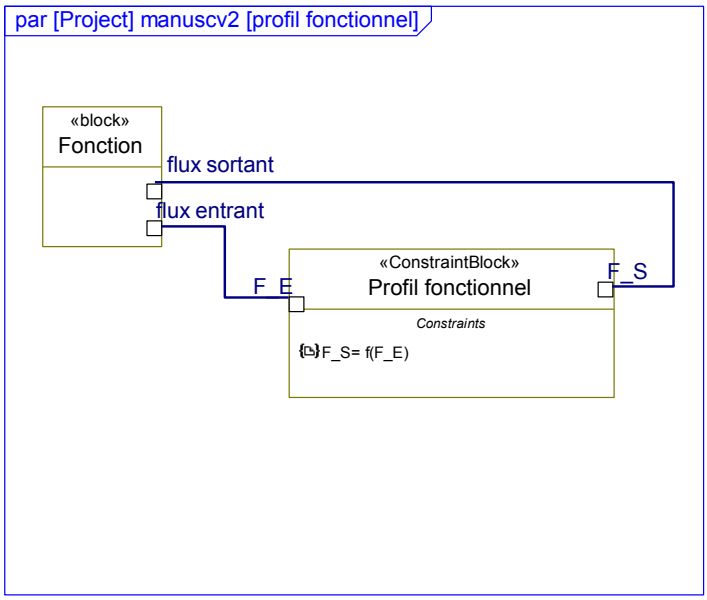


Figure 35 – PD relatif au profil fonctionnel

IV.8. Construction de la partie comportementale du modèle SysML4CMPQ

Il est à présent nécessaire d'agréger les différents diagrammes issus des différentes étapes de modélisation précédente pour en faire un modèle unique. Les STM SysML peuvent, au sein d'une même vue, représenter l'ensemble de l'information modélisée par les SD, en plus de celle des STM partiels. Ces STM permettront également de compléter les BDD partiels suite à l'ajout des différentes transitions forcées. La vision paramétrique (les PD) du modèle doit cependant être conservée afin de pouvoir modéliser le comportement de certaines grandeurs également présentes dans les STM (paramètres des transitions, valeur de certaines conditions…).

Pour construire ces STM, nous adoptons certaines syntaxes présentes dans l'outil Rhapsody afin de permettre l'exécution de ces diagrammes en vue d'une vérification du modèle. De plus, cette exécution permet, sous l'outil Rational Rhapsody, la création systématique des diagrammes d'états à

partir des SD. Ainsi, la cohérence sémantique entre les différents diagrammes est assurée. La démarche peut être simplifiée en l'appliquant à notre démarche puisque des STM partiels sont déjà disponibles. La démarche de construction des STM complets adoptée est la suivante :

1. Se baser sur les STM partiels modélisé précédemment.

2. En partant des transitions existantes initialement dans les STM, identifier les messages déclenchés par ces transitions dans les SD. Chacun de ces messages déclenché possède un élément SysML équivalent dans les STM : les messages envoyés. Ils requièrent dans Rhapsody de spécifier le nom du message envoyé et le bloc en interaction. Comme précisé dans notre démarche, les messages envoyés portent le même nom que les messages qu'ils déclenchent.

3. Pour chaque message ainsi envoyé, modéliser sur le STM en interaction, un état d'attente au sein de l'état cible avec la syntaxe « Waitfor « nom du message envoyé » ». Relier l'état précédent du STM cible avec une transition portant le nom du message envoyé.

Ainsi, l'ensemble des STM complets peut être construit. Dans l'exemple de la politique corrective, chaque message envoyé n'entraîne pas de condition particulière. La transition « debut_AM » est elle déclenchée par le fait que l'opérateur est effectivement disponible, de manière cohérente avec les SD Figure 30 à Figure 32. Les autres messages envoyés, rassemblés dans des états fictifs « fic » (présents à des fins de clarté, cf. Figure 37), sont tous envoyés vers les STM cibles. Dans ces STM cibles, ils permettent à la transition équivalente d'atteindre un état noté « Waitfor ».

Dans les cas du mécanisme de dégradation, seules les transitions liées à sa dégradation physique sont initialement représentées. Il est cependant nécessaire de prendre en compte la remise à niveau de celui-ci liée à la maintenance, dont le SD est représenté Figure 32.

Cette condition dépend donc de la qualité de l'AM notée « Q_AM », et dont la valeur est calculée dans un PD associé. Ainsi, l'élément SysML relatif à la condition (cercle autour d'un « C ») permet de modéliser cette condition, comme modélisé Figure 36.

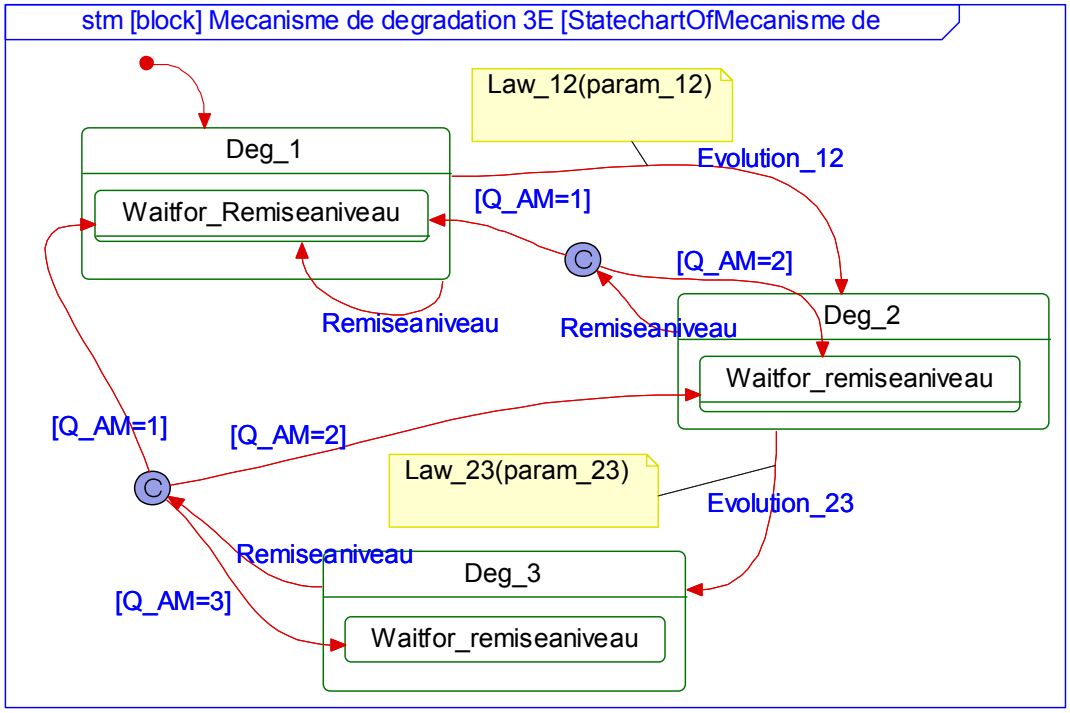


Figure 36 – STM complet d'un mécanisme de dégradation à 3 états

Dans le cas de la politique de maintenance corrective, de nombreux messages sont envoyés suite au déclenchement des transitions. Conformément à la démarche de construction des STM proposés, la Figure 37 modélise son STM complet.

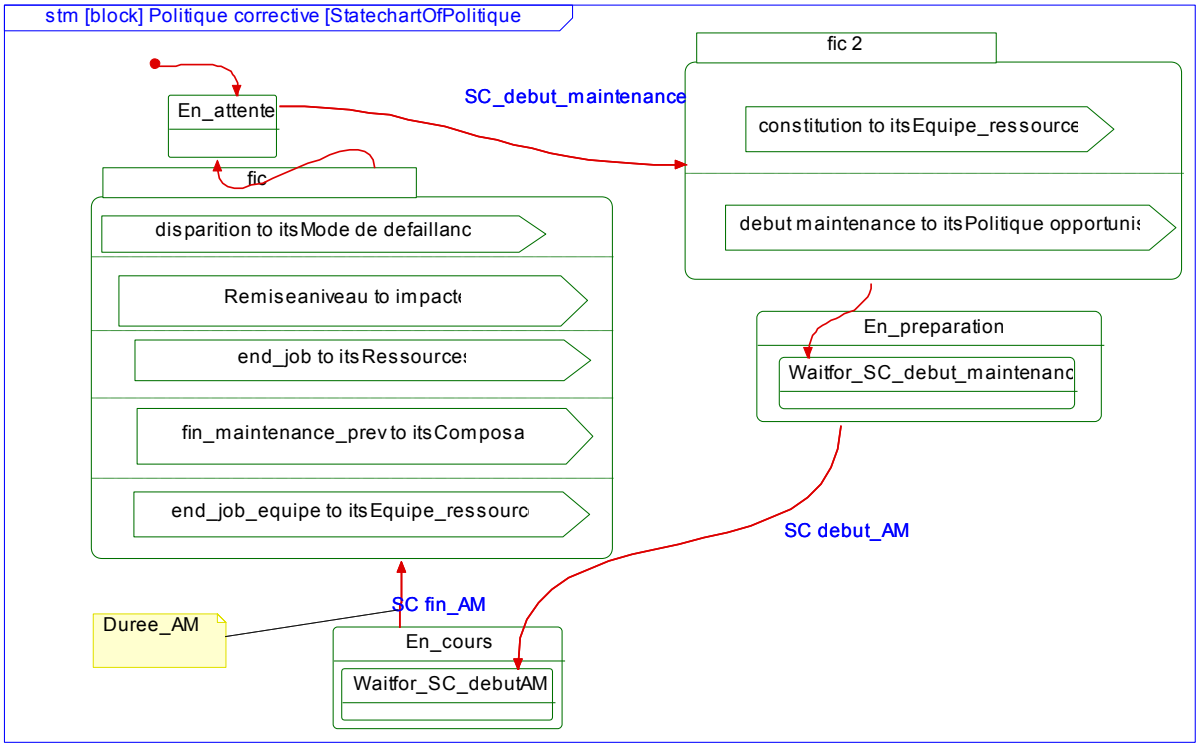


Figure 37 – STM complet d'une politique de maintenance corrective

En procédant de manière identique pour chaque bloc, alors chaque STM complet peut être construit. Une fois ces STM modélisés, alors chaque bloc peut se compléter automatiquement grâce à l'outil Rhapsody, en y ajoutant les opérations créées par cette étape de modélisation. Le BDD formé des blocs complet est également disponible.

IV.9. Vérification et validation du modèle SysML4CMPQ

En référence à la section II.7, il est à présent nécessaire de vérifier et valider le modèle SysML4CMPQ. Le modèle complet a été validé avec un expert EDF spécialiste de la CMPQ. Au cours de multiples réunions, la sémantique modélisée dans les différents diagrammes a été expliquée et discutée avec un expert, ce qui a permis de mettre en place un processus itératif de validation statique (principe d'un cycle auteur-lecteur). La vérification du modèle n'a quant à elle pas été effectuée dans ces travaux. Pour ce faire, une possibilité consiste, à partir du modèle SysML4CMPQ, d'effectuer une démarche de rétro ingénierie pour reconstruire la connaissance métier. En comparant cette connaissance métier reconstruite avec la connaissance métier initiale, le modèle pourra être vérifiée. L'outil Rhapsody permet de faciliter cette démarche. En effet, à partir d'un STM, il est possible de générer automatiquement les différents scénarios possibles sous la forme de SD, ce qui constitue le passage inverse à celui utilisé dans notre démarche (où les STM sont construits à partir des SD). Ce point doit nécessairement faire l'objet de perspectives à ce travail.

IV.10. Conclusion

A partir de la connaissance générique identifiée sous la forme de concepts au chapitre 2, nous avons détaillé dans ce chapitre la démarche de construction du modèle SysML4CMPQ (cf. Figure 38), présent en annexe.

Ce modèle se compose d'un ensemble de diagrammes validés, permettant de supporter au mieux l'étape (1') de notre démarche globale pour l'obtention d'un modèle générique indépendant de tout cas particulier. Ces diagrammes sont les suivants :

- D'un BDD modélisant les interactions entre les différents blocs génériques relatifs à chaque concept de connaissance identifié au chapitre 1;
- D'un ensemble de STM annotés relatifs au comportement de chaque bloc. Ce comportement inclus les interactions possibles entre les différents blocs du modèle;
- D'un ensemble de PD modélisant l'ensemble des interactions paramétriques possibles entre les blocs.

Le modèle SysML4CMPQ peut alors être utilisé comme un modèle pivot, c'est-à-dire pouvant servir de base à une projection du modèle pour créer d'autres modèles formalisés dans d'autres langages plus dédiés à la simulation. A partir de celui-ci des mappings vers des langages de simulation peuvent être élaborés. Par exemple, (David et al. 2010) propose des règles de transformation vers le langage ADF, (Bernardi et al. 2002) propose des règles de transformation depuis un BDD vers des réseaux de Pétri.

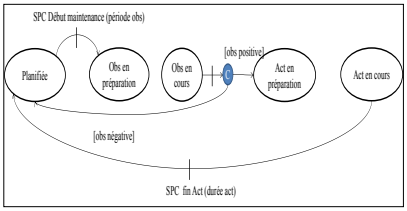
La connaissance contenue dans ce modèle peut se stocker sous la forme d'une base de données relationnelle. Cette base permet, pour l'étape (2), la création d'occurrences des concepts de modélisation rattachés au cas applicatif étudié. Toutefois, ce modèle ne sera pas exploitable pour calculer les KPIs puisque le langage SysML ne permet pas la simulation stochastique.

Ainsi, que cette base soit disponible ou non, il est à présent nécessaire d'aborder la problématique du mapping des éléments sémantiques d'intérêt du langage SysML vers le langage destiné à la simulation choisi dans notre démarche : le langage AltaRica DF, permettant de supporter à la fois les étapes (1') et (2) de la démarche, mais aussi l'étape (3) permettant l'évaluation des KPIs. Le chapitre 5 a pour objet de détailler ce passage depuis le langage SysML vers le langage ADF.

Chapitre 2: connaissance générique de la CMPQ

Vue interactionnelle

Vue comportementale des concepts



Transition	Concept en interaction	Transition forcée engendrée
Apparition mode défaillance	Maintenance corrective	Début maintenance (état « en attente » vers état « en préparation »)
	Composant	Panne (état « en marche » vers état « en panne »)
	Mode de défaillance	Occurrence forcée (état « non apparu » vers état apparu)
Début maintenance préventive (en général)	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »)
Fin AM active (quelle que soit la stratégie préventive)	Composant	Fin maintenance préventive (état « en maintenance préventive » vers état « en marche »
Fin AM active (stratégie corrective)	Composant	Fin panne (état « en panne» vers état « en marche »
Fin AM active (quelle que soit la stratégie)	Mécanisme de dégradation/ Symptôme	Remise à niveau ij (depuis un état de dégradation élevé i vers un état de dégradation équivalent ou moins élevé j)
Fin AM active (stratégie corrective)	Mode de défaillance	Disparition (état « apparu » vers état « non apparu »
Fin observation négative (quelle que soit la stratégie préventive)	Composant	Fin maintenance préventive (état « en maintenance préventive » vers état « en marche »
Début act (stratégie prévisionnelle)	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »

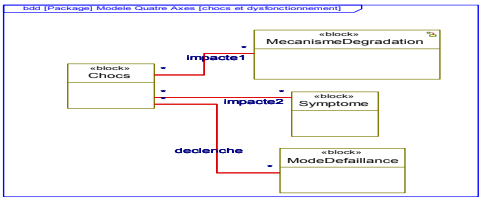
Vue statique

Concept	Caractéristiques	Comportement/ processus
Fonction	Flux entrants, flux sortants, fonction de transfert, loi et paramètre d'évolution	Continu/ discret : N états
Composant	Technologie, couts d'indisponibilité	Discret : N états (≥ 3 : en marche, en maintenance prev, en panne/ en maintenance corr)
Mode de défaillance	Maintenabilité, loi et paramètres d'occurrence	Discret : 2 états (apparu ou non)
Symptôme	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états
Mécanisme de dégradation	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états

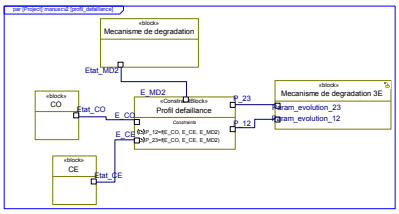
Démarche de construction du modèle SysML4CMPQ

Modèle SysMLde la connaissance générique de la CMPQ

BDD



PD



STM

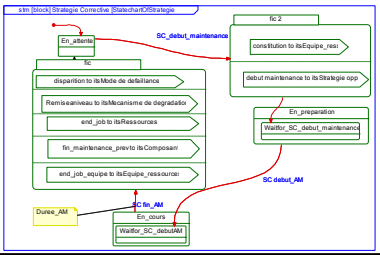


Figure 38 – De la connaissance générique au modèle SysML4CMPQ

Chapitre V. Méthodologie de construction du modèle pour la simulation stochastique supporté par le langage ADF à partir du modèle SysML4CMPQ

V.1. Introduction

Les diagrammes SysML construits au chapitre 4 permettent de modéliser la connaissance relative à la CMPQ sous la forme d'objets semi-formels sans perte sémantique par rapport à la connaissance initiale (chapitre 2) sous l'hypothèse d'un comportement à état transition. Si les diagrammes d'états supportent certains types de simulations (model-checking, simulation pas à pas), ils restent néanmoins inadaptés à la simulation stochastique, la méthode la plus efficace pour l'évaluation des KPIs. Cela les rend donc inadaptés pour supporter l'étape (3) de simulation stochastique du modèle. Comme proposé au chapitre 3, le langage ADF, reposant sur le formalisme des automates de mode (Rauzy 2002), a été privilégié pour cette étape. L'objectif de ce chapitre est donc de proposer un modèle générique à base du langage ADF permettant de supporter dans un premier temps (de manière alternative au modèle SysML4CMPQ) l'étape (1') de la démarche sur l'aspect création d'une librairie d'objets génériques atomiques (COTS). Dans un second temps, nous nous intéressons aux moyens permettant la modélisation progressive de systèmes génériques par assemblage des modèles atomiques génériques ADF. Ce second point permet de supporter l'étape (2) de la démarche et d'obtenir un modèle générique d'un système. Ce modèle est finalement instancié selon les spécificités du système particulier pour permettre la réalisation de l'étape (3) de la démarche. Ainsi, nous proposons :

- Un mapping depuis les éléments sémantiques SysML utilisés au chapitre 4 vers les éléments sémantiques ADF, afin de former une librairie d'objets génériques réutilisables, appelés atomes de modélisation,
- Un guide de modélisation d'un système global par assemblage des atomes de modélisation ADF.

Vis-à-vis de ces enjeux, nous présentons premièrement une description de la syntaxe et de la sémantique du langage ADF et des automates de mode, dans leur aspect atomique comme dans leurs aspects de composition. La sémantique ainsi définie est parallèlement mise en relation avec les prérequis liés à la simulation du modèle (transition à paramètres variables, transitions conditionnées…) de sorte à établir clairement les enjeux du mapping entre les langages.

Ensuite, après avoir repris les limites du langage ADF (impossibilité de modéliser des systèmes bouclés, représentation de fonctions de transfert booléennes uniquement), notre objectif est de spécifier les équations de contraintes des PD du modèle SysML4CMPQ de manière conforme avec la sémantique du langage ADF.

A partir des points précédents, le mapping entre les éléments sémantiques d'intérêt des 2 langages est réalisée, par l'intermédiaire de règles de correspondances algébriques. De ce mapping résulte la définition de l'ensemble de la bibliothèque ADF requise pour les études de CMPQ. Ensuite, les capacités du langage ADF pour gérer la complexité des systèmes sont abordées à travers la proposition d'une démarche permettant de combiner les modèles atomiques pour aboutir au modèle générique d'un système. Le modèle est finalement partiellement validé.

V.2. Description du langage AltaRica DF

V.2.1 Un formalisme basé sur les automates de mode

Ce paragraphe reprend la description formelle du langage définie dans (Rauzy, 2002). Un automate de mode est un automate à entrées/sorties. Il a un nombre fini d'états, appelés également modes. A chaque instant, l'automate se trouve dans un unique mode, duquel il peut évoluer à chaque occurrence d'évènement. A chaque mode, une fonction de transfert détermine la valeur des flux de sortie en fonction de celle des flux d'entrée.

Bien que cette thèse propose une utilisation alternative en modélisant des échanges de flux non seulement fonctionnels mais aussi « informationnels », c'est-à-dire impactant les transitions des automates, les automates de mode sont initialement décrits sur un exemple simple et une utilisation plus classique du formalisme. Ainsi, la Figure 39 illustre un automate de mode modélisant l'exemple simple d'un composant soumis uniquement à une politique de maintenance corrective, sans considération de ses dégradations. Il répond en outre aux hypothèses suivantes : il est supposé fonctionner parfaitement dans son état « en_marche », évoluer vers l'état « en_panne » à l'instant de l'occurrence de l'évènement « défaillance », rentre dans l'état « en_maintenance » à l'instant de l'occurrence de l'évènement « début_maintenance » et revenir dans l'état « en_marche » à l'instant de l'occurrence de l'évènement « fin_maintenance ». L'automate se trouve initialement dans l'état « en_marche ». La variable « outflow » modélise le flux fonctionnel sortant du composant et est exprimée en fonction de son flux d'entrée (dans notre modèle SysML4CMPQ, cette variable de est modélisée dans le *constraintblock* « profil_fonctionnel »).

A noter que nous ne disposons pas actuellement d'éditeur graphique d'automates de mode. Les figures représentant ces automates sont réalisées par un logiciel de diagramme ne supportant aucun outil de contrôle de la syntaxe – sémantique du formalisme.

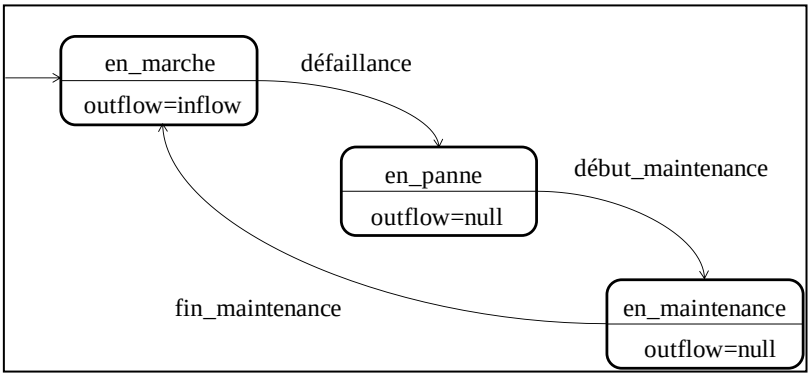


Figure 39–Automate de mode d'un composant simple

Par rapport à ce formalisme, nous proposons donc de repréciser sans ambiguïté ses éléments fondateurs tels qu'utilisés pour la CMPQ.

Ainsi, soit $\mathcal{D}$ l'ensemble fini de symboles appelés constantes et soit $\mathcal{V}$ l'ensemble fini de symboles appelés variables ($\mathcal{D} \cap \mathcal{V} = \emptyset$), $\mathcal{D}$ est appelé le domaine. Il est supposé donné une relation dom de $\mathcal{V}$

vers $2^{\mathcal{D}}$, tel que quel que soit *v* appartenant à $\mathcal{V}$, $dom(\mathcal{V}) \neq \emptyset$. $dom(\mathcal{V})$ est le domaine de la variable *v*, autrement dit $dom(\mathcal{V})$ est l'ensemble des valeurs possibles prises par *v*.

Soit $\mathcal{U} \subseteq \mathcal{V}$, nous désignons par dom ($\mathcal{U}$) des domaines des variables de $\mathcal{U}$: *dom*($\mathcal{U}$)=$\prod_{v \in \mathcal{U}} dom(v)$. En d'autres termes, dom ($\mathcal{U}$) est l'ensemble de toutes les valuations possibles des variables de $\mathcal{U}$. Soit U ∈ *dom(*$\mathcal{U}$*)*, nous désignons par *U[v]* la valeur de la variable v dans la valuation U et par U[v←c], c ∈ *dom(v)*, la valuation partout égale à *U* sauf en v , où elle peut être égale à c.

Un automate de mode est un 9-uplet $\mathcal{A} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, \Sigma, \delta, \sigma, I\}$, où :

- $\mathcal{D}$ et dom sont le domaine et la fonction du domaine définis précédemment,
- S, F^{in}, F^{out} sont trois sous-ensembles disjoints deux à deux de $\mathcal{V}$. Les variables de S, F^{in} and F^{out} sont respectivement appelées variables d'état, flux d'entrée et de sortie,
- Σ est un ensemble fini de symboles appelé évènements,
- δ est une fonction partielle de $dom(S) * dom(F^{in}) * \Sigma$ vers $dom(S)$. δ donne les valeurs suivantes des variables d'états depuis leurs valeurs courantes, les valeurs des flux d'entrée et l'évènement dont l'occurrence entraîne un changement de mode,
- σ est une fonction totale de $dom(S) * dom(F^{in})$ vers $dom(F^{out})$. σ donne les valeurs des flux de sortie depuis la valeur actuelle des variables d'état et la valeur des flux d'entrée,
- *I* appartient à $dom(S)$ et est appelé le mode initial.

En reprenant l'automate de mode présenté Figure 39 et en supposant que les flux d'entrée et de sortie appartiennent à l'ensemble {null, moyen, nominal} (cet ensemble peut être bien plus conséquent pour permettre par exemple l'expression d'un flux de sortie selon un pourcentage du flux d'entrée), il peut être décrit formellement de la manière suivante :

:

- $dom(S)$ ={en_marche , en_panne , en_maintenance},
- F^{in} ={inflow} et F^{out} ={outflow} avec dom(F^{in})=dom(F^{out})={null, moyen, nominal},
- Σ={défaillance, début_maintenance, fin_maintenance},
- σ et δ sont définis graphiquement Figure 39,
- *I*={en_marche}.

Le langage de modélisation ADF, permettant de supporter le formalisme des automates de mode dont la sémantique a été précédemment définie, se compose de 7 champs principaux. Ils sont relatifs à la sémantique des automates de mode, et seront notés en **gras** dans la suite du manuscrit. Ces champs sont les suivants :

- Le champ **node**, où l'objet est défini. Ce champ est équivalent à $\mathcal{A}$,

- Le champ **state**, où l'ensemble des états accessibles dans un node est défini. Ce champ est équivalent à *dom(S)*,
- Le champ **flow**, où les différents flux transitant dans un **node**, leur direction et leur type (réel, booléen…) sont définis. Les flow in sont équivalents à F^{in} . Les flow out sont équivalents à F^{out} ,
- Le champ **event**, où les différents évènements d'un node sont définis. Ce champ est équivalent à Σ,
- Le champ **trans**, où les impacts des différents event sur les state sont définis. Ce champ est équivalent à σ ,
- Le champ **assert**, où la fonction de transfert d'un node est modélisée, en fonction des états courants des nodes et de leurs flux d'entrée et de sortie. Ce champ est équivalent à δ ,
- Le champ **init**, où l'état initial du node est défini. Ce champ est équivalent à l'ensemble *I*.

Les aspects liés à la simulation stochastique du langage sont liés à l'outil utilisé pour compiler le code. Il existe également des instructions dédiées à la composition des automates de mode. Elles seront décrites après avoir posé les fondements de ces compositions.

V.2.2 Les automates de modes et la CMPQ

En revenant aux besoins liés à la simulation d'un modèle relatif à la CMPQ, le langage ADF et les automates de mode ont été choisis notamment pour leur capacité à modéliser des aspects liés :

- à des transitions aux paramètres variables et dépendants d'autres automates de mode,
- à des transitions simultanées entre automates de mode,
- à des transitions composées et ou complexes.

Afin de prendre en compte des transitions à paramètres variables, il est nécessaire de détailler plus précisément les fonctions σ et δ. En effet, c'est la fonction δ qui permet de définir, selon un état, un flux d'entrée, et un évènement, la prochaine valeur de dom(S) atteinte par l'automate de mode (elle définit donc les transitions). Le flux d'entrée intervenant dans cette fonction permet en effet de prendre en compte une donnée pouvant venir d'un autre automate de mode (par exemple son état), et de la faire influer sur la transition. Ainsi, nous adaptons, tout en restant conforme avec la sémantique du langage, l'utilisation classique des automates de mode, en faisant influer les flux d'entrées d'un mode, non pas sur la fonction de transfert σ, mais sur les transitions entre états. Il est cependant nécessaire de créer un nombre d'évènements $e_i \in \Sigma$ équivalent au combinatoire des flux d'entrée. De tels flux d'entrée sont par la suite appelés « flux informationnels », afin de contraster avec les flux fonctionnels classiquement modélisés par les automates de mode.

Concernant la fonction σ, il est nécessaire de l'adapter puisque cette fonction est totale, c'est-à-dire que son domaine correspond à son ensemble de départ. Ainsi, puisque les flux d'entrée peuvent dans notre cas être définis comme impactant uniquement sur une transition, il est nécessaire de créer des flux de sortie fictifs, permettant de correctement définir σ. La Figure 40 représente ce point pour deux automates de mode A_1 et A_2. L'automate de mode A_1 représente l'utilisation détournée, mais n'allant pas à l'encontre de la sémantique du formalisme, des automates de mode. La notation σ_{fic} est utilisée car les flux créés sont fictifs. L'automate de mode A_2 représente l'utilisation classique des automates de mode dans le cas où les flux d'entrée n'impactent pas les transitions de l'automate.

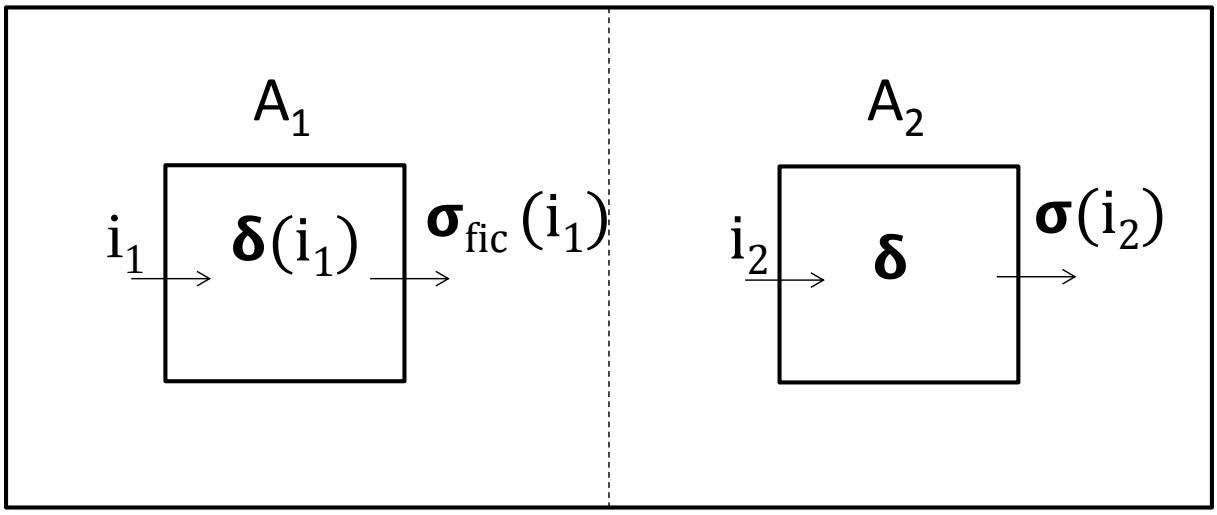


Figure 40 – Création de flux de sortie fictifs pour la modélisation des contraintes informationnelles par les automates de mode

Les automates de mode ont été également choisis pour leur capacité à modéliser les deux autres types de transitions. Pour montrer cette faisabilité, il est nécessaire de détailler leurs aspects de composition, et leurs aspects quantitatifs.

V.2.3 La composition des automates de mode

Le choix du langage ADF et des automates de mode pour la CMPQ fait au chapitre 3 est en partie basé sur sa capacité à décrire un système sous une forme hiérarchique de composants réutilisables. Les automates de mode peuvent être assemblés par le biais de trois opérations : les produits libres, les connections et les synchronisations. Celles-ci seront décrites tout d'abord de manière formelle (A. Rauzy 2002), puis de manière informelle où le « sens physique » des différents éléments sémantiques manipulés sera décrit.

V.2.3.1 Les produits libres

Soit $\mathcal{A}_1, \ldots, \mathcal{A}_n$ n automates de mode ($\mathcal{A}_i = \{\mathcal{D}, dom, S_i, {F^{in}}_i, {F^{out}}_i, \Sigma_i, \delta_i, \sigma_i, I_i\}$). Nous pouvons supposer sans perte de généralité que leurs vocabulaires sont distincts, c'est-à-dire que quels que soient i et j, 1≤i<j≤n, $(S_i \cup {F^{in}}_i \cup {F^{out}}_i) \cap (S_j \cup {F^{in}}_j \cup {F^{out}}_j) = \emptyset$ et $\Sigma_i \cap \Sigma_j = \emptyset$.

La composition parallèle de $\mathcal{A}_1, \ldots, \mathcal{A}_n$ désignée par $\prod_{i=1}^{n} \mathcal{A}_i$, est un automate de mode $\mathcal{A} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, \Sigma, \delta, \sigma, I\}$ tel que :

$S = \bigcup_{i=1}^{n} S_i$, $F^{in} = \bigcup_{i=1}^{n} {F^{in}}_i$, $F^{out} = \bigcup_{i=1}^{n} {F^{out}}_i$, $\Sigma = \bigcup_{i=1}^{n} \Sigma_i$

δ est obtenu en élevant les δ_i au niveau de $\mathcal{A}$: soit $S_1 \in dom(S_1), \ldots, S_n \in dom(S_n)$; soit $I_1 \in dom({F^{in}}_1), \ldots, I_n \in dom({F^{in}}_n)$; et soit $T_i \in dom(S_i)$ et $e \in \Sigma_i$ tel que $T_i = \delta_i(S_1, I_i, e)$. Alors $\delta(S_1, \ldots, S_n, I_1, \ldots, I_n, e) = \{S_1, \ldots, S_{i-1}, T_i, S_{i+1}, S_n\}$

De la même manière σ est obtenu en élevant les σ_i au niveau de $\mathcal{A}$:

$\sigma(S_1, \dots, S_n, I_1, \dots, I_n) = \{\sigma_1(S_1, I_1), \dots, \sigma_n(S_n, I_n)\}$

Enfin, $I = \{I_1, \dots, I_n\}$.

En d'autres termes, le produit libre, ou composition parallèle, consiste à grouper un ensemble d'automates de mode. Cette opération est interne, en effet le produit libre d'un automate de mode est lui-même un automate de mode. Cette opération permet la description hiérarchique de systèmes ou d'ensemble d'objets sous la forme d'une arborescence. La Figure 41 schématise cette opération d'un point de vue graphique sur des composants A_1 et A_2 identiques à l'automate décrit dans l'exemple précédent : il s'agit simplement de représenter une boite englobant ces composants, i et o symbolisant les flux d'entrée et de sortie.

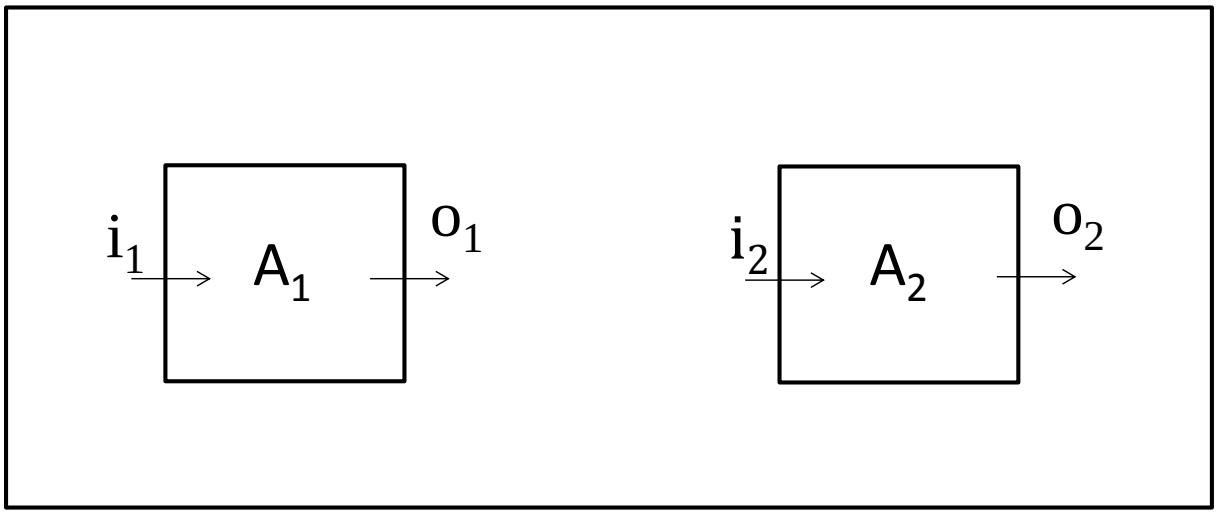


Figure 41 – Produit libre entre automates de mode

V.2.3.2 Les connections

Soit $\mathcal{A} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, \Sigma, \delta, \sigma, I\}$, soit $i \in F^{in}$ et soit $o \in F^{out}$. o est dit causalement indépendant de *i*, si la propriété suivante est vérifiée.

$\forall\ S \in dom\ (S), \forall\ I \in dom(F^{in}), \forall\ c\ \in dom(i),$

$\sigma(S, I)[o] = \sigma(S, I[i \leftarrow c] = [o]$

Cette définition est cependant très dure à vérifier. Heureusement, il existe de très simples conditions syntaxiques pour vérifier l'indépendance causale. Par exemple, il suffit que i n'intervienne pas dans les équations définissant la valeur de *o*.

Soit $\mathcal{A} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, \Sigma, \delta, \sigma, I\}$, un automate de mode, soit o $\in F^{out}$ et soit $i_1, \dots, i_k \in F^{in}$ tel que o est causalement indépendant de $i_1, \dots, i_k$ et dom(o) ⊆ dom(i_1),… dom(o) ⊆ dom(i_k).

Soit $F_*^{in} = F^{in} \backslash \{i_1, \dots, i_k\}, soit\ S\ \in dom(S)\ et\ soit\ I \in dom(F_*^{in})$. Nous désignons par $I_{S,o/i_1,\dots,o/i_k}$ la valuation de F^{in} telle que :

$$I_{S,o/i_1,\dots,o/i_k}[v] \stackrel{\text{def}}{=} \begin{cases} I[v] si\ v\ \in\ F_*^{in} \\ \sigma(S,I')[o]\ sinon \end{cases}$$

Où *I'* est l'extension de I dans la valuation de F^{in} (par définition, $\sigma(S,I')$ est la même, quelle que soit l'extension I').

L'automate de mode $\mathcal{A}$ dans lequel $i_1,\dots,i_k$ sont connectés à *o* est l'automate de mode $\mathcal{A}_{o/i_1,\dots,o/i_n} = \{\mathcal{D}, dom, S, F_*^{in}, F^{out}, \Sigma, \delta', \sigma', I'\}$, où $\delta'\ et\ \sigma'$ sont définis comme :

Pour tout $S\ \in dom(S), I \in dom(F_*^{in}) et\ e\ \in\ \Sigma, si\ \delta(S, I_{S,o/i_1,\dots,o/i_n}, e)$est défini, alors $\delta'(S,I,e) est\ défini\ et\ \delta'(S,I,e) = \delta(S, I_{S,o/i_1,\dots,o/i_n}, e)$.

Pour tout $S\ \in dom(S), I \in dom(F_*^{in}), \sigma'(S,I) = \sigma(S, I_{S,o/i_1,\dots,o/i_n})$.

En pratique, pour effectuer une connection, il suffit de substituer *o* aux i_j. Une connection peut donc être simplement vue comme le fait de renommer un flux. Ainsi, les connections permettent de restreindre le comportement du produit libre défini précédemment. En effet, une connection consiste à relier le flux d'entrée d'un composant à son flux de sortie. Cette opération fait cependant apparaitre une contrainte principale du langage ADF identifiée au chapitre 3 : la nécessité d'éviter les boucles entre les connections. Ce point est à prendre en compte avec attention lors de la définition des bibliothèques d'objets ADF, puisque nous ne manipulons pas uniquement des flux fonctionnels mais aussi des flux informationnels. En faisant le parallèle avec la théorie des graphes, un ensemble d'automates de mode reliés pouvant être vu comme un graphe orienté, il faudra faire attention à ne pas modéliser de circuits dans le modèle complet *M* (c'est-à-dire $\forall\ (\mathcal{A}_1,\dots,\mathcal{A}_N) \in M, \mathcal{A}_1 \neq \mathcal{A}_N$) pouvant être assimilé à un graphe orienté.

Par exemple, il est impossible de modéliser deux mécanismes de dégradations en interdépendance par exemple. Plus généralement, les interactions paramétriques entre deux composants ne pourront être représentées que de manière unilatérale (si un mécanisme de dégradation A influe sur la cinétique d'évolution d'un mécanisme de dégradation B, alors le mécanisme de dégradation B ne peut influer sur la cinétique d'évolution du mécanisme de dégradation A). Cette hypothèse n'est donc pas réaliste en regard de la connaissance générique. Cependant, nous faisons l'hypothèse réaliste qu'il est rare de disposer d'un REX permettant de quantifier une interdépendance entre deux mécanismes de dégradation.

La Figure 42 reprend la Figure 41 en y incluant une connection telle que définie précédemment entre les composants A_1 et A_2.

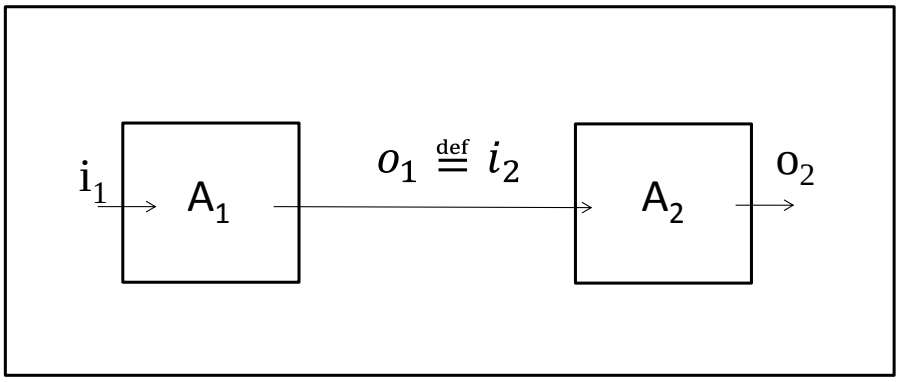


Figure 42 – Connections entre automates de mode

V.2.3.3 Les synchronisations

Un des principaux points forts d'ADF est les synchronisations. Elles consistent à franchir un ensemble de transitions simultanément. Les transitions à franchir simultanément sont définies au niveau du produit libre des automates de mode, et impliquent la suppression des transitions synchronisées de leurs automates de mode initiaux. Une transition globale est alors définie pour l'ensemble du produit libre des automates de mode. Dans le contexte de la CMPQ, elles permettent donc de modéliser les transitions simultanées.

Par exemple, soit A l'automate de mode d'un composant identique à celui de la Figure 39 et R l'automate de mode d'un opérateur de maintenance. Le fait de synchroniser l'évènement « défaillance » et l'évènement « start_job » signifiant le début du travail de l'opérateur consiste à transformer deux transitions en une pour créer une synchronisation symbolisant une réparation. Ainsi, cette synchronisation est faite de la manière suivante, en utilisant la notation « nom de l'automate.état » pour définir les états et une flèche annotée « nom de l'automate.transition » pour définir les transitions:

$A.state=en\_marche \xrightarrow{A.défaillance} A.state=en\_panne$

$R.state=en\_attente \xrightarrow{R.start\_job} R.state=en\_travail$

deviennent

$$\begin{pmatrix} A.state = en\_marche \\ R.state = en\_attente \end{pmatrix} \xrightarrow{réparation} \begin{pmatrix} A.state = en\_panne \\ R.state = en\_travail \end{pmatrix}$$

Plus formellement, pour définir les synchronisations, il est nécessaire d'établir des notations de compatibilité et de compositions des valuations.

Soit *S* un sous ensemble de $\mathcal{V}$ et soit *S, S' et S'' ∈ dom (S). S' et S''* sont dits incompatibles par rapport à *S* si il existe *v* tel que *S[v], S'[v] et S''[v]* sont tous distincts. Autrement, ils sont dits compatibles.

La composition de *S'* et *S''* par rapport à *S*, désignée $S' o_S S''$, est créée si la valuation est définie de la manière suivante :

$$\forall v \in S, S' o_S S'' = \begin{Bmatrix} S'[v] si\ S'[v] \neq S[v] \\ S''[v] sinon \end{Bmatrix}$$

Propriété 1 (composition). Soit S un sous ensemble de $\mathcal{V}$ et soit $S \in dom(S)$. Appliquée aux valuations compatibles deux à deux par rapport à S, o_S commutative et associative.

Il est à présent possible de définir formellement la synchronisation :

Soit $\mathcal{A} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, \Sigma, \delta, \sigma, I\}$ un automate de mode et soit $\overrightarrow{e_1}, \dots, \overrightarrow{e_r}$ r vecteurs de synchronisations. Soit Σ' le sous ensemble de Σ des évènements intervenant dans les $\overrightarrow{e_i}$. La synchronisation de $\mathcal{A}$ avec $\overrightarrow{e_1}, \dots, \overrightarrow{e_r}$ est un automate de mode $\mathcal{A}\backslash\overrightarrow{e_1}, \dots, \overrightarrow{e_r} = \{\mathcal{D}, dom, S, F^{in}, F^{out}, (\Sigma\backslash\Sigma') \cup \{\overrightarrow{e_1}, \dots, \overrightarrow{e_r}\}, \delta', \sigma, I\}$, où δ' est défini de la façon suivante :

Quel que soit $e \in \Sigma\backslash\Sigma'$, S ∈ dom(S) et I ∈ dom($F^{in}$), si $\delta(S, I, e)$ est défini alors $\delta'(S, I, e)$ également et :

$\delta'(S, I, e) \stackrel{\text{def}}{=} \delta(S, I, e)$

Quel que soit $\overrightarrow{e_i}$ $\langle e_1, \dots, e_k \rangle$, S ∈ dom(S) et I ∈ dom($F^{in}$), si $\delta(S, I, e_1)$, ..., $\delta(S, I, e_k)$ sont définis et compatibles deux à deux, alors $\delta'(S, I, \overrightarrow{e_i})$ est défini et

$$\delta'(S, I, \overrightarrow{e_i}) \stackrel{\text{def}}{=} \delta(S, I, e_1) o_S \dots o_S \delta(S, I, e_k)$$

Certains champs complémentaires décrits précédemment (produits libres, connections et synchronisations) existent afin de pouvoir gérer l'assemblage des automates de modes. Ces champs se définissent à un niveau d'abstraction supérieur puisqu'ils ont pour but de modéliser les interactions entre différents **nodes**.

Ces champs sont les suivants :

- Le champ **sub**, qui permet de définir les entités de base appartenant à un sous-système,
- Le champ **assert** enrichi, qui permet de définir les connections entre composants,
- Le champ **sync**, où les synchronisations entre les différentes transitions sont modélisées.

Les **nodes** créés à partir de ces opérations appliquées à des **nodes** atomiques (formant ainsi des macro-**nodes**) sont eux-mêmes sont réutilisables. Pour les créer nous avons défini dans ce chapitre une procédure de construction de ces macro-**nodes**. Ces champs sont suffisants pour formaliser les atomes de modélisation de base de notre modèle supportés par ADF qui constitueront la base de la bibliothèque d'éléments atomiques réutilisables. Cependant, il est nécessaire de considérer la dimension quantitative du langage permettant l'instanciation des différents **nodes**, dissociée de la sémantique d'ADF et rendant possible la simulation stochastique.

V.2.4 Le langage ADF et la simulation stochastique

Les probabilités associées aux événements sont spécifiées dans un champ **extern** propre à l'outil complémentaire utilisé. Ce champ permet donc l'instanciation des différents **nodes**. Nous en donnons un aperçu des possibilités d'un tel champ dans le cas de l'outil à notre disposition, SIMFIA développé par EADS APSYS (*www.apsys.fr*). L'instanciation des **nodes** sera ensuite mise en pratique au niveau du cas d'étude. Ce champ **extern** comporte les instructions suivantes :

- l'instanciation des lois de probabilités associées aux **event** par l'intermédiaire du mot clé « law ». Dans SIMFIA, la syntaxe de cette instruction est la suivante : « law <nom_event>=nom_loi(paramètre(s)) ». Il est important de remarquer que, ces lois ne faisant pas partie intégrante du langage, celles-ci, ainsi que leur(s) paramètre(s) ne peuvent être définis comme variables. Ainsi, pour prendre en compte des **event** fonctions de l'état d'autres objets par exemple, il sera nécessaire d'éclater les transitions SysML en plusieurs **event**,
- l'instruction « priority » permettant de donner un ordre de priorité entre plusieurs **event** immédiats tirés en même temps. Ces priorités sont des entiers, et la priorité par défaut (0) est la plus faible. La syntaxe de cette instruction est la suivante : « priority{<nom_event>}=int »,
- l'instruction « bucket » permettant d'associer des probabilités de succès entre plusieurs **event** pouvant être tirés en même temps. Ces **event** doivent être au préalable définis avec une « law » de type « constant » suivie de leurs probabilités de succès, tirées aléatoirement. La syntaxe de cette instruction est la suivante : « bucket{<nom_event1> ,< nom_event2>…, <nom_eventN>}=call ».

Ces différentes instruction rendent possible sous cet outil la simulation stochastique par compilation du code ADF. Cependant seules certaines transitions complexes et composées peuvent être prises en compte. Ces transitions doivent être franchies à l'occurrence d'évènements discrets uniquement. Il est également possible, par la commande « bucket », de modéliser des transitions conditionnées. Cependant, il n'est pas possible de définir les deux en même temps (par exemple, une loi entraînant soit le passage à un état, soit le passage à un autre, ne peut être temporisée). Pour ce faire, il est nécessaire de créer des états fictifs dans les automates de mode, pour prendre en compte ce point. Ainsi, en reprenant les prérequis de modélisation établies au chapitre 2, seule le prérequis relatif à la modélisation de lois variables et complexes ne peut être prise en compte de manière exhaustive. En effet, SIMFIA borne notre étude à la prise en compte des processus discrets uniquement. Si une perte sémantique est alors inévitable par rapport au modèle SysML4CMPQ, ceci reste cohérent avec la simulation stochastique privilégiée dans le cas des études de sûreté de fonctionnement.

Ce champ **extern** comprend également des commandes de type « observer » et « predicate ». Cependant, ces champ n'interviennent pas dans l'aspect quantification du langage, mais indiquent les indicateurs à relever par le logiciel de simulation de l'outil. Ces commandes permettent d'évaluer l'ensemble des éléments nécessaires pour le calcul KPIs requis dont les règles d'évaluation ont été spécifiées au chapitre 2. En effet, si les couts globaux ne sont calculables qu'après une étape de post traitement, SIMFIA peut nous donner le nombre de fois qu'une transition a été tirée ou le temps d'attente dans chaque état sur une période donnée.

V.2.5 Conclusion sur le langage ADF

Le langage ADF, basé sur le formalisme des automates de mode, permet de respecter la majeure partie des conditions nécessaires à la simulation stochastique d'un modèle conforme au modèle SysML4CMPQ. En effet, il permet notamment de modéliser des transitions simultanées, des transitions fonctions d'états d'autres objets. Il permet également de modéliser certaines conditions complexes (conditionnées) ou composées.

Cependant, il est nécessaire, avant d'aborder le mapping entre les éléments sémantiques d'intérêts des deux langages et les équivalences entre les éléments SysML et ADF, de restreindre la forme donnée aux équations de contrainte (définies dans les PD) du modèle. Les points suivants synthétisent la manière de modéliser dans le langage ADF les prérequis de modélisation exigés.

Pour la modélisation des transitions fonctions d'états d'autres objets :

Chaque transition SysML doit être « éclatée » selon un nombre équivalent à chaque valeur possible prise par leur paramètre. Pour ce faire, il faut spécifier dans les diagrammes paramétriques une équation par combinaison de flux entrants impactant sur la transition.

Il sera donc nécessaire de créer un nombre de **nodes** supérieur au nombre de STM du modèle SysML4CMPQ. Ce nombre est équivalent au nombre de **flow in** potentiel par **node**. Dans le contexte de la CMPQ, cette perte de généricité conduisant certes à la formalisation d'un nombre de **nodes** supérieur au nombre de STM n'est cependant pas rédhibitoire. En effet, seuls quelques concepts sont sensibles à ce point (les concepts étant sujets à des interactions paramétriques impactant une de leurs caractéristiques), et parmi eux certains ne dépasserons jamais un nombre peu élevé de **flow in** (nous faisons l'hypothèse réaliste qu'il est dans la pratique très rare de prendre en compte beaucoup de facteurs dans ces interactions paramétriques). Les concepts les plus sensibles à ce point sont les concepts dysfonctionnels des composants (mécanismes de dégradation, mode de défaillance…) pour lesquels les transitions sont très souvent fonctions d'autres concepts. Ce point est illustré par la suite.

Pour la modélisation des transitions composées :

Ce type de transition est dans le contexte de la CMPQ illustré un tirage aléatoire composé avec une loi de type exponentielle par exemple. Il est nécessaire pour prendre en compte ce type de transition d'une part de créer des états fictifs permettant de modéliser une transition classique, puis, à partir de cet état fictif, de créer un nombre de transitions équivalent au nombre de résultat possible du tirage aléatoire. Ce point n'a pas d'incidence du point de vue de la fiabilité du résultat (le temps de passage dans l'état fictif est nul), mais a pour conséquence d'alourdir la simulation.

Pour la modélisation des transitions forcées :

Si ADF, par l'intermédiaire des synchronisations, permet de modéliser simplement des transitions forcées, il ne faut pas oublier le fait qu'une transition d'un **node** donné peut être synchronisée avec plusieurs transitions provenant de plusieurs **nodes** différents (par exemple, plusieurs politiques de maintenance peuvent s'appliquer à un seul **node**). Il est donc nécessaire à ce niveau de créer autant

transitions à synchroniser qu'il y a de **nodes** disposant de transitions déclenchant cette transition forcée.

Pour la modélisation des systèmes bouclés :

Ce type de système n'étant pas modélisable en ADF, il faudra vérifier pour chaque cas d'étude la condition de l'absence de circuit dans le graphe orienté équivalent au modèle du cas d'étude.

Si plusieurs travaux se sont intéressés aux transformations entre les méta-modèles des deux langages (David, 2009), aucun d'entre eux n'avait vocation à effectuer la simulation stochastique. En ce sens, les précédents points fournissent un canevas pour la phase de mapping depuis SysML vers ADF à réaliser, pouvant éventuellement parfaire une transformation déjà existante entre ces deux langages. Afin de rendre ce mapping possible, il est tout d'abord nécessaire de restreindre la forme des équations de contrainte du modèle SysML4CMPQ de sorte à les rendre compatible avec le langage ADF. En appliquant la même démarche à un autre langage que le langage ADF, il sera nécessaire de trouver la forme adéquate à donner à ces équations.

V.3. Restriction de la forme des équations de contraintes du modèle SysML4CMPQ

Les PD ont deux objectifs : tout d'abord permettre à l'expert d'évaluer des paramètres en fonction d'autres, et d'autre part d'être compatibles avec la sémantique du langage ADF lorsque le résultat de ces évaluations est variable, pour les *constraintblocks* fonctionnels comme pour les *constraintblocks* dits informationnels. Les équations de contrainte de ces *constraintblocks* seront prises en compte de manière différente en ADF : dans le champ **assert** en ce qui concerne les *constraintblocks* fonctionnels, et dans le champ **trans** pour les *constraintblocks* informationnels. Par la suite, nous ne nous intéresserons qu'aux *constraintparameter* variables, c'est-à-dire relatifs aux états des *blocs*. Les autres sont laissés dans les PD et sont prises en compte dans le calcul des résultats des équations de contrainte, mais non prises en compte dans les équations discrètes définies dans le langage ADF.

Dans le modèle SysML4CMPQ, les *constraintblock* informationnels font intervenir uniquement des entrées équivalentes aux états de chaque objet. En effet, seuls les états de ceux-ci peuvent être des variables dans le modèle. De plus, la syntaxe particulière d'ADF n'autorise que la considération de variables booléennes. Ainsi, afin de faciliter le passage vers ce langage constituant notre support à la simulation stochastique, un canevas doit être défini pour permettre à l'utilisateur de ne paramétrer que des équations effectivement simulables.

Les paramètres intervenants dans ces *constraintblock* informationnels sont donc équivalents aux états des diagrammes d'états mis en jeu. La grandeur impactée n'est en effet pas le **flowout** du **node**, mais bien la probabilité de transition.

De plus, afin de ne prendre en compte que des variables booléennes, les équations doivent être définies sous la forme d'équations logiques. Ainsi, les variables d'états doivent être précédées de l'identifiant de l'objet auquel elles appartiennent. Cette restriction de la sémantique des équations de contrainte implique une explosion combinatoire du nombre de *constraintblocks*. Il y aura en effet un nombre de *constraintblock* équivalent au combinatoire des *constraintparameters* entrants intervenant

dans ce bloc. Cette explosion combinatoire n'est cependant pas rédhibitoire puisqu'il est dans la pratique très difficile de quantifier l'impact de nombreux facteurs influents a priori.

Les équations doivent donc avoir la forme suivante. Soit f l'équation de contrainte permettant de calculer le *constraintparameter* en sortie o selon les *constraintparameters* $(i_1, i_2, \dots, i_n)$, et g_j^{i1} une valeur accessible du *constraintparameter* i_1 :

$$f: o = if\{i_1 = g_j^{i1}\ and\ i_2 = g_j^{i2}\ and \dots\ i_n = g_j^{in}\ then\ X\}$$

X est la valeur résultante de l'équation prise par o. Cette forme implique de transformer la fonction générique f en un nombre élevé d'équations discrètes équivalentes.

Dans le cas des *contraintblocks* fonctionnels, la forme de l'équation est identique. La différence réside dans le fait que les *constraintparameter* en entrée ne sont pas forcément des états des blocs associés. Selon la nature du constraintblock (fonctionnelle ou informationnelle) auquel elle appartient, l'équation de contrainte sera codée différemment en ADF.

En suivant les précédentes règles pour la restriction de la forme des équations de contrainte, il est possible d'établir une correspondance entre le modèle SysML4CMPQ et le modèle générique ADF requis. En effet, toutes les équations de contraintes sont formulées de manière compatible avec la sémantique d'ADF, ce qui permet de ne pas modéliser de la connaissance sans aucun sens en ADF, et donc d'assurer la fiabilité du modèle dans le calcul des KPIs. Etant donné que les diagrammes d'états restent les mêmes, l'ensemble du modèle SysML4CMPQ peut maintenant être annoté afin de modéliser le plus finement possible les équivalences entre les deux langages.

V.4. Définition des notations

Dans le but de systématiser le passage du langage SysML vers le langage ADF, il faut annoter l'ensemble des éléments sémantiques ADF et SysML d'intérêts. Afin de pouvoir être implanté pour la création de passerelles entre outils logiciels, il est nécessaire de disposer d'algorithmes et de règles structurés. Les éléments à définir sont donc l'ensemble des éléments du langage ADF afin d'automatiser la construction de la bibliothèque de **nodes** générique, pour pouvoir la réutiliser dans le but de modéliser un système particulier. En restant cohérent avec les notations utilisées pour la définition du langage, les éléments d'intérêt du langage sont annotés Tableau 26. Nous avons ajouté à ces éléments la notion de familles de **node,** équivalentes aux concepts de modélisation du modèle SysML4CMPQ.

Tableau 26 – Notations des éléments ADF

Notation	Elément ADF
$N\_\mathcal{A}$	Nombre des familles de **nodes** ADF à créer
$\mathcal{A} = \{a_i\}$	Ensemble des familles de **nodes** ADF
$a_{j(i)}$	Node *i* de la famille de **node** a_j

$\mathcal{S}^{a_j} = \{s_i^{a_j}\}$	Ensemble des **states** du **node** a_j
$N\_\mathcal{S}^{a_j}$	Nombre des **states** du **node** a_j
I^{a_j}	Etat initial du **node** a_j
$\Sigma^{a_j} = \{ e_i^{a_j}\}$	Ensemble des events du **node** a_j
$N\_\Sigma^{a_j}$	Nombre des events du **node** a_j
$\sigma_{kl}^{a_j} = \{\sigma_{kl,i}^{a_j}\}$	Ensemble des **trans** depuis l'état k vers l'état l du **node** a_j
$N\_\sigma_{kl}^{a_j}$	Nombre des **trans** depuis l'état k vers l'état l du **node** a_j
$\mathcal{F}in^{a_j} = \{fin_i^{a_j}\}$	Ensemble des **flowin** du **node** a_j
$N\_fin^{a_j}$	Nombre de **flowin** du **node** a_j
$\mathcal{F}out^{a_j} = \{fout_i^{a_j}\}$	Ensemble des **flowout** du **node** a_j
$N\_fout^{n_j}$	Nombre de **flowout** du **node** a_j
$\mathcal{N}^* = \{n_i^*\}$	Nombre de sous systèmes
$N\_n_i^*$	Nombre d'objets ADF appartenant au sous système n_i^*

Des éléments annexes (par exemple la taille des différents ensembles) ont également été annotés, afin de faciliter l'étape de création du code ADF.

Afin de pouvoir rétablir des correspondances entre ces éléments et les éléments sémantiques SysML, l'ensemble des éléments des diagrammes SysML doivent également être annotés. La Figure 43 et la Figure 44 décrivent graphiquement ces notations dans le cas respectivement des diagrammes d'état et des diagrammes paramétriques. Nous supposons ici que l'ensemble des *blocs* ciblés par les *messages envoyés* est pris en compte dans la modélisation afin de pouvoir les identifier de manière unique.

Les éléments ajoutés au formalisme (paramètres de certaines transitions) ont une importance lors du passage à ADF, c'est pourquoi ils sont annotés. Les transitions ont été « éclatées » en plusieurs catégories. Il n'est pas utile, pour le passage à ADF, de prendre en compte les sous états de type « Waitfor ». En revanche, les transitions forcées doivent être annotées différemment des transitions paramétrées puisque le langage ADF ne les prend pas en compte de la même manière (respectivement appartenant aux ensembles $\mathcal{T}f_{kl}^{a_j}$ *et* $\mathcal{T}_{kl}^{a_j}$). De plus, les conditions liées à ces transitions ont été annotées. Les *états* ont également besoin d'être annotés tout comme les messages envoyés. La Figure 43 annote graphiquement certains de ces éléments. Concernant la Figure 44 annotant les PD, les *blocs* doivent être annotés, tout comme les *constraintparameter*. Nous distinguons toutefois les *constraintparameter* résultats de la *constraint,* et ceux en étant les paramètres. En effet, les

constraintparameters en entrée d'un *contraintblock* sont équivalents, exceptés pour les *constraint* fonctionnelles, aux différents *états* de leurs *blocs* respectifs. Les équations de *constraintsblock* k sont notées f_k^j. Le Tableau 27 reprend l'ensemble des notations définies jusqu'à présent.

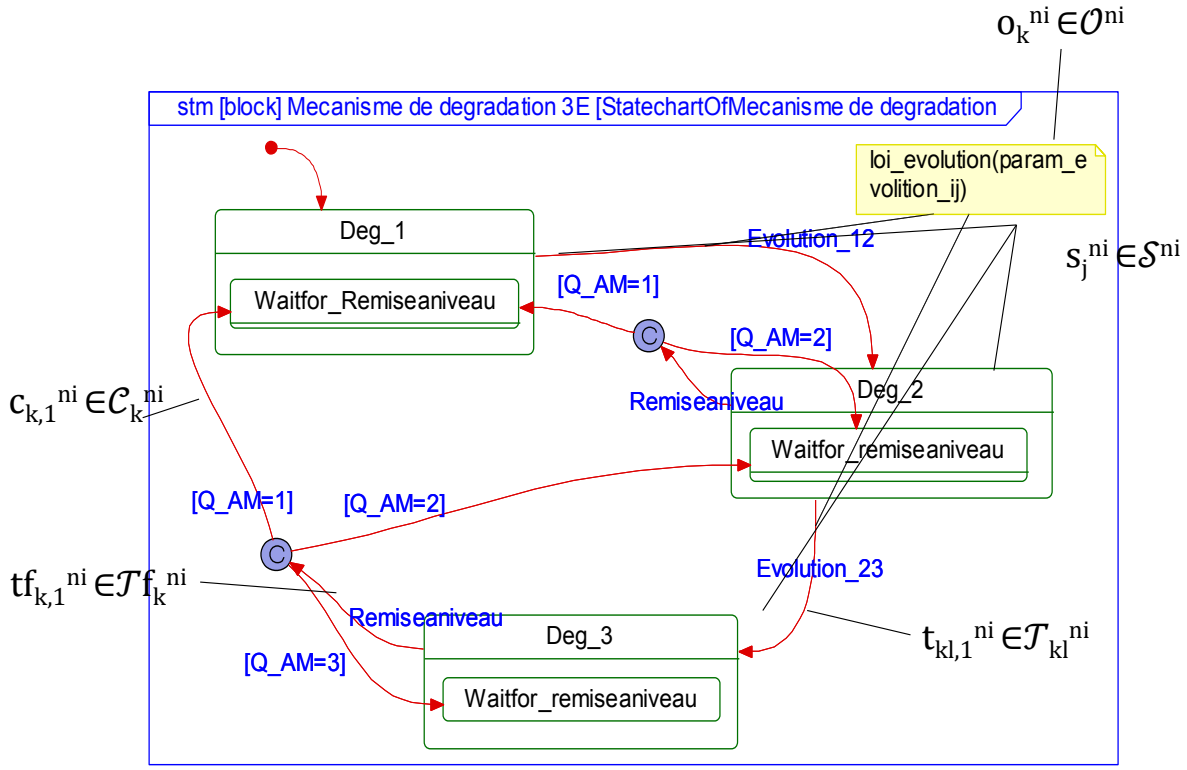


Figure 43 – Diagramme d'état du modèle SysML4CMPQ annoté

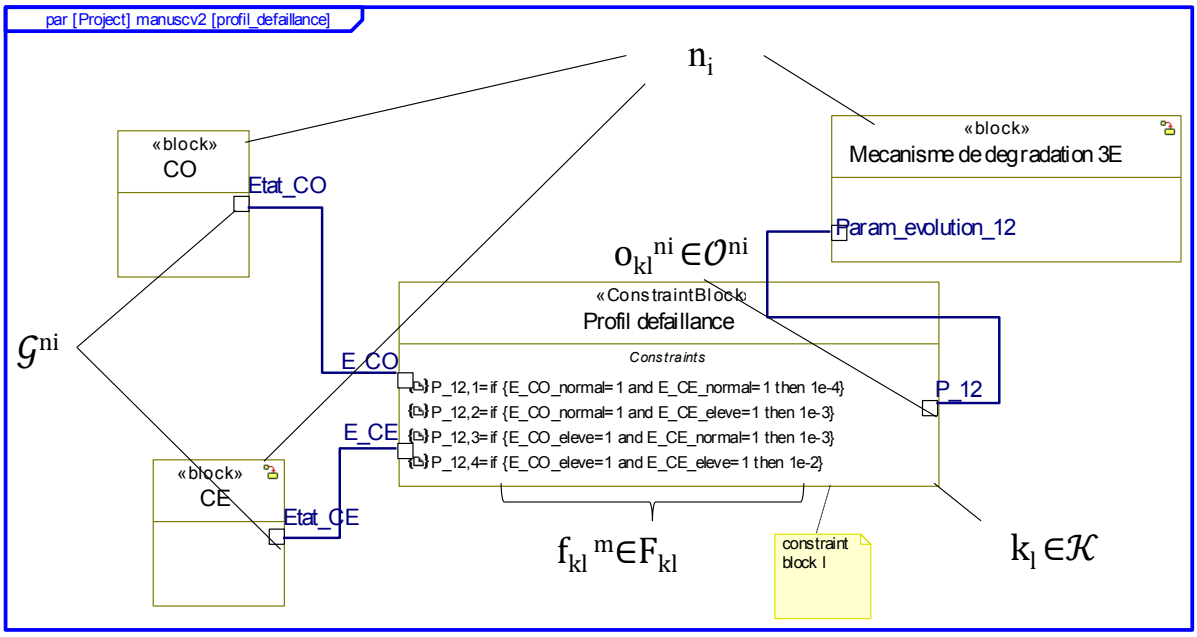


Figure 44 – Diagramme paramétrique du modèle SysML4CMPQ annoté

Tableau 27 - Notations des éléments issus des diagrammes SysML

Notation	Elément SysML
$N\_\mathcal{N}$	Nombre de diagrammes d'états / bloc du modèle SysMCMPQ
$\mathcal{N} = \{n_i\}$	Ensemble des diagrammes d'états/ bloc
$\mathcal{G}^{n_j} = \{g_i^{n_j}\}$	Ensemble des états du diagramme d'état/ bloc a_j
$N\_\mathcal{G}^{n_j}$	Nombre d'états du diagramme d'état/ bloc n_j
$\mathcal{T}_{kl}^{n_j} = \{t_{kl}^{n_j}, i\}$	Ensemble des transitions entre deux états k et l du diagramme d'état/ bloc n_j
$N\_\mathcal{T}_{kl}^{n_j}$	Nombre de transitions entre deux états k et l du diagramme d'état/ bloc n_j
$\mathcal{T}_k^{a_j} = \{t_k^{n_j}, i\}$	Ensemble des transitions depuis un état k du diagramme d'état/ bloc n_j vers une condition
$N\_\mathcal{T}_{kl}^{n_j}$	Nombre de transitions depuis un état k du diagramme d'état/ bloc n_j vers une condition
$\mathcal{T}f_{kl}^{a_j} = \{tf_{kl}^{n_j}, i\}$	Ensemble des transitions forcées entre deux états k et l du diagramme d'état/ bloc n_j
$N\_\mathcal{T}f_{kl}^{n_j}$	Nombre de transitions forcées entre deux états k et l du diagramme d'état/ bloc n_j
$N\_\mathcal{K}$	Nombre des *constraintblock* du modèle SysML4CMPQ
$\mathcal{K} = \{k_l\}$	Ensemble des *constraintblock* du PD k
$o_{k_l}^{n_j}$	*Constraintparameter* résultats d'un *constraintblock* k_l et appartenant au bloc n_j.
$\max(o_{k_l}^{n_j})$	Nombre de valeurs prises par le *constraintparameter* $o_k^{n_j}$
$\{c_{k,l}^{n_j}\}$	Transition conditionnée depuis un état k vers un état l
$F_{kl} = \{f_{kl}^m\}$	Ensemble des équations de contrainte du *constraintblock* k_l
$t_{initl}^{n_j}$	Transition initiale vers l'état l

Les différents éléments sémantiques ADF et SysML étant à présent annotés, des règles formelles pour le mapping entre les éléments sémantiques d'intérêt des deux langages peuvent à présent être définies. Nous nous intéressons tout d'abord à la création des atomes de modélisation équivalents aux concepts de connaissance, puis à la création des modèles de systèmes particuliers par assemblage des atomes.

V.5. Création des atomes de modélisation ADF

Ces différents atomes de modélisation constituent la base d'objets générique (de COTS) permettant de supporter l'étape (1') de la démarche de manière complémentaire au modèle SysML4CMPQ. Pour ce faire, chaque étape de création est illustrée selon un exemple relatif au mécanisme de dégradation. Nous décrivons ce passage des concepts du langage SysML (les éléments annotés Tableau 26) à ceux du langage ADF (les éléments annotés Tableau 27) de deux manières : de manière informelle et graphique, en utilisant la vision des automates de mode pour le langage ADF, en établissant des équations identifiant les équivalences entre les différents éléments sémantiques des deux langages.

V.5.1 Le champ node

Chacun des diagrammes d'état du modèle SysML4CMPQ modélise d'un point de vue comportemental une connaissance d'intérêt du modèle. Afin de pouvoir construire l'ensemble des systèmes et de leurs axes contributeurs, il est nécessaire de pouvoir manipuler dans le langage ADF ces différents diagrammes d'états. Or, chaque **node** ADF est :

- dépendant de ses **flows in**, variables selon les *constraintparameter* définis dans le modèle SysML4CMPQ,
- dépendant de ses objets en interactions. En effet, certaines transitions ne sont déclenchées que si un autre objet existe.

S'il est possible de contourner le second point en éclatant une transition forcée en N transitions forcées identiques, nous devons créer des familles de **nodes (**cf. Figure 45) pour prendre en compte les différents **flow in.** En effet, chaque équation de contraintes informationnelles nécessite d'être prise en compte dans chaque **node**, et ces équations sont variables selon les **flow in** qu'elle prend en compte.

Par exemple, l'ensemble des **nodes** présents en bibliothèque et manipulables par l'utilisateur relatifs au mécanisme de dégradation à 3 états pourraient être :

- Mécanisme de dégradation à 3 états avec un **flow in** (par exemple, une condition opérationnelle),
- Mécanisme de dégradation à 3 états avec deux **flow in** (par exemple, une condition opérationnelle et une condition environnementale),
- Etc…

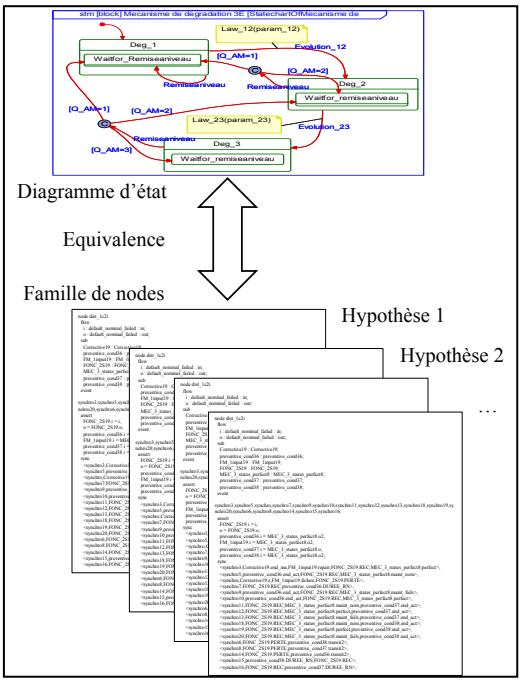


Figure 45 – Correspondances entre un diagramme d'état et une famille de nodes

Par la suite, nous utilisons la notion de famille de **nodes** pour regrouper ces différents **nodes** plus particuliers. Ainsi, il y a correspondance entre les familles de **nodes** et les diagrammes d'état du modèle SysML4CMPQ. Par conséquent :

$$N\_\mathcal{N} = N\_\mathcal{A}\,, et\ \forall\, a_i\ \in \mathcal{A}\,, \forall\, n_j\ \in \mathcal{N}, a_k = n_k\ \forall\ k\ \in [1, N\_\mathcal{N}] \cap \mathbb{N}$$

V.5.2 Les champs state et init

Les automates de mode reposent, comme les diagrammes d'états, sur un formalisme état transition. Cependant, la sémantique plus formelle des automates de mode ne permet pas la prise en compte des transitions complexes, c'est-à-dire à la fois dépendant du temps et d'une condition. Comme précisé lors de la définition des notations, chaque transition complexe a été éclatée en une transition temporelle vers une condition et deux conditions liées. Pour prendre en compte ces transitions successives, il faut donc créer en ADF des états fictifs, modélisant l'instant où la transition temporelle est franchie. Ensuite la transition conditionnée tirée est franchie instantanément.

Ainsi, pour chaque famille de **nodes**, s'il ne comporte pas de transition conditionnée :

$$N\_S^{\mathcal{A}} = N\_\mathcal{G}^N;\ \forall g_i^N \in \mathcal{G}^N, \forall s_j^{\mathcal{A}} \in \mathcal{S}^{\mathcal{A}}\ alors\ g_k^N = s_k^{\mathcal{A}}\ \forall\ k\ \in \left[1, N\_\mathcal{G}^{\mathcal{N}}\right] \cap\ \mathbb{N}$$

Et, pour chaque transition conditionnée, il convient d'ajouter les états fictifs suivants :

$$If\ \exists t_k^{n_j}\ \in \mathcal{T}_k^{n_j}, \qquad alors\ \exists\ s_{k'}^{a_j} = fic\_s_k$$

L'état initial du code ADF sera le même que l'état initial du diagramme d'état. Ainsi pour chaque famille de **nodes** :

$$\forall\ t_{initk}^{n_j}\ \ alors\ \ I^{a_j} = s_k^{n_j}$$

La Figure 46 montre graphiquement, en utilisant les automates de mode pour représenter le langage ADF, cette équivalence pour l'exemple du mécanisme de dégradation à 3 états modélisés **Figure 36** du chapitre 4. Ce diagramme d'état modélise un mécanisme de dégradation sujet à une maintenance imparfaite, pouvant être déclenchée par plusieurs politiques de maintenance différentes. En appliquant la règle ci-dessus, la famille de **nodes** équivalente dispose de 3 états fictifs supplémentaires, étant donné que chaque transition est conditionnée à la qualité de l'action de maintenance. L'état initial est l'état « Deg_1 ». La Figure 47 présente en parallèle une partie de code ADF équivalent, les états y sont manipulés par la variable « Var ».

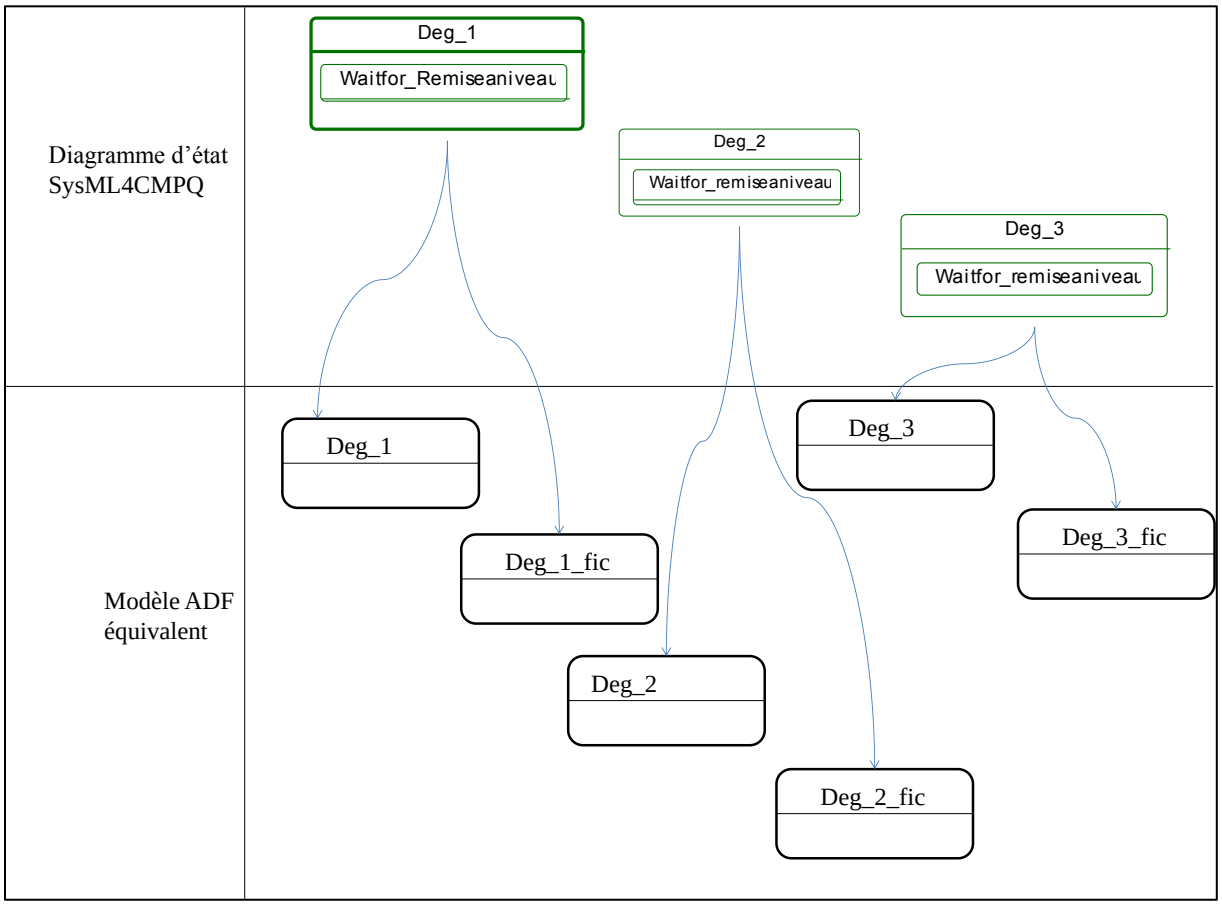


Figure 46 – Correspondances graphiques entre les états SysML et les états ADF

```
node mecanisme degradation 3 etats
  state
Var : {Deg_1, Deg_1_fic, Deg_2 Deg_2_fic, Deg_3, Deg_3_fic };
  init
Var:=deg_1;
```

Figure 47 - Code créé pour le mécanisme de dégradation

V.5.3 Le champ flow

Les **flows** correspondent aux flux entrants et sortants de l'objet, les **flowin** et les **flowout**. Traditionnellement, le langage ADF est utilisé pour modéliser des flux physiques entre composants.

Ainsi, la prise en compte des flux tels que définis dans les contraintes fonctionnelles se fait de manière intuitive. Or, notre proposition implique la prise en compte des flux informationnels entre objets. Les **flow** sont donc modélisés dans le modèle SysML par les *constraintparameter* définis dans les PD informationnels. Un *constraintparameter* étant un résultat d'une équation sera un **flowin** pour l'objet duquel il fait partie, tandis qu'un *constraintparameter* étant entrée de l'équation sera un **flowout** pour l'objet duquel il fait partie.

Dans ADF ces flow, bien que pouvant modéliser des réels, sont de nature discrète (par exemple « la fonction du composant est remplie à 70% ou 0% »). Il est donc nécessaire d'éclater ces flow en fonction du nombre d'états desquels ils sont fonction. Ainsi, un *constraint* parameter intervenant dans une équation de contrainte k_i peut prendre des valeurs restreintes à l'ensemble des états de son STM associé, puisque seuls ceux-ci sont variables dans le cas des *constraint* de natures informationnelles (par exemple, si l'état de la condition opérationnelle est nominal, alors le taux de dégradation est…).

Il n'est pas possible, dans le langage ADF, d'être générique sur ces **flows**. En effet, chaque **node** de chaque famille $a_{j(i)}$ possède son unique ensemble de flow.

Soit $\mathcal{S}^{n_j}$ les états intervenant dans F_{k_i}l'ensemble des équations de k_i, soit $o_{k_i}^{n_l}$ les résultats de l'équation appartenant au *bloc* n_i, les **flowin** et **flowout** se définissent de la manière suivante pour chaque **node** $a_{j(i)}$ (chaque famille de **nodes** possède en effet au moins i **nodes**) :

$$fout_m^{a_j(i)} = fin_m^{a_{l(i)}} = s_m^{n_j}, \forall m \in [1, N\_S^{a_j}]$$

Ainsi, les **flowin** et les **flowout** se définissent en parallèle sur les **nodes** impactés et impactant les équations F_{k_i}. En reproduisant la règle ci-dessus pour tous les blocks intervenant dans les équations F_k, l'ensemble des **flowin** et des **flowout** liés à cette équation peut être défini.

Dans le cas des *constraint* de natures fonctionnelles, soit $o_{k_i}^{n_l}$ la sortie d'un block « fonction » a_l intervenant dans le calcul de la sortie d'un autre block « fonction » a_j :

$$fin_m^{a_{j(i)}} = fout_m^{a_{l(i)}} = o_{ki\,m}^{n_l} \forall m \in [1, \max(o_{ki}^{n_j})]$$

En reproduisant ces opérations pour toutes les *constraint* possibles, une famille de **node** partielle peut être constituée. Chaque $a_{j(i)}$ est en effet particulier et *v* est pour l'instant identique au nombre de *constraintblock*.

Considérons à présent l'exemple d'un mécanisme de dégradation à 3 états dont le profil de dégradation n'est impacté que par une condition environnementale à 2 niveaux d'intensité (faible et élevée) et une condition opérationnelle à 2 niveaux d'intensité (faible et élevée).

Dans ce cas, 4 **flowin** doivent être définis pour ce mécanisme de dégradation, un pour chaque état pouvant rentrer en jeu dans le calcul de son profil de dégradation. Les **flow** étant, dans ADF, purement déclaratifs, cette transformation n'est donc pas représentable graphiquement selon la vue graphique des automates de mode. La Figure 48 montre la correspondance entre un PD particulier (le profil de dégradation d'un mécanisme de dégradation à 3 états, impacté par deux *constraintparameters*) et le

node de la famille des mécanismes de dégradation à 3 niveaux. De la même manière, selon les règles formulées précédemment, 2 **flowout** associés à leurs états seront définis dans chaque **node** contributeur, à savoir les conditions environnementales et opérationnelles.

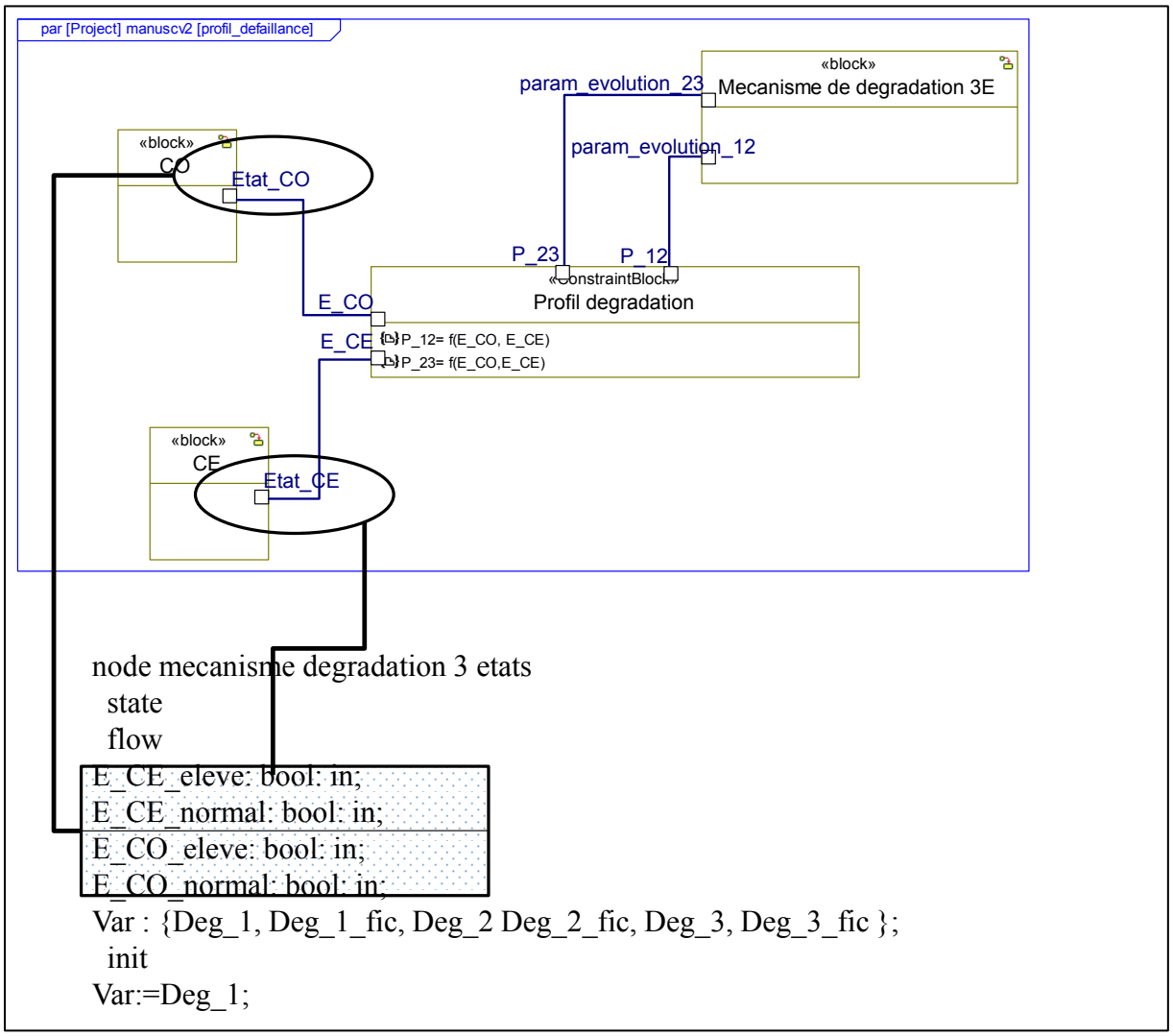


Figure 48 – Correspondances entre les flow ADF et les constraintparameter SysML

Par ailleurs, nous avons précisé au moment de la définition formelle du langage, l'obligation de définir des **flowout** fictifs dans le cas où seuls des **flowin** interviendraient dans un **node**. En effet, il faut au moins un flux de sortie par combinaison de flux d'entrée et d'états, et donc, dans le cas des flux « informationnels » uniquement :

$$\forall\, fin^{a_{l(i)}}, \qquad \exists\ fout^{a_{l(i)}}$$

Leur fonction de transfert **δ** est définie lors de la création du champ « assert ».

V.5.4 Les champs event, trans et extern

Les **event** correspondent à la partie déclarative des **trans** ADF**.** Les **trans** permettent de déclarer les différentes transitions relatives aux transitions du modèle état transition, depuis leur état de départ vers

leur état d'arrivée. Leurs paramètres de déclenchement sont déclarés dans le champ **extern** sauf dans le cas des transitions synchronisées.

Les transitions du modèle SysML4CMPQ peuvent être de plusieurs types :

- Les transitions fonctions d'un facteur constant;
- Les transitions fonctions d'un paramètre non constant donné par les PD,
- Les transitions conditionnées,
- Les transitions forcées.

V.5.4.1 Les transitions fonctions d'un facteur constant

C'est dans ce cas que la situation est la plus simple à définir. La transition est en effet à paramètre constant, et il y a équivalence entre les transitions SysML et les **trans** et **event** ADF. Ainsi, la **trans** créée par cet **event** ADF est la suivante :

$$\sigma_{kl} = s_k | -(e^{a_j}) \rightarrow s_l$$

Les lois et paramètres relatifs à chaque évènement peuvent alors être spécifiés dans le champ **extern** du langage ADF, selon la syntaxe suivante dans le cas de **l'event** précédemment créé :

Law (event e^{a_j}*)=loi(X)*

où "loi" et "X" sont respectivement la loi et son paramètre particulier définis dans le PD associé.

V.5.4.2 Les transitions à paramètres variables

Dans ce cas, la transition est fonction d'un ou plusieurs **flowin** ADF. Le nombre de **flowin** ADF étant identique au nombre de résultats possibles pour une équation de contrainte donnée, noté $\max(o_k^{n_j})$. Le résultat de cette équation est noté $o_k^{n_j}$ pour le diagramme d'état n_j. Ces transitions peuvent être définies de la manière suivante :

$$\forall\, t_{kl}^{n_j}\left(o_{k_i}^{n_j}\right) \in \mathcal{T}_{kl}^{n_j}, \qquad alors\ \exists\, e^{a_{j(i)}} \in \Sigma^{a_j}\ tel\ que\ e^{a_{j(i)}} = t_{kl}^{n_j}\_m\ \forall m \in [1, \max(o_{k_i}^{n_j})] \cap N$$

La **trans** créée par chaque **event** ADF ainsi créé est la suivante :

$$\sigma_{kl,m} = s_k \text{ and } (f_{k_i,m}) | -(e^{a_{j(i)}}) \rightarrow s_l$$

Les lois et paramètres relatifs à chaque évènement peuvent alors être spécifiés dans le champ **extern** du langage ADF, selon la syntaxe suivante dans le cas de **l'event** précédemment créé :

Law (event e^{a_j}*)=loi(X)*

où "loi" et "X" sont respectivement la loi et son paramètre particuliers définis dans le PD associé. Les instructions présentes dans le champ **extern** étant liées à un cas particulier, elles seront renseignées au moment de la simulation pour un système particulier, permettant ainsi de manipuler des **nodes** génériques.

En reprenant l'exemple du PD transformé Figure 48, le **flowin** du mécanisme de dégradation à 3 états peut prendre 4 valeurs différentes, égales au combinatoire des entrées de l'équation du PD (2 états * 2 états). La Figure 49 représente la correspondance entre les deux modèles à états.

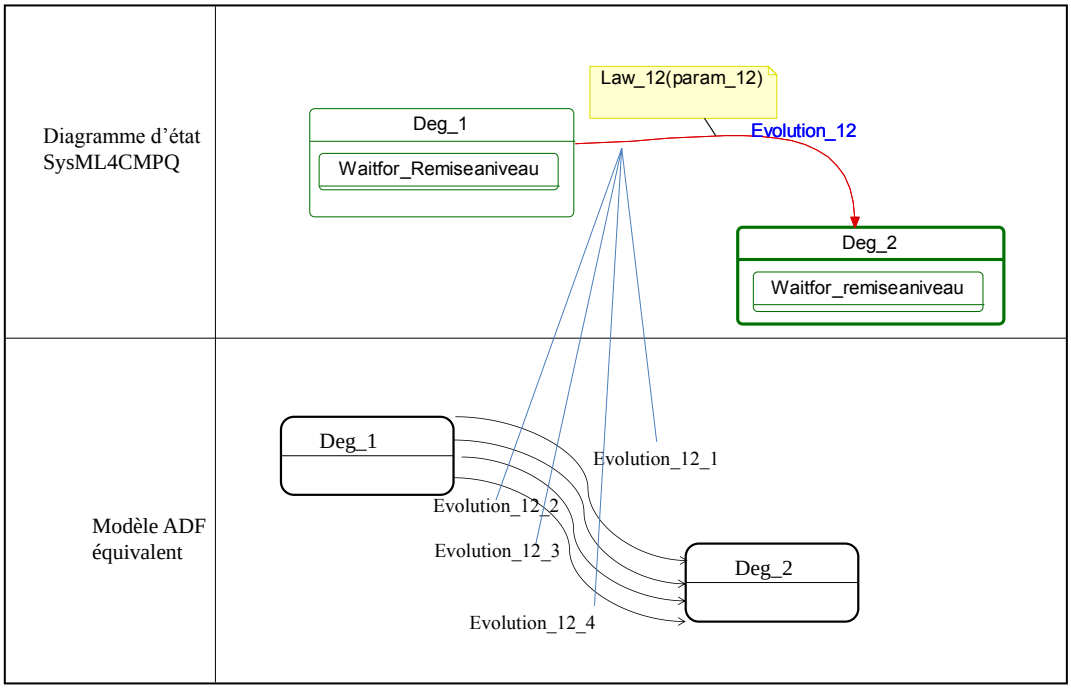


Figure 49 – Correspondances entre les transitions à paramètres variables

Afin d'implanter le code pour créer l'atome, il faut en outre renseigner dans le champ **extern** la valeur prise par l'équation du PD. La Figure 50 reprend le code partiel relatif au mécanisme de dégradation selon le PD pris en exemple.

```
node mecanisme degradation 3 etats
  flow
CE_eleve: bool: in;
CE_normal: bool: in;
CO_eleve: bool: in;
CO_normal: bool: in;
  state
Var : {Deg_1, Deg_1_fic, Deg_2 Deg_2_fic, Deg_3, Deg_3_fic };
  init
Var:=deg_1;
  event
Evolution_12_1, Evolution_12_2,  Evolution_12_3,  Evolution_12_4
  trans
(Var=Deg_1) and (CE_normal=1) and (CO_normal=1) |- Evolution_12_1-> Var:=Deg_2
(Var=Deg_1) and (CE_eleve=1) and (CO_normal=1) |- Evolution_12_2-> Var:=Deg_2
(Var=Deg_1) and (CE_normal=1) and (CO_eleve=1) |- Evolution_12_3-> Var:=Deg_2
(Var=Deg_1) and (CE_eleve=1) and (CO_eleve=1) |- Evolution_12_4-> Var:=Deg_2
```

Figure 50 – Code relatif au mécanisme de dégradation

V.5.4.3 Les transitions conditionnées

Les transitions conditionnées se composent d'une transition classique, pouvant être fonction ou non d'un **flowin**, et se modélise en langage ADF de la même manière que les transitions précédentes. Elle va aboutir à la création d'un état ADF fictif dont la création a déjà état modélisée, puisque les transitions composées (une loi exponentielle ne peut être composée avec un tirage aléatoire dans ce cas) ne peuvent être prises en compte sous ADF. De cet état fictif, un nombre **d'events** ADF équivalent au nombre d'états accessibles depuis la transition conditionnée doit être créé.

Ainsi, dans le cas d'une condition permettant d'atteindre 2 états :

$$\forall\ (c_{k,l}^{n_j}(p1), (c_{k,m}^{n_j}(p2))\ \in\ \mathcal{C}_{kl}^{a_j}, alors\ \exists\ e^{a_j}\ \in \Sigma^{a_j}\ tel\ que\ e^{a_j} = c_{k,l}^{n_j} et\ \exists\ e^{a_j}\ \in \Sigma^{a_j}\ tel\ que\ e^{a_j} = c_{k,m}^{n_j}$$

Les **trans** créées par ces deux **event** ADF ainsi créés sont les suivants :

$$\sigma_{k\prime l} = s_{k\prime}| - (e^{a_j}) \rightarrow\ s_l$$

$$\sigma_{k\prime m} = s_{k\prime}| - (e^{a_j}) \rightarrow\ s_m$$

Cette partie de la transition conditionnée créée deux **event** liés. En effet, il y a une probabilité d'atteindre soit l'état *l* (p1) soit l'état *m*.

Le champ **extern** d'ADF permet de déclarer ce type de loi, sous la forme suivante :

Law (event e^{a_j}*)=bucket(p1)*

Law (event e^{a_j}*)=bucket(p2)*

V.5.4.4 Les transitions forcées ou simultanées

Les transitions forcées constituent un point important dans la considération des aspects systèmes. Dans le modèle SysML4CMPQ, chaque transition forcée (permettant d'atteindre un sous état « Waitfor ») peut être déclenchée par différents diagrammes d'états envoyant un message indiquant le déclenchement de cette transition. En ADF, ces transitions devront être synchronisées lors de la modélisation du système particulier. Ainsi, il est nécessaire, pour chaque **node**, de définir un nombre élevé transitions forcées déclenchant cette transition. Par exemple, si deux politiques de maintenance sont appliquées au mécanisme de dégradation, alors deux transitions forcées relatifs aux remises à niveau doivent être créées. Ainsi :

$$\forall\ tf_{kl}^{n_j}\ \in\ \mathcal{T}f_{kl}^{n_j},\qquad alors\ \exists\ m\ e^{n_{j(v)}} \in \left((\Sigma^{n_j(v)})^m\right)\ tel\ que\ e^{n_{j(v)}}\_m\ = t_{kl}^{a_j}$$

Les **trans** créées par ces **event** ADF ainsi créées sont les suivantes :

$$\forall\ m, \sigma_{kl,m} = s_k| - (e^{n_{j(v)}}\_m) \rightarrow\ s_l$$

Le champ **extern** du langage n'a pas lieu d'être renseigné au niveau atomique. En effet, celui-ci se définie une fois après avoir défini les synchronisations entre ces transitions forcées et les messages les déclenchant.

V.5.5 Le champ assert

Afin de compléter les codes relatifs aux atomes de modélisation ADF, il est nécessaire de spécifier le champ **assert.** Ce champ permet d'exprimer les **flowout** en fonction de **flowin** et de l'état courant du **node** considéré. Dans notre modèle SysML4CMPQ, ce sont donc,,= dans le cas des *constraintblocks* informationnels, les états qui représentent ces **flowout** indépendamment de toute variable d'entrée. De la manière dont les **flow out** ont été définis, il existe $fout_k^{a_{j(v)}} = \boldsymbol{\sigma}(s_k)$

Ainsi, dans ce cas, ces équations se représentent simplement sous la forme :

$$\forall\, fout_k^{a_{j(v)}} \in Fout^{a_{j(v)}}, \qquad \delta: fout_k^{a_{j(v)}} = if\ (var = (s_k)\ then\ 1\ else\ 0)$$

En reprenant notre exemple du mécanisme de dégradation, c'est sur les **nodes** des conditions opérationnelles et environnementales (et donc sur l'objet impactant l'évolution du mécanisme de dégradation) que ces équations sur les **flowout** doivent être définies. Ainsi, le code ADF relatif à cette définition est présenté Figure 51.

```
node Condition Operationnelle
flow
   CO_eleve: out: bool,
   CO_normall; out: bool,
…
Assert
   CO_eleve= if (Var=eleve) then 1 else 0;
   CO_normal= if (Var=normal) then 1 else 0;
```

Figure 51 – Code relatif à une condition opérationnelle

Par ailleurs, comme précisé précédemment, il est nécessaire de définir des **flow out** fictifs (c'est-à-dire n'ayant ni sens physique ni sens informationnel. Un tel **flow out** n'est connecté à aucun **flow in**) afin de définir la fonction totale **δ**, lorsque le **node** dispose de **flow in**. Ces flow out n'ayant pas d'incidence dans le modèle, il suffit de définir de nouveaux **flow out** simplement équivalent aux états sur le **node** et ensuite d'introduire ces flow out pour définir **δ** :

$$\forall\, s_l^{a_{j(v)}} \in S^{a_{j(v)}}, \exists\, \forall fout_l^{a_{j(v)}} \in Fout^{a_{j(v)}}\ tel\ que\ \delta: s_l^{a_{j(v)}} = fout_l^{a_{j(v)}}$$

Dans le cas des *constraint* fonctionnelles, ce champ est équivalent aux équations F_k. Ainsi, soit $fout_m^{n_{j(v)}}$ un résultat d'une *constraint* fonctionnelle f_k^i, alors:

$$\forall\, f_k^i \in F_k, \qquad \exists\, fout_m^{n_{j(v)}} \in Fout^{n_{j(v)}}, fout_m^{n_{j(v)}} = f_k^i$$

Cet ensemble d'atomes de modélisation, assimilables à des composants sur étagère (COTS) est stocké en bibliothèque dans le logiciel SIMFIA. Ce point sera illustré lors l'application de la démarche à un cas d'étude.

L'ensemble des atomes de modélisation associés aux concepts de connaissance générique est maintenant formé. Toutefois, ces atomes ne prennent pas en compte les opérations d'ADF liées à la

maitrise de la complexité des systèmes (produit libre, connections et synchronisations). Il est en effet possible, par assemblage des **nodes** atomiques à l'aide des opérations ADF, de créer des **nodes** de plus haut niveau génériques, ou macro **nodes** génériques.

V.6. La modélisation d'un système particulier pour une CMPQ à partir des macro-nodes génériques

Les **nodes** correspondant aux atomes de modélisation ADF (« équivalents » aux concepts atomiques identifiés au chapitre 2) formant la bibliothèque d'atomes de modélisation génériques ont été créés. De multiples assemblages entre eux sont envisageables, et il est donc nécessaire de proposer une démarche de création systématique de ces macro-**nodes** à partir des **nodes** atomiques précédents et des opérations d'ADF (produit libre, connections et synchronisations).

V.6.1 Création des macro-nodes génériques

Les macro-**nodes** peuvent à présent être créés par assemblage des différents atomes de modélisation ADF, en utilisant les opérations du langage liées à la gestion de la complexité (produit libre, connections et synchronisations). Cette étape est grandement simplifiée si l'utilisateur de l'outil dispose de diagrammes SysML du système particulier. En effet, il est nécessaire de connaître les différents éléments du système et leurs interactions (par exemple le système est constitué de 100 composants, dont seules les défaillances sont modélisées…) avant de pouvoir former les macro-**nodes**. Les étapes de cette procédure de construction sont :

1. Identification des atomes de modélisation génériques à assembler

Afin d'identifier les **nodes** de chaque famille les plus adéquats, il faut identifier les relations tirées des PD ainsi que les différents objets d'intérêts dans le système à modéliser (nombre de composants, type de programme de maintenance appliqué…). Dans le contexte de la CMPQ, nous proposons par la suite une démarche d'identification séquentielle de ces atomes à assembler.

2. Création des atomes de modélisation génériques (par produit libre ADF)

Ces objets associés entre eux vont former un macro-**node** réutilisable. Ils peuvent aussi bien représenter les aspects dysfonctionnels d'un composant (selon le nombre de modes de défaillance….) que les aspects liés à l'organisation de la maintenance (opérateurs associés à un unique composant ou à un sous-système). Nous notons N_{ADF*} le nombre d'éléments présents dans un sous-système relatif à un macro-**node**. Il peut exister plusieurs atomes de modélisation du même type (par exemple plusieurs mécanismes de dégradation) dans chaque macro-**node**. Il est donc nécessaire de les identifier de manière différente.

Le code modélisant un tel macro-**node** est défini dans le champ sub de la manière suivante :

$$\forall \left(a_{j(v)}\right) \in (N_{ADF*}), k = 1, a_j(k) : a_j si\ a_i \neq a_j\ \forall\ a_i \in (N_{ADF*})$$

$$Si\ \exists\ a_i\ tel\ que\ a_j = a_i, alors\ a_j(k+1) : a_j$$

Le code Figure 52 est celui relatif à un sous-système appelé « subsystem1 » formé par deux mécanismes de dégradation, un symptôme, un mode de défaillance et le composant qu'ils impactent.

```
node subsystem 1
 sub
MecanismeDegradation1:MecanismeDegradation3E
MecanismeDegradation2:MecanismeDegradation3E
Symptome1:Symptome3E
ModeDefaillance1:ModeDefaillance
Composant1:Composant
```

Figure 52 – Création du code ADF relatif au champ sub

3. Relier les connecteurs des atomes de modélisation en interaction (connections ADF)

Les flux de sortie d'un atome sont les entrées d'autres atomes appartenant au même niveau de granularité. Par exemple, un flux de sortie peut être l'entrée de plusieurs objets différents. La condition pour deux objets d'être reliés est la suivante :

$$\forall\, j \in [1, N_{ADF*}] \cap \mathbb{N}, \forall\, i \in [1, Nfin^{a_{j(v)}}] \cap \mathbb{N}, \exists j' \in [1, N_{ADF*}] \cap \mathbb{N}, \exists\, i' \in [1, Nfin^{a_{j(v)}}]\ tel\ que\ Fin_i^{a_{j(v)}} = Fout_{i'}^{a_{j(v)}'}$$

Cette condition est respectée si les **flow** des deux **nodes** ont été construits de manière parallèle.

Elle entraîne la définition du code suivant, dans le champ **assert** du langage ADF de niveau sous-système :

$$a_{j(v)}.Fin_i^{a_j} = a_{j(v)'}Fout_{i'}^{a_{j(v)}'}$$

En reprenant l'exemple, le code créé par les connections entre les symptômes et le mode de défaillance est disponible Figure 53. A noter que les connecteurs ont été nommés de manière analogue aux *constraintparameter* du modèle SysML4CMPQ. Ici, la relation modélisée découle d'une équation de contrainte informationnelle.

```
node subsystem 1
 sub
symptome3E.E_1=Modedefaillance.param_occur
symptome3E.E_2=Modedefaillance.param_occur
symptome3E.E_3=Modedefaillance.param_occur
…
```

Figure 53 – Création du code ADF relatif au champ assert

4. Synchroniser les évènements (synchronisations ADF)

Les messages SysML envoyés ont été annotés tableau 2. Dans les diagrammes d'états, les messages envoyés déclenchent immédiatement une transition dans les diagrammes d'états cibles. Ainsi, en ADF, ces messages doivent être synchronisés ensemble.

De la manière dont ont été construits les $\sigma_{kl,m}$ (les transitions forcées), on peut en déduire l'existence d'une autre transition entrainant son forçage telle que :

$$\forall\ {\sigma_{kl,m}}^{a_{i(v)}}\ , \exists\ {\sigma_{k'l'}}^{a_{j(v)}}\ tel\ que \exists\ {\sigma_{k'l'}}^{a_{j(v)}}\ déclenche\ \ {\sigma_{kl,m}}^{a_{i(v)}}$$

Cette condition entraîne la définition du code suivant, dans le champ **sync** du langage ADF de niveau sous système.

$$<\ {\sigma_{kl,m}}^{a_{i(v)}}\ , {\sigma_{k'l'}}^{a_{j(v)}}\ >$$

Par rapport à l'exemple, la remise à niveau du mécanisme de dégradation s'effectue de manière forcée lorsqu'une action de maintenance est réalisée. Supposons donc que cette AM est réalisée suite à une politique corrective. La Figure 54 reprend le code relatif à cette synchronisation.

```
sync
   <politique_corrective.fin_AM, Mecanisme_degradation_3etats.remise_a_niveau>
```

Figure 54 – Code relatif à une synchronisation

A ce niveau, les macro-**nodes** génériques sont créés. La dernière étape concerne la procédure d'instanciation des **nodes** et des macro-**nodes**. Elle sera illustrée lors de la présentation du cas d'étude.

5. Instancier les transitions : utilisation du champ extern

Dans notre cas d'application, l'outil de simulation permettant la compilation du code généré est celui disponible sous SIMFIA. Le champ **extern** ne faisant pas partie du langage, les définitions proposées ne sont valables que pour cet outil de simulation. Les différentes transitions ont déjà été instanciées dans les atomes de modélisation. Etant donné les besoins liés à chaque étude de CMPQ, et notamment de changer les paramètres des transitions afin de comparer par exemple plusieurs programmes de maintenance, il est possible de directement modifier dans le code, conformément aux équations des PD, la valeur de ces transitions. Il faut simplement veiller à ce que les équations fassent intervenir les mêmes **nodes,** et non pas d'autres **nodes** de la même famille.

La démarche de création systématique des macro-**nodes** ADF a été présentée. Les macro-nodes génériques peuvent être eux-mêmes stockés en librairie, et être instanciés à d'autres systèmes particuliers. Il est à présent nécessaire de s'intéresser à la manière d'organiser l'utilisation de cette démarche afin de former des macro-**nodes** pertinents en regard de la connaissance relative à la CMPQ.

V.6.2 Modélisation d'un système particulier pour une CMPQ

La modélisation d'un système global pour une CMPQ a été organisée selon 4 axes : le SP, son contexte, l'activité de maintenance et son SS. Pour un utilisateur de l'outil support aux études de CMPQ (incluant les **nodes** atomiques en bibliothèque), il est nécessaire de se poser un certain nombre de questions lui permettant de modéliser au mieux son système d'intérêt. Ces questions sont guidées par le BDD du modèle SysML4CMPQ.

Comment modéliser le SP ?

Il est tout d'abord nécessaire de modéliser l'architecture fonctionnelle du SP. En effet, tout système dispose d'une fonction, supportée par au moins un composant physique. Des outils tels que les analyses SADT permettent de modéliser ces architectures. A partir de la fonction principale d'un système, celle-ci est décrite en sous-fonctions jusqu'à arriver au niveau des composants physiques supportant les fonctions de niveau élémentaire. Chacun de ces composants est soumis à des dégradations. Il est donc nécessaire pour l'utilisateur de spécifier le nombre de mécanismes de dégradation, de symptômes et de modes de défaillance modélisables. L'analyse du REX couplée à une étude AMDEC du système est essentielle pour déterminer ces relations. Ce REX doit en outre intégrer les informations liées à l'activité de maintenance et au contexte pour identifier les **nodes** atomiques adéquats à sélectionner. A partir de cet ensemble de **nodes** atomiques, le macro-**node** générique relatif au SP peut être modélisé par application de la démarche de création des macro-**nodes** génériques.

Comment modéliser son activité de maintenance ?

L'activité de maintenance d'un système est donnée par le programme de maintenance préventive que l'on souhaite lui appliquer. Un tel programme doit spécifier :

- les différentes politiques de maintenance préventives appliquées,
- les objets sur lesquels elles s'appliquent (composant, sous-système, système)

Etant donné que les interactions entre maintenance et SS sont purement dynamiques, il n'est pas nécessaire de considérer le SS pour identifier les **nodes** atomiques d'intérêt. Si aucun SS n'est modélisé, alors les transitions de sortie des états de préparation seront instantanées lors de leur instanciation.

Ces différentes politiques peuvent avoir à être synchronisées entre elles. Ainsi, en appliquant notre démarche de création des macro-**nodes** génériques à ces **nodes** atomiques, le macro-**node** relatif à l'activité de maintenance peut être créé.

Comment modéliser le SS ?

Afin de poursuivre sa modélisation, l'utilisateur de l'outil doit à présent s'intéresser aux ressources nécessaires à l'exécution du programme de maintenance. Ceci ce réalise par l'intermédiaire d'un REX adapté. Ces ressources peuvent avoir à être synchronisées et connectées (par l'intermédiaire du node « équipe de ressources ». Ainsi, le macro-**node** relatif au SS peut être créé.

Comment modéliser le contexte du SP ?

A partir d'un REX adapté spécifiant les différentes CO et CE appliquées à chaque composant et leur impact sur ceux-ci, l'utilisateur peut créer le macro-node relatif au contexte du SP par application de la démarche.

Comment modéliser le système global ?

Par simple assemblage des 4 macro **nodes** précédents en appliquant la démarche de création des macro-**nodes**, le macro-**node** relatif au système global est modélisable. En effet, par l'intermédiaire des interactions présentes dans les STM (synchronisations) et dans les PD (connections), la démarche peut s'effectuer de manière analogue sur des macro-**node** que sur des **nodes** atomiques.

Chaque macro-**node** peut être instancié indépendamment l'un de l'autre, permettant ainsi la réduction de l'effort de modélisation. Par exemple, pour comparer l'impact de deux programmes de maintenance sur le SP, seul le macro-**node** relatif à l'activité de maintenance doit être ré-instancié.

V.7. Vérification et validation du modèle ADF pour la CMPQ

En référence à la section II.7, il est nécessaire de valider et vérifier les atomes de modélisation ADF ainsi que le guide d'instanciation et d'assemblage de ces atomes. De la même manière que pour le modèle SysML, la validation doit se faire en concertation avec un expert dépositaire de la connaissance métier modélisée. Si les réunions menées en collaboration avec EDF ont conduit à la validation du modèle SysML, elles n'ont pas porté sur la validation du modèle ADF. Les atomes de modélisation ADF, de par leur codage, sont en effet difficilement validables par une simple procédure de lecture et de concertation avec les experts du domaine métier modélisé.

Pour vérifier le modèle ADF, celui-ci étant simulable, il est possible de créer un modèle analytique équivalent, puis de comparer les résultats obtenus par simulation avec les résultats théoriques. Nous avons vérifié un sous ensemble très réutilisé dans la CMPQ par un processus markovien de sauts équivalent. Ceci nécessite cependant d'effectuer certaines hypothèses, n'entrainant ainsi qu'une vérification partielle du modèle ADF. Une perspective de ce travail est d'étendre cette vérification à un modèle complet.

Considérons le processus markovien de sauts (cf. Figure 55) modélisant la dégradation d'un composant sous les hypothèses suivantes :

- Le composant dispose de 3 états de dégradation et d'un état défaillant accessible uniquement depuis l'état de dégradation le plus élevé,
- Seule une politique de maintenance corrective entrainant des réparations parfaites est appliquée au composant,
- Les différents taux (dégradation, défaillance, réparation) sont constants et distribués selon une loi exponentielle.

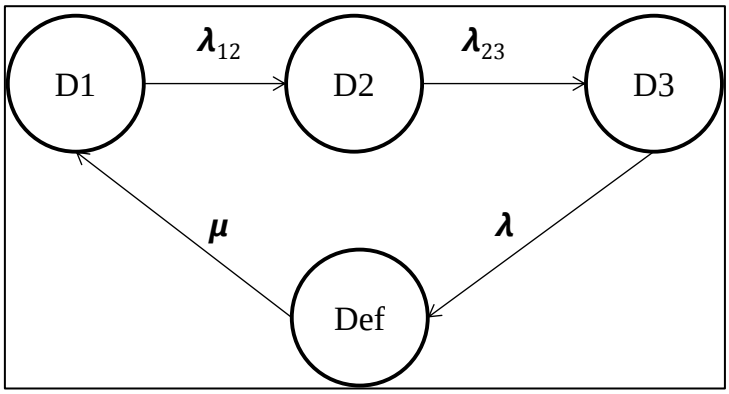


Figure 55 – Processus markovien de sauts équivalent au cas test de vérification

Le générateur infinitésimal de ce processus ergodique est donné par la matrice A.

$$A = \begin{pmatrix} -\lambda_{12} & \lambda_{12} & 0 & 0 \\ 0 & -\lambda_{23} & \lambda_{23} & 0 \\ 0 & 0 & -\lambda & \lambda \\ \mu & 0 & 0 & -\mu \end{pmatrix}$$

Le processus étant ergodique, il dispose par définition d'une loi stationnaire π telle que $\Sigma\pi = 1$. De plus, en appliquant l'équation de Chapman-Kolmogorov à ce processus, alors $\pi A = 0$. Ainsi, la loi stationnaire π de ce processus est solution du système linéaire:

$$\begin{Bmatrix} \pi A = 0 \\ \Sigma\pi = 1 \end{Bmatrix}$$

Dans notre cas, cette loi stationnaire $\pi = (\pi_{D1}, \pi_{D2}, \pi_{D3}, \pi_{Def})$ correspond au temps moyen passé dans chaque état, et ce temps moyen pourra être comparé aux résultats de la simulation. Ces temps servent de base significative au calcul de la plupart des KPIs de la CMPQ.

Le modèle ADF équivalent nécessite de considérer 3 atomes de modélisation différents : un mécanisme de dégradation à 3 états et 3 flow out un mode de défaillance à 3 flow in, et une politique de maintenance corrective, dont la transition « SC_fin_AM » est synchronisée avec la remise à niveau parfaite du mécanisme de dégradation et la disparition du mode de défaillance.

Le Tableau 28 montre les résultats obtenus analytiquement par la chaine de Markov et ceux obtenus par simulation du modèle ADF équivalent sous l'environnement SIMFIA, pour 4 jeux de taux donnés, et pour 10000 histoires sur une période de 72000h.

Tableau 28 – Résultats de la vérification partielle du modèle

Jeux de taux (h^{-1})				Processus markovien de sauts (résultat théorique)				Modèle ADF (résultat par simulation)			
λ_{12}	λ_{23}	λ	$\boldsymbol{\mu}$	π_{D1}	π_{D2}	π_{D3}	π_{Def}	π_{D1}	π_{D2}	π_{D3}	π_{Def}
0,01	0,01	0,01	0,1	0,3225	0,3225	0,3225	0,03225	03224	0,3221	0,323	0,03220
0,01	0,01	0,05	0,01	0,3125	0,3125	0,0625	0,3125	0,3119	0,3222	0,0628	0,3119

0,05	0,05	0,05	0,01	0,0125	0,125	0,125	0,625	0,0125	0,125	0,125	0,625
0,1	0,1	0,1	0,001	0,00971	0,00971	0,00971	0,9708	0,00982	0,00965	0,00973	0,9704

Les résultats trouvés sont très satisfaisant puisqu'ils présentent des variations de l'ordre de 10^{-3} entre les résultats théoriques et les résultats obtenus par simulation de notre modèle. Pour aller plus loin dans cette phase de vérification, il faudrait maintenant construire un modèle analytique plus important en terme de modélisation et de comparer avec le modèle ADF équivalent.

V.8. Conclusion

A partir des contraintes liées à la sémantique formelle du langage ADF, nous avons proposé dans ce chapitre un mapping depuis les éléments sémantiques du langage source, SysML, et plus particulièrement les éléments sémantiques de ce langage utilisés pour le modèle SysML4CMPQ, et le langage ADF. Ce mapping aboutit à la définition d'une bibliothèque d'objets atomiques ADF génériques, et à un guide d'assemblage de ces objets atomiques pour former des macro-**nodes** permettant de former *in fine* un modèle de niveau système. Le code ADF ainsi formé a été partiellement formellement vérifié. Sa validation nécessite son utilisation au sein d'EDF.

Ces différentes règles de correspondances permettent d'assurer une cohérence sémantique entre le modèle SysML4CMPQ proposé au chapitre 4, lui-même construit de manière fidèle à la connaissance relative à la CMPQ identifiée au chapitre 2, et le modèle ADF (cf. Figure 56). Cependant, ces règles doivent être adaptées au langage ADF. Celui-ci ne permet en effet que par des moyens détournés de prendre en compte des phénomènes tels que les transitions à paramètres variables. Ainsi, cela complique le mapping entre les langages en entraînant une perte de généricité importante dans les concepts de modélisation. Il est en effet parfois nécessaire de créer un nombre important de **nodes** ADF pour modéliser un concept unique dans le modèle SysML4CMPQ (selon les objets en interaction). D'autres phénomènes (systèmes bouclés) ne pouvant être pris en compte, alors il est nécessaire de voir au cas par cas si la modélisation d'un système est correcte. Si ce type de système est physiquement fréquent, il est assez rare dans le contexte de la CMPQ de pouvoir les modéliser, puisqu'ils apparaissent lorsqu'un REX très fourni est disponible (permettant par exemple de quantifier l'interdépendance de mécanismes de dégradation).

Ainsi, n'importe quel système sur lequel un décideur veut prendre une décision liée à la CMPQ est modélisable en ADF. Il est modélisable de deux manières :

- depuis un modèle SysML complet du système construit de manière cohérente avec le canevas donné par le modèle SysML4CMPQ. Dans ce cas, en implémentant une passerelle conformément aux règles proposées dans ce chapitre, le modèle ADF complet peut être créé automatiquement,
- directement en utilisant les objets atomiques ADF et le guide d'assemblage des macro-**nodes**.

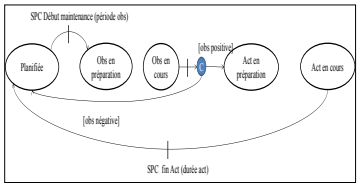
A partir de ce modèle, les règles d'évaluation des KPIs relatifs à la CMPQ définies au chapitre 2 sont quantifiables de manière stochastique grâce à la simulation de monte carlo réalisable par l'outil SIMFIA. Celui-ci permet en effet d'obtenir des résultats des temps d'attente dans chaque état du modèle et du nombre de transitions tirées. En termes de validation, celle effectuée dans ce chapitre

n'est que partielle. Le chapitre 6 présente un cas d'étude industriel concret permettant la validation plus globale le modèle final.

Chapitre 2: connaissance générique de la CMPQ

Vue interactionnelle

Vue comportementale des concepts



Transition	Concept en interaction	Transition forcée engendrée
Apparition mode défaillance	Maintenance corrective	Début maintenance (état « en attente » vers état « en préparation »)
	Composant	Panne (état « en marche » vers état « en panne »)
	Mode de défaillance	Occurrence forcée (état « non apparu » vers état apparu)
Début maintenance préventive (en général)	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »)
Fin AM active (quelle que soit la stratégie préventive)	Composant	Fin maintenance préventive (état « en maintenance préventive » vers état « en marche »
Fin AM active (stratégie corrective)	Composant	Fin panne (état « en panne» vers état « en marche »
Fin AM active (quelle que soit la stratégie)	Mécanisme de dégradation/ Symptôme	Remise à niveau ij (depuis un état de dégradation élevé i vers un état de dégradation équivalent ou moins élevé j)
Fin AM active (stratégie corrective)	Mode de défaillance	Disparition (état « apparu » vers état « non apparu »
Fin observation négative (quelle que soit la stratégie préventive)	Composant	Fin maintenance préventive (état « en maintenance préventive » vers état « en marche »
Début act (stratégie prévisionnelle)	Composant	Début maintenance préventive (état « en marche » vers état « en maintenance préventive »

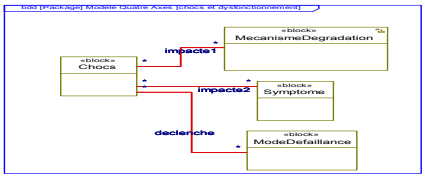
Vue statique

Concept	Caractéristiques	Comportement/ processus
Fonction	Flux entrants, flux sortants, fonction de transfert, loi et paramètre d'évolution	Continu/ discret : N états
Composant	Technologie, couts d'indisponibilité	Discret : N états (≥ 3 : en marche, en maintenance prev, en panne/ en maintenance corr)
Mode de défaillance	Maintenabilité, loi et paramètres d'occurrence	Discret : 2 états (apparu ou non)
Symptôme	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états
Mécanisme de dégradation	Maintenabilité, loi et paramètres d'évolution	Continu/ discret : N états

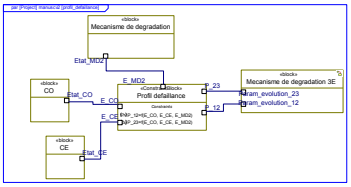
Démarche de construction du modèle SysML4CMPQ

Modèle SysMLde la connaissance générique de la CMPQ

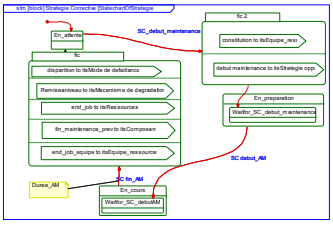
BDD



PD



STM



Mapping entre éléments sémantiques

Modèles atomique

```
node CorrectiveStrategy
 state
  Var : {Var___PLAN,Var___EN_PREP,Var___EN_COUR} ;
 event
  begin_mc,prep_mc,end_mc;
// Les valeurs du champs init sont données pour la mission
:Top mission   et la phase : Top phase
 init

  Var := Var___PLAN;
 trans
  { (Var = Var___EN_PREP)) |- begin_mc -> Var :=
Var___EN_COUR;
   ((Var = Var___PLAN)) |- prep_mc -> Var :=
Var___EN_PREP ;
   ((Var = Var___EN_COUR)) |- end_mc -> Var :=
Var___PLAN ;
 extern
  law <event begin_mc> = Dirac(0);
  law <event prep_mc> = Dirac(0);
  law <event end_mc> = exponential(0.1);
```

Chapitre 5: modèle ADF générique de la CMPQ

Guide d'assemblage et d'instanciation

Step 1:
Step 2:
Step 3:

Figure 56 – De la connaissance générique au modèle ADF

Chapitre VI. Application de la démarche à un cas d'étude réel : le système ARE

VI.1. Introduction

Pour déployer certaines étapes de validation du modèle générique ADF et pour montrer la faisabilité et l'intérêt de l'approche de quantification des programmes de maintenance proposée en termes d'utilisabilité et de crédibilité de l'outil, nous appliquons cette démarche à un système réel issu d'une installation de production d'énergie. L'objectif est, à partir du modèle ADF générique (issu de l'étape (1')) matérialisé par la librairie des atomes génériques de modélisation, de l'instancier à ce cas d'étude réel (pour réaliser l'étape (2)) afin d'en évaluer les KPIs par simulation stochastique (pour réaliser l'étape (3)).

L'étude considère, en se basant sur les données et informations réelles issues du retour d'expérience disponibles, un système de maintenance d'un robinet gros débit du système de régulation de débit d'eau alimentaire d'une centrale nucléaire, que nous appellerons ARE par la suite, particulièrement critique en termes de disponibilité de production. Dans ce chapitre, nous présentons le système ARE étudié, ses règles de fonctionnement et de dysfonctionnement, le comportement de ses différents matériels ainsi que les besoins du système ARE en regard de la CMPQ. Puis nous explicitons le mode d'obtention des données utiles (taux de dégradation, de défaillance…) issues du REX, et les hypothèses liées à ce mode d'obtention. Ces données sont cependant fictives bien que cohérentes puisqu'élaborées en étroite collaboration avec des experts d'EdF. Enfin, nous appliquons la démarche proposée en fonction des hypothèses faites sur les données. Les résultats obtenus sont validés par comparaison avec les résultats d'autres études menées sur le système ARE avec les mêmes données, et d'autres propositions de programmes de maintenance seront évaluées afin de mettre en valeur certaines valeurs ajoutées de l'approche telle que la réutilisabilité de la librairie ADF fournie.

VI.2. Objectifs de l'application

Les besoins d'EDF actuels sur le système ARE sont de pouvoir effectuer une étude de CMPQ afin d'évaluer les KPIs définis dans le **Tableau 15** pour aider à la décision concernant le choix des périodes d'observation des maintenances conditionnelles, sur une période de 10 ans.

Pour cela, des AMDEC ont été réalisées sur le système, et un REX a été analysé afin d'obtenir des taux de dégradation et de défaillance approximés.

VI.3. Présentation du système ARE

VI.3.1 Choix du cas d'application

Le système ARE est composé de plusieurs composants, présentant des interactions les uns avec les autres, dont le fonctionnement et le dysfonctionnement peuvent être évalués grâce aux AMDEC réalisées en amont de l'étude de CMPQ et au REX disponible. Une quarantaine de composants, fidèles aux concepts de connaissance identifiés et modélisés dans ce manuscrit, peuvent donc être identifiés sur la robinetterie gros débit et son contrôle-commande. Des modes de défaillance et des mécanismes de dégradation, également fidèles aux concepts de connaissance, sont associés à chacun d'entre eux est

associé à des concepts dysfonctionnels variables, et diverses politiques de maintenance supportées par un système de soutien leurs sont appliquées, permettant ainsi de considérer un panel conséquent des concepts de modélisation du chapitre 2. Ces politiques sont aujourd'hui appliquées selon une période donnée, que ce soit concernant les politiques basées sur les contrôles (observation approfondie de l'état d'un composant) ou les politiques basées sur les inspections (observation simple de l'état d'un composant). Le système est également soumis à de la surveillance en fonctionnement et à des rondes.

Le système ARE parait donc être parfaitement cohérent dans la connaissance générique identifiée au chapitre 2, puis transformée *in fine* en atomes de modélisation ADF. La connaissance disponible pour ARE est en effet incluse dans la connaissance générique, et des études fonctionnelles et dysfonctionnelles réalisées en amont de l'étude de CMPQ permettent d'obtenir suffisamment de données quantifiables pour l'évaluation des KPIs.

VI.3.2 Vision fonctionnelle du système ARE

Le système ARE permet une régulation de niveau des générateurs de vapeur. Il peut être considéré comme structuré en deux sous-systèmes : le premier est un robinet régulant un débit d'eau nécessaire au bon fonctionnement du générateur, et le second est une chaine de régulation surveillant les valeurs de ce débit. La fonction globale de ce système « réguler le niveau du générateur de vapeur » peut être vue comme un booléen. Les états intermédiaires de dégradation fonctionnelle ne seront donc pas considérés dans le modèle. Le bon fonctionnement de chaque composant est nécessaire pour accomplir cette fonction. Ainsi, les fonctions associées à chaque composant ne sont pas considérées, et le système global peut être vu sous la forme d'un assemblage de composants en série nécessaires pour que le système global remplisse sa fonction.

La Figure 57 représente une vue schématique de ces deux sous-ensembles.

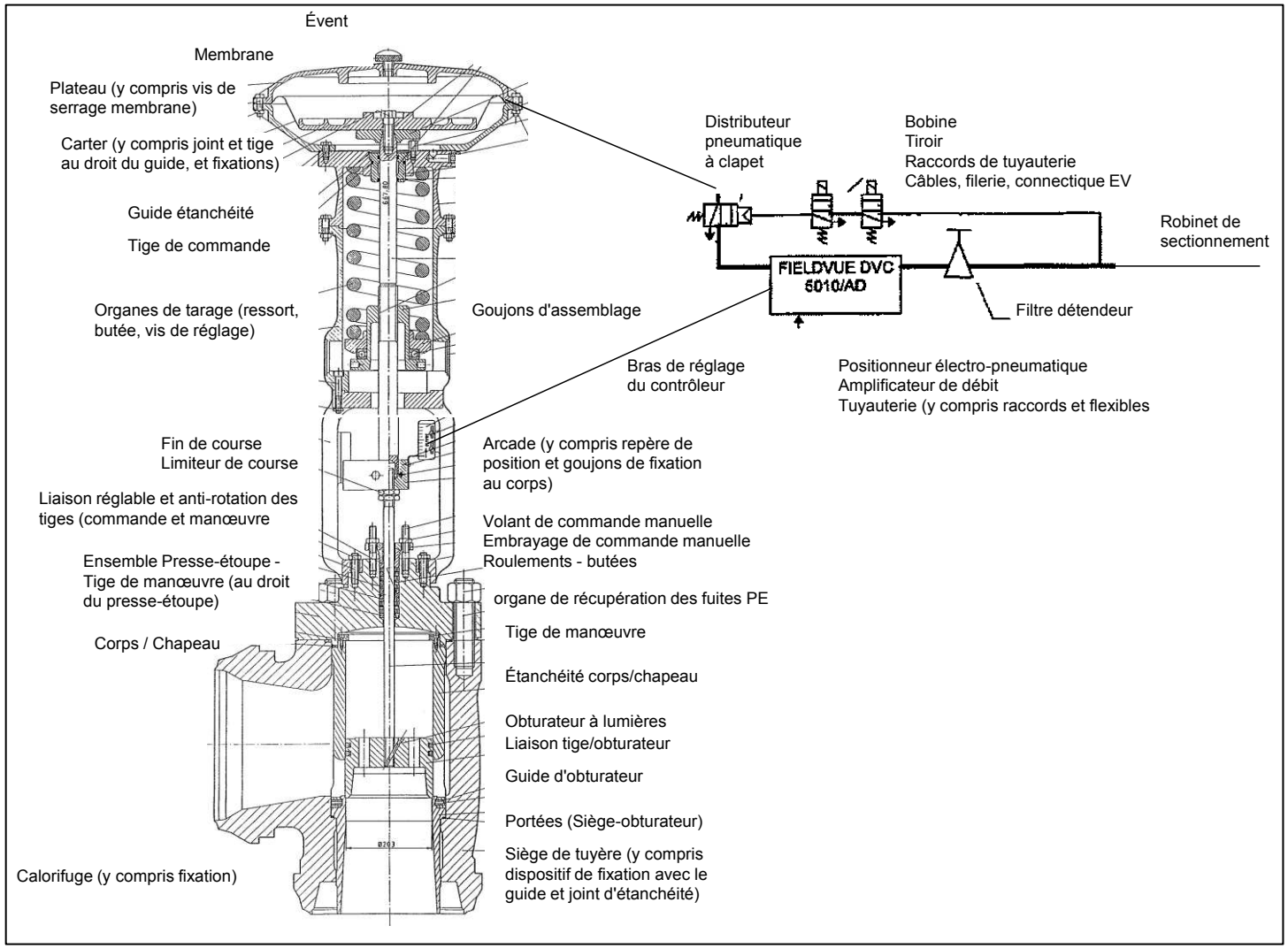


Figure 57 – Schéma du système ARE

VI.3.3 Vision dysfonctionnelle du système ARE

Le REX disponible pour le système ARE est matérialisé sous la forme d'une base de données dans laquelle les évènements indésirables ont été répertoriés lors des maintenances préventives et correctives sur une période de dix ans (2000-2010). Ces évènements, bien qu'existants, sont plutôt rares, voire très rares. Les taux de dégradation/défaillance élicités par la suite sont en effet de l'ordre de 10^{-7} par heure.

Par ailleurs, une AMDEC a été menée sur les différents composants du système ARE. Cette AMDEC a permis de mettre en évidence les principaux modes de défaillances, symptômes et mécanismes de dégradation associés à chacun des composants du système. Ceux-ci sont donc identifiés Tableau 29, avec leur mécanismes de dégradation associés. En comparaison avec la Figure 57, certains composants ne sont pas répertoriés dans ce tableau. Il s'agit des composants sur lesquels aucune défaillance/dégradation n'a été observée sur la période d'observation considérée.

Tableau 29 – Composants et mécanismes de dégradation

Composant/ sous ensemble	Mécanisme de dégradation
Presse étoupe	Perte caractéristiques mécaniques
Tige obturateur	Desserrage
Electrovanne 3voies	Brulure isolant
	Rupture par fatigue
Positionneur	Déconnexion
	Encrassement
Filtre détendeur	Bouchage
	Dérive
Fin de course	Défaut de réglage
Amplificateur de débit	Dérive
Tige carter	Usure par frottement
Distributeur	Usure
Raccords et flexibles	Inétanchéité par vieillissement
Tuyauterie	Desserrage
Membrane	Vieillissement
Guide d'étanchéité	Grippage

A chaque mécanisme de dégradation a été associé une ou plusieurs politiques de maintenance, faisant ainsi le lien entre l'AMDEC et le REX. Ces politiques permettent de définir le nombre d'états de dégradations associé à chaque mécanisme. Le Tableau 30 reprend les différents mécanismes de dégradations identifiés par l'AMDEC, et y associe les politiques de maintenance appliquées et le nombre d'états de dégradation en découlant. La règle est la suivante :

- si le mécanisme de dégradation dispose d'une politique préventive basée sur des contrôles alors il n'y a que deux niveaux de dégradation (« dégradé 1 », ou nominal, et « dégradé 2 » pour reprendre les notations de la Figure 15),
- si le mécanisme de dégradation dispose d'une politique préventive basée sur des inspections alors il n'y a également que deux niveaux de dégradation, identiques aux précédents,
- si le mécanisme de dégradation dispose à la fois d'une politique préventive basée sur des contrôles et d'une politique de maintenance basée sur des inspections, alors l'état « dégradé 3 » doit être modélisé. En effet, la prise en compte de ces deux types d'AM d'observations implique que certains états de dégradations peu avancés ne sont pas observables par de simples inspections.

Dans le tableau, la colonne « I » précise si des politiques de maintenance préventives basées sur des inspections sont appliquées sur le mécanisme de dégradation en relation, la colonne « C » précise si des politiques de maintenance préventive basées sur des contrôles sont appliquées sur le mécanisme de dégradation en relation, et la colonne « Net » précise le nombre d'états de chaque mécanisme de dégradation.

D'autres d'observations viennent en pratique compléter ces éléments : la surveillance continue en salle de commande et les rondes. Il est difficile de distinguer le rôle de la surveillance en salle de commande des autres inspections externes. En effet, celles-ci servent souvent à détecter des pannes des composants. Nous choisirons de ne pas les modéliser, en supposant que toute panne de composant est détectée immédiatement puisque la surveillance en salle de contrôle est continue.

Par ailleurs, les rondes servent à détecter des états de dégradation très peu avancés nécessitant une plus grande finesse du modèle (4 états de dégradations par exemple). Nous avons cependant décidé de les modéliser afin de montrer la capacité du modèle à les prendre en compte, mais en leur attribuant une valeur de période supérieure à 10 ans n'entraînant aucun impact sur la disponibilité du système.

Ainsi, seuls les contrôles et les inspections auront un impact significatif sur le fonctionnement du système.

Tableau 30 – Les mécanismes de dégradation du système ARE

Mécanisme de dégradation	Ronde	I	C	Net
Corrosion érosion du corps chapeau	Non	Oui	Oui	3
Perte caractéristiques mécaniques du presse étoupe	Oui	Oui	Non	2
Desserrage de la liaison tige/ obturateur	Non	Non	Oui	2

Brulure isolant câbles électrovannes 3 voies	Non	Oui	Non	2
Rupture par fatigue électrovannes 3 voies	Non	Non	Oui	2
Déconnexion du positionneur	Non	Non	Oui	2
Encrassement du positionneur	Non	Oui	Oui	3
Dérive du positionneur	Oui	Oui	Non	2
Bouchage filtre détendeur	Non	Non	Oui	2
Dérive détendeur	Non	Non	Oui	2
Défaut de réglage de fin de course	Non	Non	Oui	2
Dérive de l'amplificateur de débit	Non	Non	Oui	2
Usure par frottements tige-carter	Oui	Oui	Non	2
Usure distributeur pneumatique à clapet	Oui	Oui	Oui	3
Inétanchéité par vieillissement des raccords et flexibles	Oui	Oui	Oui	3
Desserrage tuyauteries électrovannes	Oui	Oui	Oui	3
Vieillissement de la membrane	Oui	Oui	Oui	3
Grippage du guide d'étanchéité	Oui	Oui	Oui	3

La rareté des évènements conduit forcément, dans le but d'évaluer des taux significatifs, à certaines hypothèses restrictives par rapport au modèle initial. Elles sont les suivantes :

- Indépendances des mécanismes de dégradation et des modes de défaillance,

Le REX disponible ne permet pas de quantifier les impacts entre les différents mécanismes de dégradation de manière claire. Ainsi, chaque taux de dégradation/défaillance est indépendant. Cette hypothèse est donc cohérente avec le fait qu'ADF ne permettent pas de modéliser des systèmes bouclés, et conduit également à l'indépendance des modes de défaillance. En effet, dans notre modèle, deux atomes de modélisation en interdépendance paramétrique forment un système bouclé.

- Chaque mécanisme de dégradation entraine l'occurrence d'un symptôme à son état de dégradation le plus avancé,

Par définition, une inspection ne permet de relever que les symptômes d'un mécanisme de dégradation. Ainsi, pour qu'une inspection sur un mécanisme de dégradation soit efficiente, celui-ci doit déclencher un symptôme. Afin de distinguer inspection été contrôles, ce symptôme n'apparait que lorsque le composant est dans un état avancé de dégradation (une inspection ne permet pas d'identifier de faibles niveaux de dégradation).

- Défaillance accessible uniquement depuis l'état de dégradation limite,

Les dégradations étant relativement lentes, cette hypothèse est cohérente et nettement simplificatrice pour l'obtention des taux.

Ainsi, selon ces hypothèses, le modèle standard de dégradation adopté est représenté Figure 58, dans le cas d'un mécanisme de dégradation à 3 états.

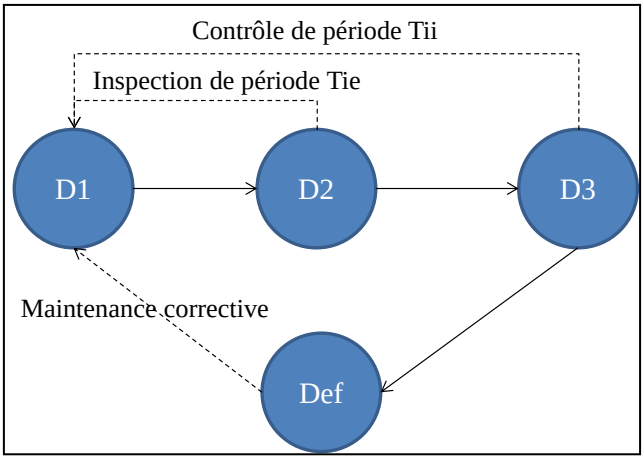


Figure 58 – Représentation du comportement d'un composant pour l'obtention de ses taux de défaillance

(Lair, 2012) met en place une méthodologie d'obtention des taux de dégradations selon ce modèle et le REX disponible. En accord avec le modèle de dégradation ci-dessus, les taux de défaillance et de dégradation ont pu être estimés. Le Tableau 31 présente ces valeurs, λ_{12} modélisant le taux de transition depuis l'état D1 vers l'état D2, λ_{23} modélisant le taux de transition depuis l'état D2 vers l'état D3, et λ modélisant le taux de transition depuis l'état de dégradation le plus avancé (D2 ou D3) vers l'état défaillant. Parfois, λ_{23} n'est pas défini à cause du manque de REX. Le modèle adopté n'a alors que 2 états de dégradation (D3 n'existe pas).

Tableau 31 – Taux de dégradations et de défaillances

Mécanismes de dégradation	λ_{12} (h^{-1})	λ_{23} (h^{-1}) (si besoin)	λ (h^{-1})
Perte caractéristiques mécaniques du presse-étoupe	1,15741E-06		4,28196E-05
Desserrage de la liaison tige/ obturateur	1,65344E-07		4,74938E-06
Brûlure isolant câbles électrovannes 3 voies	3,30688E-07		8,32148E-04
Rupture par fatigue électrovannes 3 voies	1,65344E-07		2,77383E-04
Déconnexion du positionneur	1,65344E-07		2,77383E-04

Encrassement positionneur	1,98E-06	1,65904E-04	8,19444E-04
Dérive positionneur	1,15741E-06		5,18163E-05
Inétanchéité air du positionneur	3,30688E-07		1,92541E-04
Bouchage Filtre détendeur	1,65344E-07		4,74938E-06
Dérive détendeur	1,15741E-06		2,77383E-04
défaut de réglage du fin de course	9,92063E-07		4,97711E-04
Dérive amplificateur de débit	3,30688E-07		2,77383E-04
Usure par frottement tige- Carter	9,92063E-07		5,06449E-05
Usure distributeur pneumatique à clapet	9,92063E-07	6,41803E-05	3,0517E-04
Inétanchéité par vieillissement des raccords et flexibles	8,2672E-07	2,77383E-04	8,32148E-04
Desserrage tuyauteries électrovanne	6,61376E-07	2,77383E-04	1,92541E-04
vieillissement de la Membrane	1,32275E-06	4,35189E-05	3,0517E-04
Grippage du guide d'étanchéité	3,30688E-07	2,77383E-04	8,32148E-04

VI.3.4 Activité de maintenance du système ARE

Les différentes politiques de maintenance appliquées au système et ses composants sont elles aussi sujettes à des données. Elles proviennent d'un entretien avec un expert EDF ayant la connaissance à la fois des couts et des durées des différentes tâches. A des fins de confidentialité, les données présentées par la suite sont non réelles, mais cohérentes et proportionnelles avec les taux de dégradation/défaillance précédents et les données réelles de maintenance. Ces données sont constantes, mais ne tiennent pas compte de l'éventuel délai lié à une indisponibilité des ressources du SS, ce qui est modélisé par ailleurs dans notre modèle pour la CMPQ.

Concernant les tâches d'observation préventives, les informations requises sont la périodicité de la tâche, les risques de fausse alarme et de non détection, le cout et la durée éventuelle.

Le Tableau 32 présente les données utilisées pour les simulations.

Tableau 32 – Données liées aux actions de maintenance d'observations

Tâche d'observation préventive	périodicité	Risque de non détection	Risque de fausse alarme	Cout (€)	Durée (h)
Surveillance en salle de contrôle	Continue	1%	0%	0	0
Ronde	1 mois	10%	0%	0	0
Inspection	1 an	2%	0%	200	1
Contrôle	3 ans	3%	5%	1000	5

Concernant les tâches de maintenance actives liées à des politiques préventives ou correctives, outre les informations sur le cout et la durée entraînées par la tâche, l'information importante est l'efficacité de la tâche, que nous avons nommée dans notre modèle générique la qualité de l'action de maintenance. Le Tableau 33 présente ces données utilisées pour notre simulation.

Tableau 33 – Données liées aux actions de maintenance actives

Tâche de maintenance active	Efficacité	Cout (€)	Durée (h)
Remplacement du robinet	100%	30000	10
Remplacement positionneur	100%	1000	1
Remplacement filtre détendeur	100%	500	1
Remplacement du fin de course	100%	400	1
Remplacement amplificateur de débit	100%	500	1
Remplacement distributeur	100%	400	1
Remplacement tige et carter	100%	2000	3
Remplacement raccords et flexibles	100%	100	1
Resserrage des tuyauteries	90%	50	0,5
Remplacement de la membrane	100%	300	3
Remplacement du guide et de la membrane	100%	400	3
Remplacement des câbles électrovannes	100%	100	1
Remplacement du presse étoupe	100%	100	2

Remplacement de la tige et obturateur	100%	1500	3

Chaque action de maintenance active est déclenchée soit par des actions de maintenance (politique de maintenance préventive conditionnelle ou prévisionnelle) ou des défaillances observées sur les composants associés (politique de maintenance corrective).

Par ailleurs, l'étude du REX n'a pas permis de déterminer des données chiffrées concernant deux axes de modélisations définis au chapitre 2 : le contexte et le système de soutien. Ceci appuie l'amplitude du domaine de modélisation identifié. De même, ce REX ne permet pas de quantifier des interdépendances entre les atomes de modélisation. Ainsi, le modèle du système ARE n'est pas un système bouclé, et il peut donc se modéliser dans le langage ADF.

VI.4. Construction du modèle du système ARE

La méthodologie de construction des modèles pour la CMPQ proposée envisage deux possibilités pour réaliser l'étape (2) de la démarche et ainsi instancier le modèle générique à un cas particulier : soit les diagrammes SysML du système ARE sont disponibles, auquel cas la création du modèle ADF se réalise de manière automatique et transparente pour l'utilisateur ; soit ces diagrammes ne sont pas disponibles, et l'utilisateur doit spécifier les équations de contrainte de manière cohérente avec la forme générale de compatibilité avec le langage ADF définie au chapitre 5. Cette spécification permet d'identifier les atomes de modélisation ADF adéquats, conformément à la démarche proposée dans le chapitre précédent.

Dans notre cas, les diagrammes SysML du système ARE ne sont pas disponibles. Par conséquent, l'utilisateur ne travaille que dans un environnement permettant d'effectuer des simulations stochastiques à partir du langage ADF. Nous allons présenter la construction du système ARE global à partir de ses modèles atomiques, conformément à la section VI.6.

VI.4.1 Modélisation du SP

La fonction principale du système ARE est de « réguler le niveau du générateur de vapeur ». Le bon fonctionnement de l'ensemble des composants du système permet de réaliser cette fonction. Ces différents composants peuvent donc être vus comme un système série, fournissant la fonction de structure de système global. Bien que les fonctions de niveau composant ne soient pas identifiées, nous leur attribuons une fonction fictive correspondant à leur disponibilité dépendante d'un composant amont. La fonction de structure du système est alors modélisée puisque si un composant tombe en panne, la fonction de niveau système ne sera plus remplie.

D'un point de vue dysfonctionnel, les tableaux Tableau 31 à 33 fournissent les résultats d'une AMDEC couplée à une analyse du REX maintenance. Ainsi, à chaque fonction de niveau composant peut être associée un composant physique et ses éléments dysfonctionnels adéquats (mécanisme de dégradation et modes de défaillance).

Ainsi, nous allons appliquer la démarche de création des macro-**nodes** dans le cas de l'exemple de la membrane.

Pour identifier les atomes de modélisation à sélectionner**,** il est tout d'abord nécessaire, à partir des relations présentes dans les Tableaux 30 à 33, de définir l'ensemble de ces relations. Par exemple, dans le cas de la modélisation des $\boldsymbol{\lambda}$ (entrainant la défaillance du composant), le PD générique « profil de défaillance » doit être utilisé. Celui-ci tient compte des conditions environnementales et opérationnelles, les symptômes et les mécanismes de dégradation dans cette interaction. Dans notre cas, seule l'information relative à l'impact de l'état du mécanisme de dégradation sur le composant est disponible. Ainsi, dans le cas par exemple de la membrane, les équations de contrainte f_i, issues du PD « profil de défaillance » permettant de modéliser l'impact de l'état du mécanisme de dégradation (E_Mdeg) sur le taux de défaillance (P_O) sont les suivantes :

$$f1: P\_0{,}1 = if\{E\_Mdeg = \text{deg\_1}\ \ then\ 10^{-100}\}$$

$$f2: P\_0{,}2 = if\{E\_Mdeg = \text{deg\_2}\ \ then\ 10^{-100}\}$$

$$f3: P\_0{,}3 = if\{E\_Mdeg = \text{deg\_3}\ \ then\ 2{,}197\}$$

Ces équations, dont la forme est cohérente avec le langage ADF, modélisent le fait que l'état défaillant n'est accessible que lorsque le mécanisme de dégradation associé (le vieillissement de la membrane) est dans son état de dégradation le plus avancé, selon un taux constant de 3,0517E-04 h^{-1}.

Ainsi, le **node** mécanisme de dégradation doit disposer d'au moins 3 **flow out** pour pouvoir modéliser cette interaction. En procédant de la même manière avec l'ensemble des interactions paramétriques relatives au modèle particulier, les **nodes** adéquats des familles de **node** associées peuvent être identifiés. Dans notre cas, l'utilisateur devra sélectionner le **node** « mécanisme de dégradation 3 états 3 f**low out** », puisque celui-ci n'est impacté par aucun autre atome de modélisation.

Parmi l'ensemble des atomes de modélisation disponibles, le cas d'étude doit se construire par conséquent à partir des atomes suivants:

- Un mécanisme de dégradation à trois **états** et sans **flow in**,
- Un mode de défaillance à trois **flow in**,
- Une fonction (la fonction est fictive et permet de modéliser la disponibilité de la membrane) à deux **états** (« fonction assurée », « fonction non assurée »),
- Un composant à trois états (« en panne », « en maintenance préventive », « en marche »).

Sous l'environnement SIMFIA, utilisé pour la compilation du code ADF pour la simulation stochastique, l'utilisateur devra donc sélectionner ces modèles parmi ceux proposés par la bibliothèque « atomes_de_modélisation » présentée Figure 59.

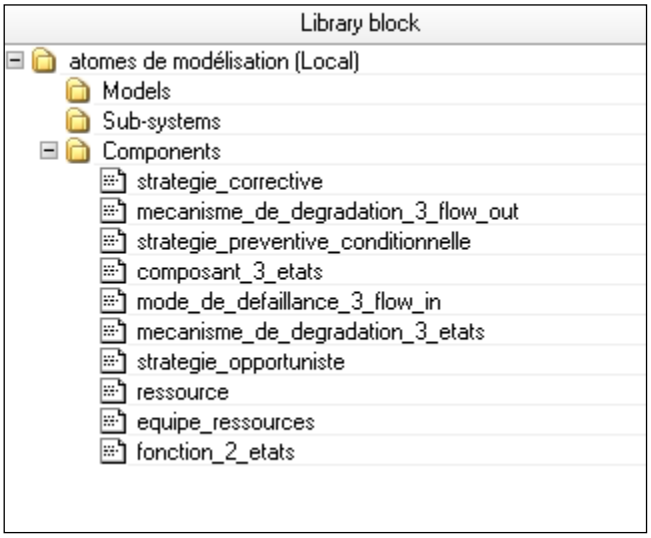


Figure 59 – Librairie « atomes_de_modélisation » sous SIMFIA

L'étape suivante de la démarche de création des macro-**nodes** réside dans la connection des différents atomes de modélisation préalablement sélectionnés. Ces relations sont données par les équations de contraintes. Ainsi, dans le cas de la membrane, seuls les connecteurs des mécanismes de dégradation et du mode de défaillance doivent être reliés.

A ce stade de la démarche, il faut maintenant s'intéresser aux synchronisations. En effet elles sont définies dans les différents diagrammes d'états du modèle SysML4CMPQ par l'intermédiaire des *messages envoyés*. Lorsqu'un message n'a pas d'atome de modélisation associé, alors la synchronisation n'est pas définie. Ainsi, dans l'exemple de la membrane, la seule synchronisation créée est :

- le passage vers l'état en panne du composant et la perte de la fonction,

Les autres synchronisations sont liées à l'activité de maintenance. Sans celle-ci, le système n'est en effet pas réparable.

A ce niveau, le code relatif à chaque composant indépendant du Tableau 29 est créé et complet. Ces macro-**nodes** génériques doivent à présent eux-mêmes être assemblés pour former le système principal global. Pour ce faire, il faut réappliquer la démarche de création de macro-**nodes**. Cette fois-ci, elle consiste simplement en la connection des différentes fonctions de chaque macro-**node** de composant de sorte à former la fonction de structure du système. La Figure 60 reprend la structure du code global relatif au macro-node générique correspondant au système principal d'ARE, appelé « ARE_SP ». Etant donné l'absence de vue graphique du langage ADF, la Figure 61 reprend ce code d'un point de vue « boite noire ». Cette vision informelle est réalisée sous l'interface SIMFIA. Elle ne permet pas d'illustrer les synchronisations du code, mais elle donne une vision alternative et intuitive des produits libres et des connections entre les macro-**nodes** entrant en jeu dans la modélisation du SP. Les connecteurs libres sont issus des modèles atomiques de mécanismes de dégradation des macro-nodes de composants. Ils sont destinés à être connectés avec les politiques de maintenance préventives lors de la création du macro-node de niveau supérieur (englobant SP et activité de maintenance).

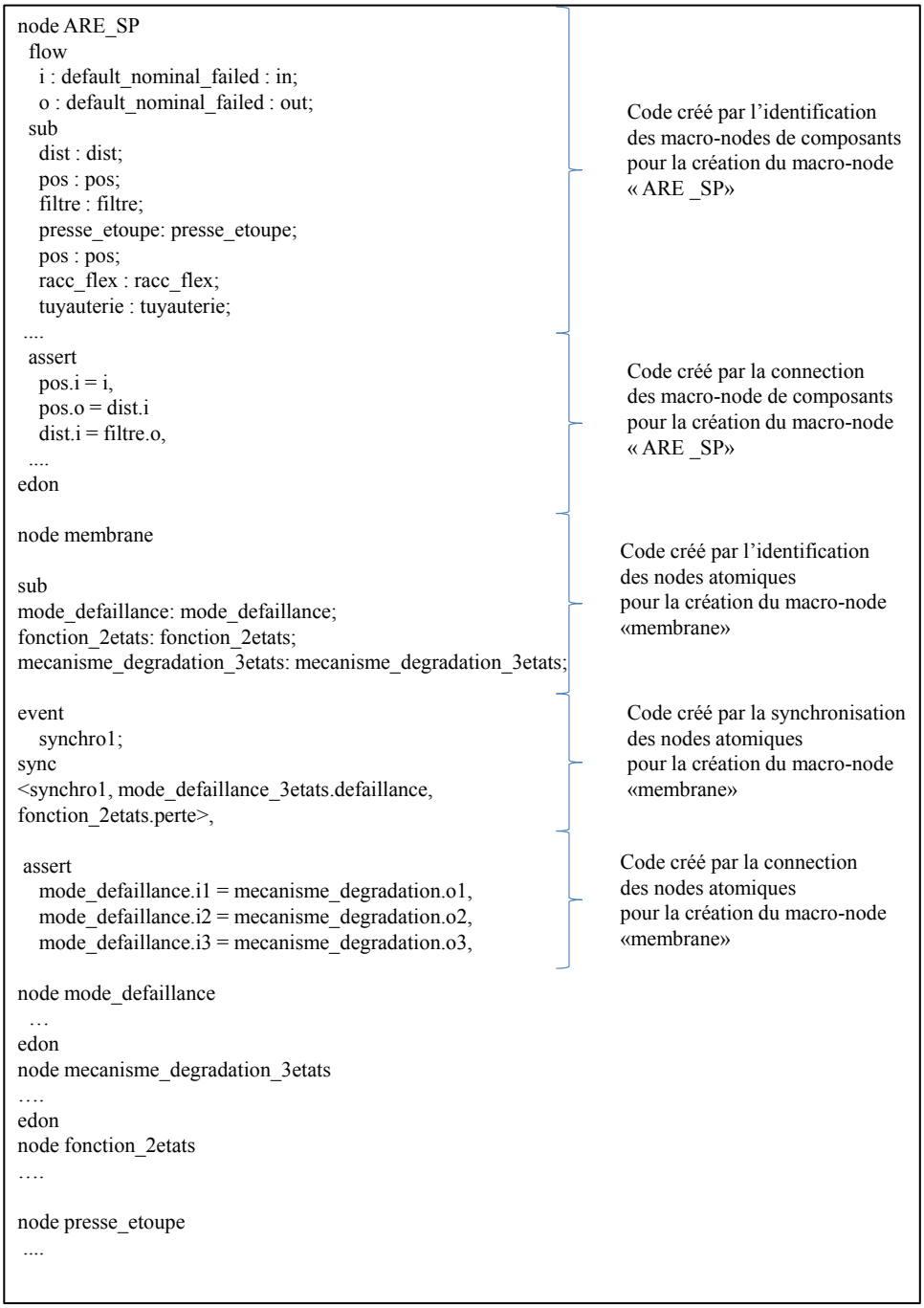


Figure 60 – Structure du macro-node du SP du système ARE

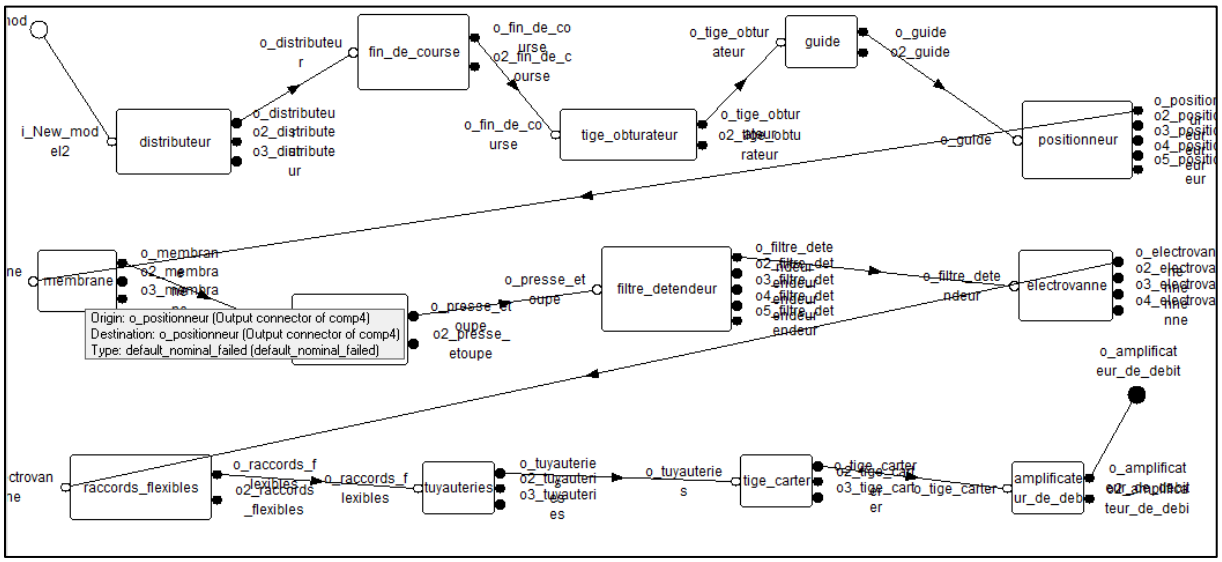


Figure 61 – Vision « boite noire » du SP du système ARE

VI.4.2 Modélisation de l'activité de maintenance

Dans notre exemple, 2 ou 3 politiques de maintenance préventive sont appliquées à chaque composant (cf. Tableau 32). De plus, chacun d'entre eux dispose d'une politique de maintenance corrective. Il faut donc sélectionner ces politiques de maintenance en librairie autant de fois qu'elles ont de composants en interaction.

En termes de connections, ces politiques ne sont connectés qu'aux éléments du SP en interaction. Ainsi, dans le macro-**node** relatif à l'activité de maintenance du système ARE, aucune connection n'est créée. En revanche, des synchronisations existent entre les différentes politiques de maintenance préventives, puisque celles-ci sont exécutées de manière simultanée sur chaque composant. La Figure 62 reprend la structure du code global relatif au macro-**node** générique correspondant au système principal d'ARE, appelé « ARE_SP ». La Figure 63 reprend partiellement ce code d'un point de vue « boite noire ».

```
node  ARE_AM
sub
   preventive_controle_membrane: preventive_controle
   preventive_insp_membrane: preventive_insp
   corrective_membrane: corrective
   preventive_controle_amplificateur: preventive_controle
   corrective_amplificateur: corrective
   …

event
   synchro_controle
sync
   <synchro_controle, preventive_controle_membrane.debut
_obs, preventive_controle_amplificateur.debut_obs,….>

node preventive_controle_membrane
 …
edon

node preventive_insp_membrane
 …
edon

node preventive_insp_membrane
 …
edon
```

Code créé par l'identification des nodes atomiques pour la création du macro-node « ARE _AM»

Code créé par la synchronisation des nodes atomiques pour la création du macro-node «ARE_AM»

Figure 62 – Structure du macro-node de l'activité de maintenance du système ARE

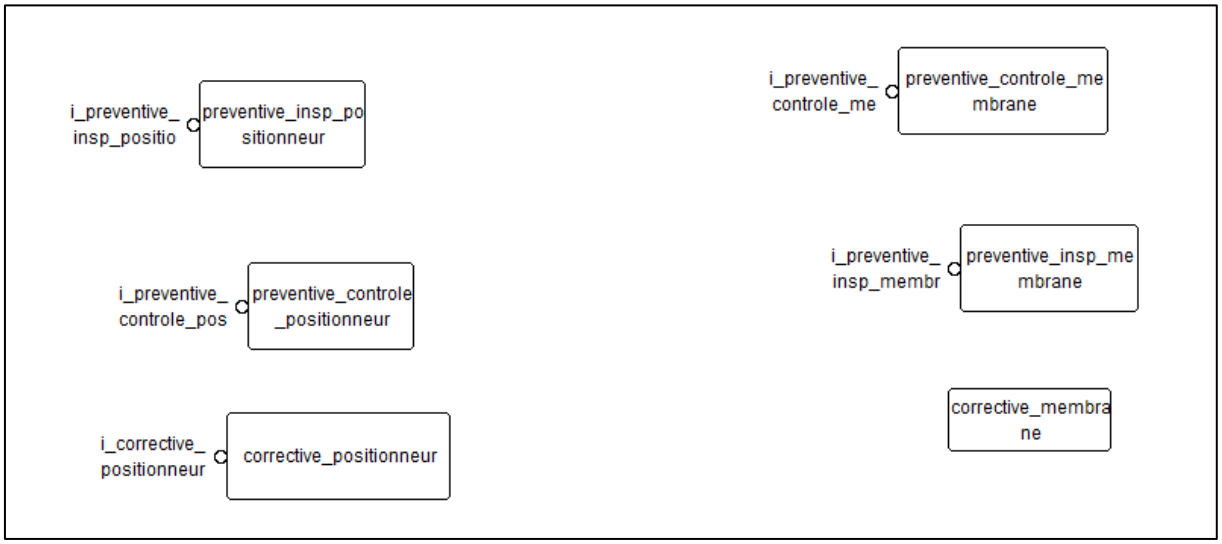


Figure 63 – Vision « boite noire » partielle de l'activité de maintenance du système ARE

Le macro-**node** générique relatif à l'activité de maintenance du système ARE est à présent disponible et stockable en libraire. Dans ce cas d'étude, aucun contexte ni système de soutien n'est considéré, et donc le macro-**node** relatif au cas d'étude complet (activité de maintenance et SP) peut être créé à partir de ces deux macro-**nodes**.

VI.4.3 Modélisation du système global

Les macro-**nodes** d'intérêt étant identifiés, il est nécessaire de les connecter et de les synchroniser pour former le cas d'étude complet relatif au système ARE et à son activité de maintenance.

Conformément aux équations de contrainte liées aux observations, les connections doivent être réalisées entre chaque mécanisme de dégradation et sa ou ses politiques de maintenance préventives associées. Ceci modélise d'une part l'identification par l'observation de la politique du niveau de dégradation de chaque composant, et donc d'autre part le déclenchement ou non d'une AM active.

Conformément aux STM, les synchronisations à établir entre les deux macro-**nodes** sont les suivantes :

- la préparation des politiques préventives avec le passage du composant vers l'état « en maintenance préventive »,
- le retour vers l'état « planifié » des politiques préventives avec le passage du composant depuis l'état « en maintenance préventive » vers l'état « en marche »,
- la préparation des politiques correctives avec le passage du composant vers l'état « en panne »,
- la fin des politiques correctives avec le le passage du composant depuis l'état « en panne » vers l'état « en marche » et la disparition du mode de défaillance,
- la fin d'une AM active avec une remise à niveau du mécanisme de dégradation associé.

A ce niveau, le modèle global simulable relatif au système ARE et aux données présentées dans les tableaux 29 à 33 est disponible. Le code de niveau système est particulièrement intéressant de par sa modularité lorsque l'utilisateur de l'outil fait varier les hypothèses sur le cas d'étude, et par exemple la périodicité des contrôles. La structure de ce code est reprise Figure 64.Etant donné l'importance du nombre de connections à effectuer et le nombre de **nodes** en interaction, la vision boite est ici inutile.

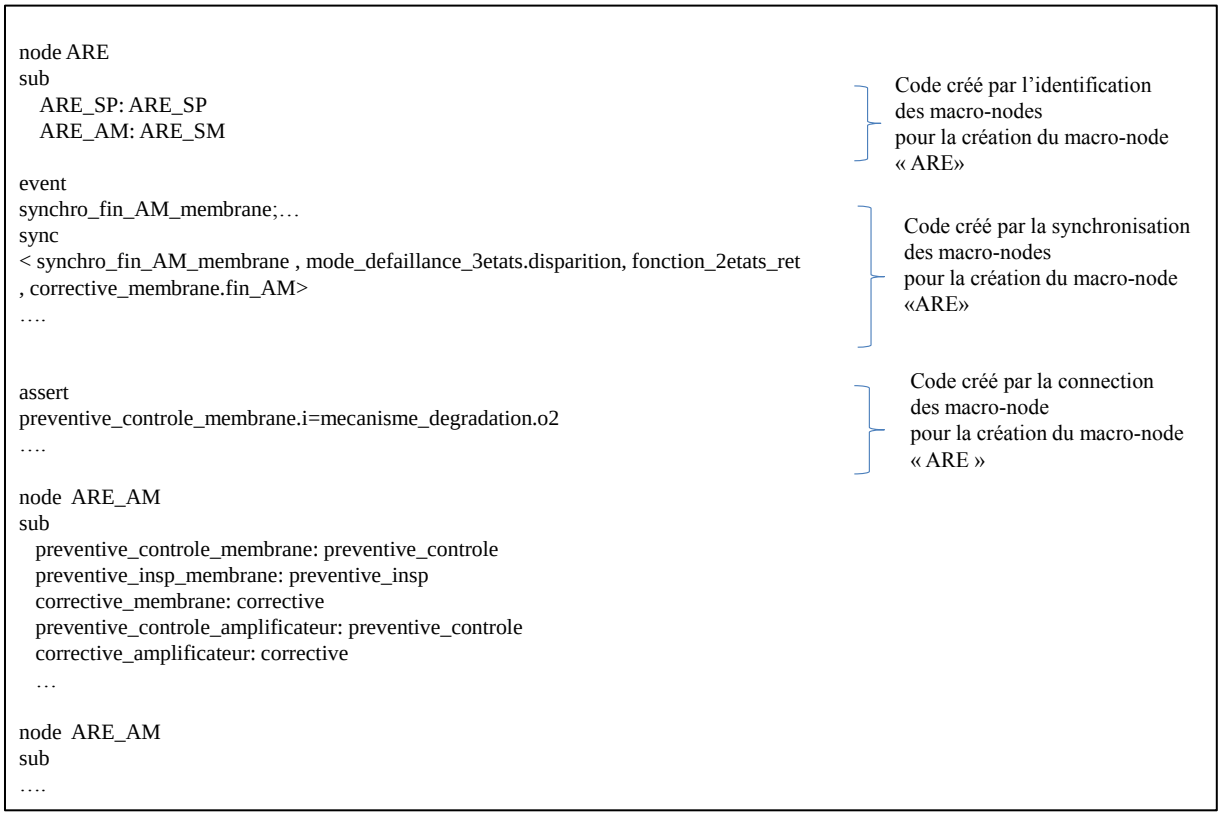


Figure 64 – Structure du macro-node du système ARE

Nous allons maintenant nous intéresser à l'étape de benchmarking sur le modèle obtenu permettant une validation de celui-ci en évaluant les différents KPIs de manière conforme aux valeurs du Tableau 31. Pour ces simulations, seul le macro-**node** de l'activité de maintenance aura à être ré-instancié, puisque les données sur le SP restent constantes.

VI.5. Résultats des expérimentations

Le modèle générique ayant été construit, il peut donc maintenant être instancié pour réaliser des simulations permettant le calcul des KPIs afin de montrer la faisabilité de l'approche, et ensuite d'en déduire ses valeurs ajoutées éventuelles. L'instanciation du modèle est expliquée section VI.6.1, et se fait conformément aux équations de contraintes spécifiées selon les valeurs des tableaux 29 à 33.

Ces valeurs ajoutées peuvent être relatives au contenu des résultats et à leur capacité à fournir des éléments prépondérants pour aider à la prise de décision en CMPQ (objet initial du travail) mais aussi être relatives à l'utilisabilité de la librairie ADF, ou encore à la réduction de l'effort de modélisation pour un modélisateur spécialiste du métier de la maintenance et non de l'outil de calcul utilisé (prérequis liées à l'utilisabilité de l'outil).

Pour répondre à une phase initiale de vérification du modèle, il est possible d'utiliser le benchmark d'EDF sur ARE à partir du jeu de données présenté précédemment devant permettre d'établir les meilleures périodicités pour les maintenances préventives. Ce benchmark a été construit par un tiers sur les mêmes hypothèses bien évidemment que celles que nous avons prises. Ce benchmark est réalisé avec les résultats obtenus par (Zille, 2009) en sachant que les simulations réalisées ont été faites

avec les RdP. Les simulations de ces réseaux ont conduit à évaluer des KPis liés à la disponibilité du système et au cout, pour différents programmes de maintenance. Le benchmark doit permettre ainsi très clairement de crédibiliser notre démarche en regard de la comparaison des résultats qu'il fournit par rapport aux résultats que nous obtenons avec notre simulation.

Dans un second temps, les évènements liées aux dégradations/défaillance étant très rares, nous avons associé des données académiques à chacun de ces évènements afin d'impacter la rareté des occurrences. En effet, lors d'une simulation sur 10 ans, le système tombe très rarement en panne.

Pour répondre au second type de prérequis (liées à l'utilisabilité de l'outil), nous étudierons donc également la prise en compte de politiques de maintenance prévisionnelle, ou autrement dit des délais sur les actions de maintenance non prévisionnelles afin de leur associer une pénalité réaliste, augmentant ainsi le retour sur l'indisponibilité fortuite. Par la suite, la prise en compte d'un système de soutien impliquant d'éventuels délais sur les AM sera étudié, et enfin la mise en place d'une politique de maintenance opportuniste

VI.5.1 Détermination de la périodicité adéquate des politiques de maintenance et validation de la démarche par comparaison avec le benchmark

Les différents tableaux de valeurs économiques présentés précédemment nous fournissent les données nécessaires au post traitement, puisque le simulateur de Monte Carlo SIMFIA ne permet d'obtenir que les temps de passage dans chaque état et le nombre de transitions tirées. Ces résultats s'avèrent cependant nécessaires à ce post -traitement destiné à l'obtention des KPIs relatifs à la CMPQ et notamment des couts des programmes de maintenance (cf. **Tableau 15**) comme illustré par la Figure 65.

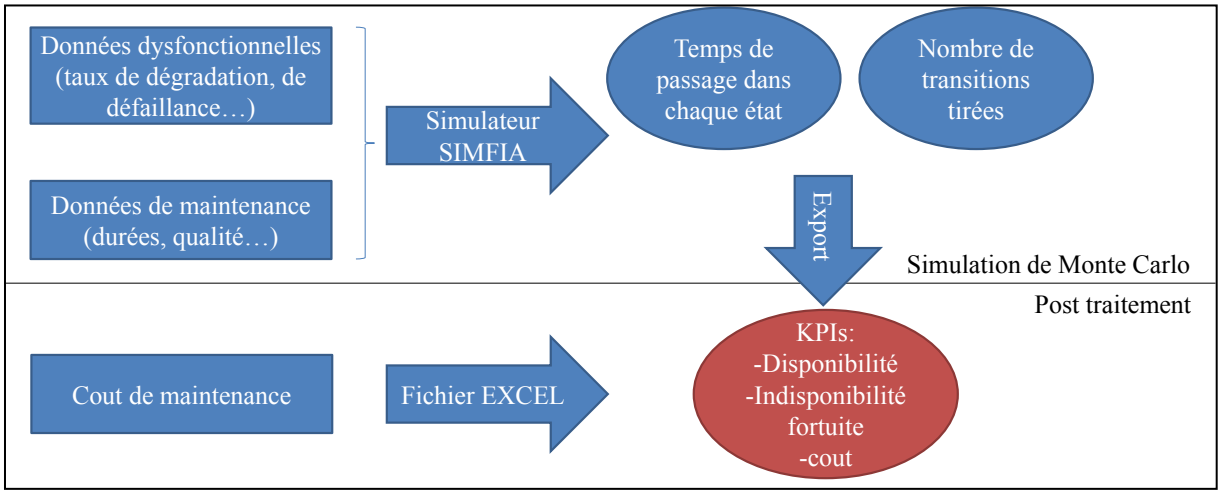


Figure 65 – Obtention des KPIs relatifs à la CMPQ

Dans le contexte de la CMPQ, la non considération du système de soutien équivaut à modéliser des politiques de maintenance prévisionnelles plutôt que conditionnelles (les actions de maintenance sont supposées prêtes à être réalisées), sans pour autant avoir d'informations économiques ce système de soutien. De plus les actions de maintenance correctives sont supposées démarrer instantanément.

D'autres facteurs importants à prendre en compte en vue de l'analyse des résultats issus des taux réels de dégradation sont les faibles taux de dégradations et de défaillance par rapport à la durée de simulation de 10 ans. En effet, la probabilité pour un composant de tomber dans l'état dégradation 1 suit une loi exponentielle avec un λ de l'ordre de 10^{-7}, autrement dit, un composant se dégrade en moyenne une fois tous les 1000000 d'heures, soit environ 14 fois la durée de la mission observée. Ainsi, nous avons simulé un nombre d'histoires important (10000 histoires) en vue d'obtenir des résultats intéressants. Cependant, dans les cas de programmes de maintenance disposant de périodes d'observations très espacées, il serait intéressant d'évaluer les KPIs sur un nombre d'histoires encore supérieur. Ce nombre d'histoires, limité par notre environnement de calcul, nous conduit à définir des intervalles de confiance à 95% sur les valeurs obtenues. Ceci est rendu possible grâce au fait que le simulateur de monte carlo présent dans SIMFIA nous donne la variance obtenue sur les KPIs. Le Tableau 35 et la Figure 66 présentent le KPI lié à l'indisponibilité ainsi obtenu sur le système ARE. Les périodes de maintenance sont données Tableau 34. Pour les obtenir, seul le macro-**node** relatif à l'activité de maintenance a été ré-instancié.

Tableau 34 – Périodes des programmes de maintenance simulés

Notation	Période d'inspection	Période de controle
T_1	6 mois	1 an
T_2	6 mois	1 an et demi
T_3	1 an	2 ans
T_4	1 an	3 ans
T_5	1 an	4 ans
T_6	2 ans	6 ans

Tableau 35 – Indisponibilité du système ARE obtenue pour 10000 histoires sur une période de 72000h

Notation	Indisponibilité moyenne	Borne inférieure de l'intervalle de confiance à 95%	Borne supérieure de l'intervalle de confiance à 95%
T_1	1,88E-03	1,32E-03	2,45E-03
T_2	1,67E-03	1,08E-03	2,26E-03
T_3	7,64E-04	3,80E-04	1,15E-03
T_4	7,06E-04	3,57E-04	1,05E-03
T_5	5,59E-04	2,53E-04	8,66E-04
T_6	3,20E-04	8,20E-05	5,59E-04

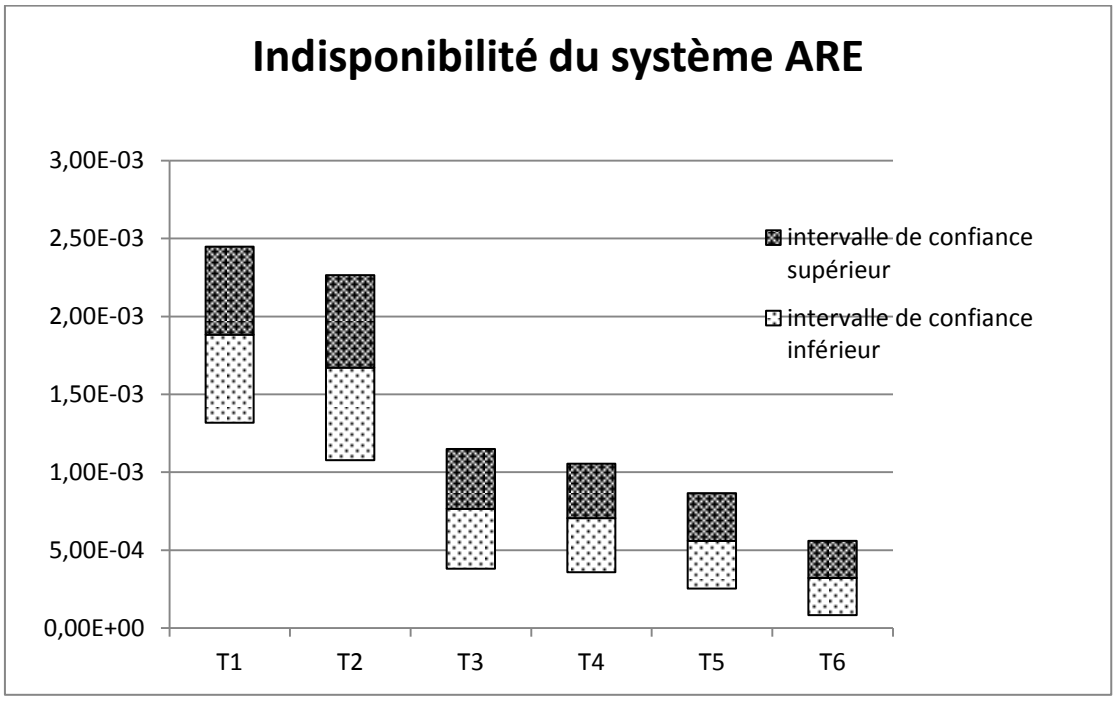


Figure 66 – Indisponibilité du système ARE

Bien qu'insuffisant pour l'aide à la décision sur le système ARE, le KPI lié à l'indisponibilité du système permet toutefois de mettre en valeur l'impact de contrôles (entrainant des arrêts de production) sur le système ARE. En effet, les dégradations/défaillances de celui étant tellement rares, les contrôles constituent le principal facteur d'indisponibilité sur le système. Le programme de période T_1 est donc celui entrainant le plus d'indisponibilité, et le programme T_6 est celui en entrainant le moins. De plus, parmi les programmes disposant de la même période de contrôle (par exemple T_3, T_4 *et* T_5) les programmes entrainant le plus d'indisponibilité du système sont ceux disposant des plus courtes périodes d'inspections. Il est donc essentiel de considérer l'indisponibilité fortuite dans l'analyse des résultats.

Les Figure 67 (a) et (b) présentent respectivement le KPI lié à l'indisponibilité fortuite du système ARE et le KPI lié à la moyenne de pannes subies par le système. Pour le calcul de ces indicateurs, les intervalles de confiance, bien qu'existants, n'ont pas été donnés pour clarifier les figures.

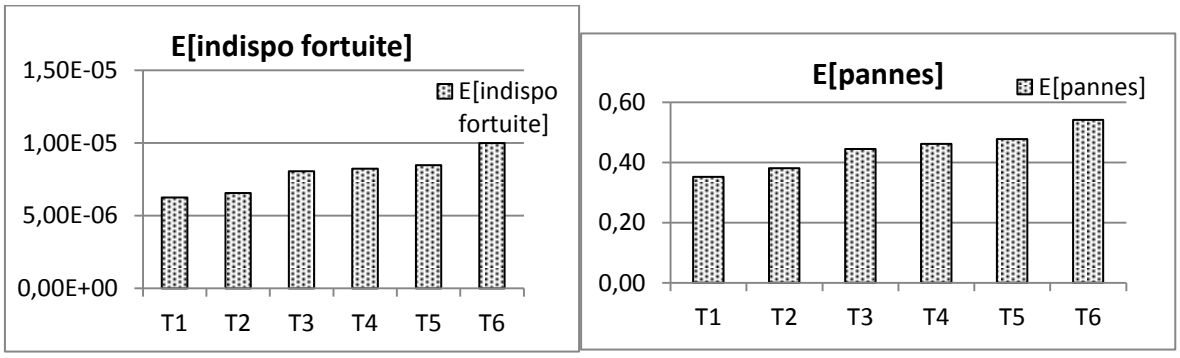


Figure 67 (a) – Indisponibilité fortuite moyenne du système ARE (b) Nombre de pannes moyen du système ARE

Les résultats obtenus valident la cohérence de notre modèle. En effet, si l'indisponibilité globale est minimale lors de l'application du programme de période T_6, l'indisponibilité fortuite et le nombre de pannes moyen sont strictement croissants selon la période de programmes, et donc maximaux pour le programme de période T_6. Cette indisponibilité fortuite est extrêmement critique sur certains systèmes, et pour cela il est important de valider les résultats obtenus.

Ainsi, afin de valider la fiabilité du modèle, il est nécessaire de comparer les KPIs obtenus avec notre démarche avec des résultats de benchmark obtenus avec les mêmes données. Ce benchmark est a été avec les résultats obtenus par (Zille, 2009) en sachant que les simulations réalisées ont été faites avec les RdP. Le Tableau 36 reprend ces résultats.

Tableau 36 – Résultats du benchmark EDF du système ARE

Périodes des inspections						
externe	Interne	E[Np(10)]	D(10)	I(10)	I_fortuite(10)	cout(€)
0,5	1	0,2925	99,80261%	0,0019739	0,000003997	56690,73
0,5	1,5	0,3332	99,82180%	0,0017820	0,000004475	53340,49
1	3	0,4443	99,84115%	0,0015885	0,000005811	52092,90
2	6	0,5488	99,85402%	0,0014598	0,000006930	52421,74

Les résultats obtenus par application de notre démarche de CMPQ sont proches de ceux obtenus avec le benchmark fourni par EDF, notamment en regard de faibles périodes de contrôles et d'inspections. En effet, dans le cas des deux premiers jeux de périodes (respectivement six mois pour les inspections et d'un an pour les contrôles, et six mois pour les inspections et un an et demi pour les contrôles), la différence entre le résultat du benchmark et le résultat obtenu par notre approche est de l'ordre de 10^{-4}, ce qui est satisfaisant. Dans les autres cas cependant, cet écart et de l'ordre de 10^{-3} pour les deux autres jeux de périodes, ce qui représente une différence significative en regard de la rareté de évènements simulés et de la très faible indisponibilité évaluée. L'indisponibilité fortuite et le nombre de panne moyen sont également proches des résultats du benchmark. Concernant l'indisponibilité fortuite, l'écart entre les deux jeux de résultats de l'ordre de 10^{-6} concernant les programmes de périodes T_1et T_2, et de 10^{-7} concernant les programmes de périodes T_4et T_6. Concernant le nombre moyen de pannes, cet écart est de l'ordre de 10^{-2} pour les programmes de périodes T_1et T_2, et de 10^{-3} pour T_4et T_6.

Si utiles soient ces KPIs, ils sont plus facilement interprétables et comparables lorsqu'ils sont traduits sous la forme de couts. Cependant, nous n'avons pas d'informations concernant le cout d'indisponibilité fortuite évalué. Nous ne pouvons dans un premier temps ne donner que les KPIs relatifs aux couts de maintenance préventifs et correctifs, sans y inclure ce cout d'indisponibilité fortuite. Ces couts sont évalués de la façon suivante à partir des résultats donnés par le simulateur. L'équation ci-dessous reprend les notations du Tableau 15, l'indice *j* modélisant les composants sujets

à une maintenance corrective et l'indice i modélisant les composants sujets à une maintenance corrective.

$$C_{main\_tot} = C_{main\_prev} + C_{main\_corr}$$
$$= \sum_i N(debut\_act_{\ i}) * Cact_i + N(debut\_insp) * Cinsp + N(debut\_ctrl) * Cctrl$$
$$+ \sum_j N(debut\_corr_{\ j}) * Ccorr_j$$

La Figure 68 présente les KPIs obtenus par application de la formule.

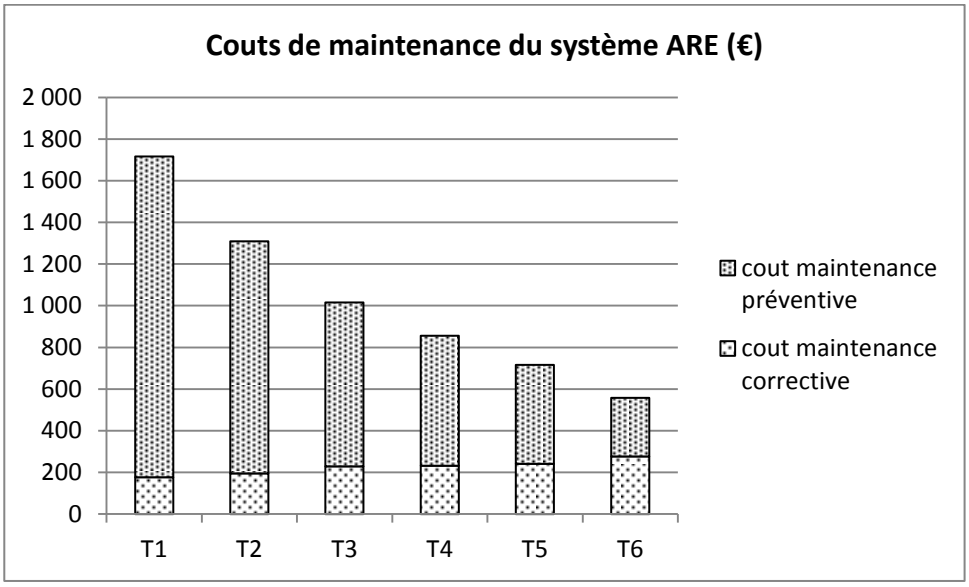


Figure 68 – Couts de maintenance du système ARE

Comme attendu, les couts globaux de chaque programme de maintenance sont proportionnels à l'indisponibilité fortuite entrainée par chaque programme, tandis que les couts correctifs de chaque programme de maintenance sont proportionnels à l'indisponibilité fortuite de chaque programme. De plus, ces KPIs font état de l'importance des couts liés aux observations. Cependant, sans idée du cout d'indisponibilité fortuite du système, il n'est pas possible de se servir de ces KPIs pour prendre une décision sur le système. En ce sens, nous proposons de donner, en fonction de couts d'indisponibilités fortuites données, le programme de maintenance le plus adéquat à appliquer au système d'un point de vue économique.

En utilisant le solveur disponible sous Excel, permettant parallèlement le post-traitement lié à l'évaluation des KPIs, le programme de maintenance le plus adéquat peut être déterminé en fonction des valeurs hypothétiques du cout d'indisponibilité fortuite, arrondies à 10^3. Le Tableau 37 présente ces résultats.

Tableau 37 – Programmes de maintenance adéquats en fonction de l'indisponibilité fortuite

Cout C_{if} d'indisponibilité fortuite (€)	Programme de maintenance le plus économique
$C_{if} < 1400$	T_6
$1400 < C_{if} < 4300$	T_5
$4300 < C_{if} < 19000$	T_2

$19000 < C_{if}$	T_1

L'étude ainsi réalisée permet de montrer l'intérêt de l'application du programme de maintenance de période T_2 sur une large plage de couts d'indisponibilité fortuite.

L'ensemble des résultats précédents a été obtenu simplement en ré-instanciant le macro-**node** relatif à l'activité de maintenance. Un ensemble de KPIs permettant de supporter une aide à la décision sur cette activité est donc disponible. Cependant, notre démarche permet de supporter d'autres types d'aide à la décision, et notamment sur le SP.

VI.5.2 Aide à la décision complémentaire : détermination du moyen adéquat d'agir sur le SP pour un programme de maintenance donné.

Supposons à présent que le programme de maintenance de période T_2 est appliqué sur le SP. D'autres aides à la décision sont possibles, mais cette fois-ci concernant le SP, que ce soit en ré-instanciant le macro-**node** relatif à ce SP, ou le macro-node relatif à l'activité de maintenance. Nous cherchons à présent à identifier dans un premier temps le composant critique du système. Une fois ce composant identifié, les décisions suivantes sont possibles :

- Appliquer au composant une politique préventive particulière,
- Remplacer le composant par un composant équivalent fonctionnellement mais plus fiable (gestion de l'obsolescence).

Le simulateur utilisé, ainsi que la granularité de notre modèle nous permettent d'obtenir par simulation stochastique l'indisponibilité fortuite par composant. La Figure 69 reprend ces résultats, pour des paramètres de simulation identiques aux précédents (10000 histoires sur 72000 heures).

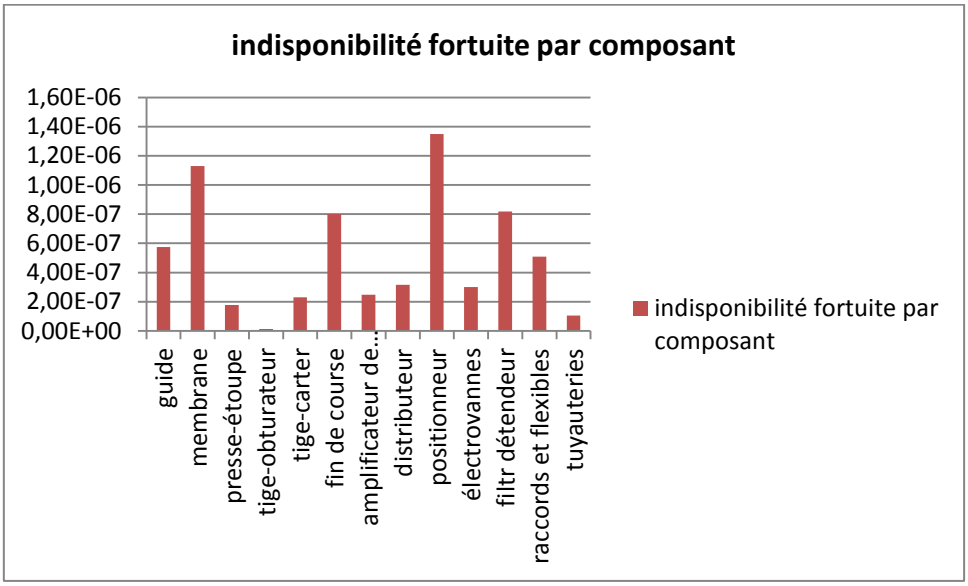


Figure 69 – Indisponibilité fortuite par composant pour un programme de maintenance donné

Ainsi, le positionneur est clairement le composant critique sur le système en termes d'indisponibilité fortuite. Afin de réduire la criticité de celui-ci, nous supposons à présent disposer d'un positionneur de nouvelle génération ne disposant pas du mode de défaillance lié à son encrassement, son mode de défaillance critique. En créant le nouveau macro-**node** relatif au SP (par simple suppression du mode

de défaillance « encrassement du positionneur » dans le modèle), nous obtenons une nouvelle indisponibilité fortuite sur le système. Celle-ci doit cependant être pondérée selon les couts liés à la mise en place du nouveau modèle de positionneur, afin de permettre au décideur de prendre la meilleure décision.

Un autre moyen d'agir sur l'indisponibilité fortuite du positionneur consiste à lui appliquer une politique de maintenance préventive supplémentaire. En reconfigurant l'activité de maintenance par une ré-instanciation du macro-**node** en relation, nous obtenons une nouvelle indisponibilité fortuite sur le système. Celle-ci doit cependant être ensuite mise en relation avec les couts de maintenance supplémentaires liés à cette nouvelle politique de maintenance.

La Figure 70 présente les résultats obtenus concernant l'indisponibilité fortuite moyenne du système, dans le cas où le programme de période T_2 est appliqué, mais aussi dans le cas où ce programme est appliqué en prenant en compte un nouveau positionneur plus performant (T_{2obs}) et dans le cas ou une politique de maintenance préventive supplémentaire basée sur des inspections de période de 1 mois est appliqué au positionneur (T_{2str}).

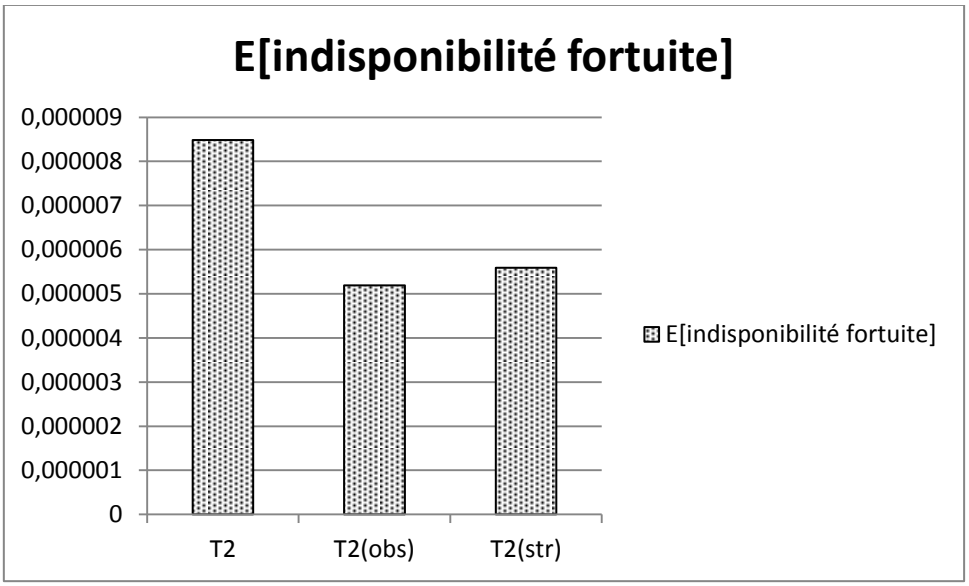


Figure 70 – Comparaison des moyens de réduction de l'indisponibilité fortuite moyenne sur le SP

En comparant les couts obtenus par application d'une nouvelle politique de maintenance et les couts obtenus en remplaçant le positionneur par un nouveau modèle plus performant, à savoir un ensemble de KPIs, une décision peut être prise sur le meilleur moyen de reconfigurer ou non le SP. A présent, nous allons nous intéresser à la mise en place d'un système de soutien en support de l'activité de maintenance.

VI.5.3 Prise en compte du système de soutien

Supposons à présent qu'un opérateur est nécessaire pour réaliser des actions de maintenance actives, dans le cas des taux réels de dégradation. Nous allons pour ce faire considérer trois options permettant de compléter les données observées par des données fournies en concertation avec un expert :

- Une équipe de ressource constituée d'un opérateur disponible à plein temps pour tout le système,

- Une équipe de ressource constituée d'un opérateur sous-traitant (et donc disponible seulement au bout de 8h pour les actions de maintenance correctives) pour tout le système,
- 1 équipe de ressource constituée de 2 opérateurs disponibles à plein temps, l'un s'occupant exclusivement du robinet, et l'autre s'occupant exclusivement de la chaine de régulation,
- 1 équipe de ressource constituée d'un opérateur sous-traitant (et donc disponible seulement au bout de 8h pour les actions de maintenance correctives) pour chaque sous-système.

Afin de construire ces différents cas d'études à partir du cas d'étude précédent, il faut tout d'abord construire le macro-**node** relatif à ce SS. Les modèles atomiques « équipe de ressource » et « opérateur » doivent y être synchronisés et connectés. La structure globale du code relatif au SS dans le cas d'une équipe de ressources constituée de deux opérateurs est donnée Figure 71. La Figure 72 reprend ce code d'un point de vue « boite noire ».

```
node  ARE_SS
sub
  equipe_ressources_ARE: equipe_ressources
  operateur1: ressource_reutilisable
  operateur2: ressource_reutilisable

event
   synchro_fin_travail
sync
  <synchro_fin_travail, equipe_ressources.end_job, operateur1.end_job, operateur2.end_job>

assert
   operateur1.o=equipe_ressource.i1
   operateur2.o=equipe_ressource.i2

node equipe_ressources
 …
edon

node operateur1
 …
edon

node operateur2
 …
edon
```

Code créé par l'identification des nodes atomiques pour la création du macro-node « ARE _SS»

Code créé par la synchronisation des nodes atomiques pour la création du macro-node «ARE_SS»

Code créé par la connection des nodes atomiques pour la création du macro-node «ARE_SS»

Figure 71 - Structure du macro-node du SS du système ARE

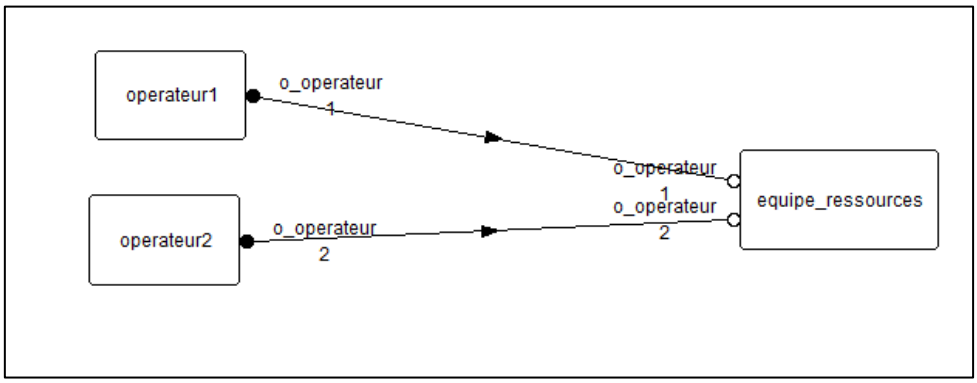


Figure 72 – V ision « boite noire » du SS du système ARE

Ensuite, en procédant conformément à la démarche de création des macro-**nodes**, ce macro-**node** doit être lui-même synchronisé avec les autres macro-**nodes** relatifs au SP et à l'activité de maintenance. Les synchronisations à mettre en place entre ces différents macro-**nodes** sont relatives aux débuts et aux fins d'actions de maintenance actives, déclenchant respectivement les évènements « start_job » des ressources et l'évènement « end_job » des ressources et de l'équipe de ressources.

Dans le cas où un opérateur sous-traitant est en charge du système, seules les synchronisations précédemment définies relatives aux débuts d'actions de maintenance correctives doivent être ré-instanciées pour prendre en compte le délai.

Dans le cas où deux opérateurs sont en charge du système, les synchronisations relatives à l'un des deux sous-systèmes doivent être modifiées afin de prendre en compte les actions du deuxième opérateur.

Pour ces configurations, les résultats concernant les KPIs d'intérêt (disponibilité du système, disponibilité de l'opérateur) et des périodes d'observation de 3 ans pour les contrôles et d'un an pour les inspections sont présentés Figure 73. La légende en ordonnée reprend les hypothèses précédentes, OP signifiant opérateur et ST signifiant sous-traitant.

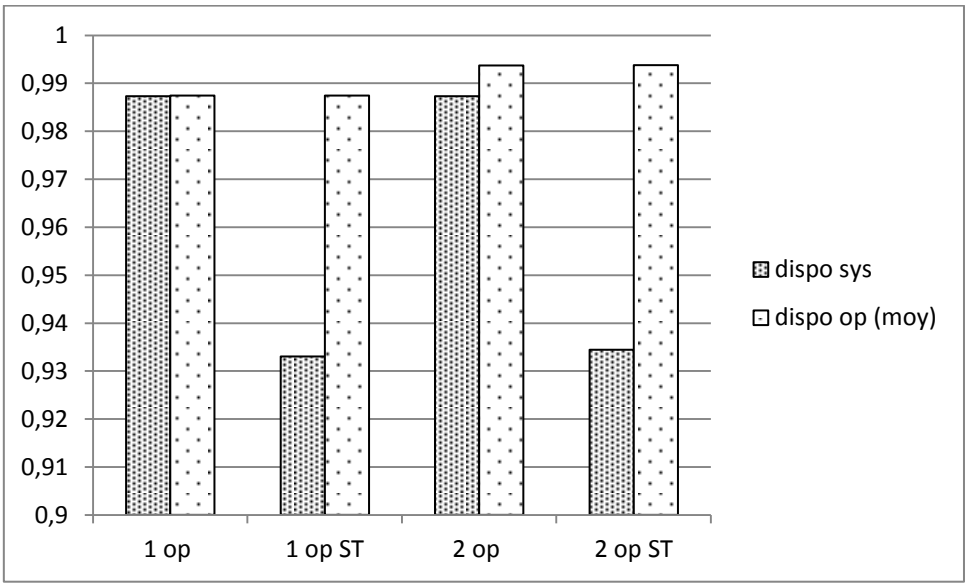


Figure 73 – Disponibilité du système et des opérateurs

Les résultats obtenus mettent en évidence l'impact de la mise en parallèle de deux opérateurs sur le système par leur plus grande disponibilité. De même la disponibilité du système est sensiblement plus élevée dans ce cas. La prise en compte de la sous-traitance se traduit quant à elle par une plus faible disponibilité du système.

Ces indicateurs fournissent un support d'aide à la décision intéressant. Par exemple, sur la base des résultats fournis, et en regard de l'objectif de CMPQ initial, les décisions à prendre pourraient être relatives aux taux de travail des employés. Par conséquent, le taux de satisfaction des employés serait impacté, et des indicateurs de niveau politique de l'entreprise seraient corrélés aux KPIs de la CMPQ.

VI.6. Conclusions des expérimentations

Afin de valider la faisabilité de l'approche de quantification des programmes de maintenance et la crédibilité du modèle de simulation, nous avons appliqué l'approche à un système réel, disposant des résultats d'un benchmark, fourni par EDF : le système ARE.

Bien que parfois peu significatifs en raison des faibles taux de dégradation et de défaillance, les résultats obtenus par simulation de taux de dégradation/défaillance réels permettent dans un premier temps de crédibiliser le modèle ADF de simulation construit et donc en partie notre démarche. En effet, les résultats quantitatifs sont d'une part cohérent concernant l'impact de la maintenance, et d'autre part proches de ceux obtenus avec le benchmark EDF à partir de réseaux de Pétri avec en prime une grande réutilisabilité du modèle par l'intermédiaire de la démarche de création systématique des macro-**nodes** génériques. Toutefois, une validation complémentaire serait nécessaire, par exemple en utilisant un moteur de calcul plus puissant, ce qui aurait permis d'avoir des résultats plus précis, étant donné la rareté des évènements et l'infime probabilité qu'une observation se révèle positive. Cette validation complémentaire pourrait également être fournie par un autre benchmark disposant d'un REX plus important.

Ce benchmark permet également de juger de l'amplitude de la connaissance modélisée. Celle-ci est satisfaisante puisqu'elle permet de modéliser l'ensemble du système ARE en réutilisant seulement 6 atomes de modélisation.

Ensuite, dans le but de montrer que différents types de décisions peuvent être prises sur le système, nous avons cherché quelles décision à appliquer sur le système pour l'application d'un programme de période donnée. En proposant deux moyens d'actions sur la disponibilité du système, nous avons simplement ré-instancié les macros-**nodes** d'intérêts avant de conclure sur le moyen d'action le plus efficace sur cette disponibilité.

Enfin un système de soutien a été considéré, afin de montrer la modularité du modèle proposé et la simplicité avec laquelle il est possible d'en créer de nouveaux macro-**nodes** et de les assembler, toujours sur le même principe de création systématique des macro-**nodes**.

L'obtention de cet ensemble de KPIs pour aider à la prise de décision en regard de la finalité CMPQ est bien en phase avec la genèse scientifique de ce travail. Cette obtention serait d'autant plus facile à mettre en œuvre si un mapping automatique (et non plus manuel) était possible pour la construction du modèle de simulation directement à partir de la connaissance modélisée dans le modèle SysML4CMPQ. Les erreurs de modélisation lors de l'implémentation du code seraient en effet fortement réduites.

Conclusion générale

Les propositions développées dans le cadre de cette thèse ont comme genèse les besoins EDF exprimés dans le cadre du passage de la méthode OMF vers la démarche AP-913, nécessitant la prise en compte de nouvelles connaissances pour modéliser l'activité de maintenance. En ce sens, notre proposition porte sur la définition d'une méthodologie pour la construction d'un modèle de quantification des programmes de maintenance complexes et de sa simulation (CMPQ).

Ainsi, de manière cohérente avec les besoins d'EDF en termes d'évaluation des performances de maintenance, cette proposition a permis, de façon duale, d'aborder une vision industrielle et scientifique de la maitrise des performances en regard du besoin de quantification pour contribuer aux différents verrous scientifiques suivants :

- L'identification des indicateurs (KPIs) d'intérêt pour la CMPQ,
- L'identification de la connaissance nécessaire à l'évaluation de ces KPIs, selon un système principal et ses points de vue fonctionnels et dysfonctionnels, mais aussi son système de soutien et les conditions dans lesquelles il évolue,
- La double formalisation de cette connaissance générique à deux fins : un modèle construit dans le langage semi formel SysML permettant de capitaliser le connaissance sans entrainer de pertes par rapport à la connaissance générique, et un modèle développé dans le langage ADF permettant la simulation stochastique de la connaissance modélisée.

Cette méthodologie aboutit au développement de deux modèles génériques. Le premier, appelé modèle SysML4CMPQ, formalisé à base du langage SysML, représente de manière univoque les connaissances issues des normes et standards relatives à un système complexe quelconque soumis à la maintenance sous la forme de concepts de modélisation. Ces concepts ont été et validés avec un expert EDF et peuvent être vérifiés par l'outil Rhapsody

Cependant, ce langage n'étant pas adapté à la simulation stochastique, le langage ADF a ensuite été choisi (support au second modèle générique), notamment pour son aptitude à la modularité, et des règles systématiques permettant un mapping unidirectionnel depuis les éléments sémantiques d'intérêt du langage SysML vers les éléments sémantiques d'intérêt du langage ADF ont été construites pour réaliser le passage entre ces deux langages. Ainsi, un résultat de cette thèse est d'une part un ensemble d'atomes de modélisation en langage ADF réutilisables et cohérents avec le modèle SysML4CMPQ initial, et d'autre part une démarche systématique de création de macro-nodes génériques incluant les aspects d'ADF liés à la gestion de la complexité du langage. Ainsi, un utilisateur sans connaissance particulière en programmation peut construire un modèle de système simulable afin d'évaluer des KPIs relatifs à la CMPQ.

En termes de validation de sa faisabilité, la méthodologie a été appliquée à un cas d'étude réel fourni par EDF : le système ARE. L'objectif était de démontrer son applicabilité après passage à l'échelle de la modélisation mais aussi d'évaluer la pertinence et le bénéfice de la méthodologie vis-à-vis de l'objectif initial de choix d'une politique de maintenance. L'analyse du REX de ce système a permis d'éliciter des taux de dégradations, et un entretien avec un expert a permis d'établir les données liées à

la maintenance de manière cohérente. Les résultats obtenus par simulation du code ADF sous l'environnement SIMFIA sont proches de ceux du benchmark d'EDF, crédibilisant ainsi les atomes de modélisation proposés. Toutefois, la rareté des évènements entraine parfois un écart non négligeable sur des valeurs très faibles. Pour compléter cette validation, il serait donc intéressant d'effectuer la simuation avec un moteur de calcul plus puissant.

Dans une seconde phase, nous avons complété les données observées sur le système ARE par d'autres données (nouvelles politiques de maintenance, obsolescence, apport d'un système de soutien) fournies en concertation avec un expert afin de montrer d'autres valeurs ajoutées de la méthode, et notamment l'utilisabilité du modèle. En effet, le modèle proposé est facilement paramétrable et réutilisable par l'intermédiaire des macro-**nodes** ADF génériques. Il permet donc de réduire considérablement l'effort de modélisation à chaque étude. Cependant, l'implémentation de la démarche systématique de construction des macro-**nodes** peut parfois s'avérer fastidieuse (dans le cas de la définition d'un grand nombre de synchronisations par exemple).

Une perspective découlant de ce point serait donc l'implantation physique d'une passerelle entre un environnement stockant le modèle SysML4CMPQ du système et l'environnement de simulation afin d'automatiser complétement et de rendre transparentes ces synchronisations. Les relations algébriques entre les éléments sémantiques d'intérêt des différents langages proposées constituent une base concrète et importante en vue de cette implantation. Dans le même objectif, ces règles peuvent permettre d'améliorer une proposition de transformation déjà existante entre ces langages. Plus généralement, un travail sur l'architecture de l'outil proposé en regard des langages utilisés fournirait une vision plus précise sur son utilisabilité.

Concernant l'amplitude de la connaissance modélisée, elle est satisfaisante puisqu'elle permet de modéliser l'ensemble du système ARE avec 6 atomes de modélisation. Cependant, pour étendre notre proposition à d'autres applications, un domaine de modélisation plus important, prenant par exemple en compte la gestion des stocks, des informations spécifiquement liées au domaine de la production, ou encore des aspects liés aux couts externes au système (inflation…), peut être considéré. Il permettrait alors d'étoffer le modèle avec d'autres concepts de modélisation. Or, si l'ensemble des concepts modélisés sont simulables par l'intermédiaire des atomes de modélisations ADF, la non possibilité de considération des systèmes bouclés contraint ces assemblages.

Une perspective permettant de résoudre ce problème serait la proposition d'un nouveau mapping, toujours depuis les éléments d'intérêt SysML du modèle SysML4CMPQ modélisant la connaissance, mais cette fois-ci vers un autre langage de simulation ne présentant pas ces contraintes, tel que par exemple les systèmes de transitions gardées, à condition qu'il dispose d'un environnement de simulation adapté. Par rapport à la démarche proposée, seules les règles algébriques de mapping entre les éléments sémantiques des langages seraient à reconsidérer. Plus généralement, quelque soit le langage de simulation choisi, une transformation de langage fondée scientifiquement (voir section III.5.) serait une perspective intéressante.

Enfin, si la robustesse du modèle est correcte en comparaison avec les résultats d'autres approches, des vérifications et des validations complémentaires sont nécessaires (voir section III.7), notamment

au niveau du modèle SysML4CMPQ, dans le cas où celui-ci servirait de pivot à une autre étude de CMPQ.

Publications de l'auteur

Ruin, T., Levrat, E., Iung, B., 2013. Maintenance programs quantification to better control production system: From SysML-based models for knowledge formalization to Altarica-based stochastic model for simulation, *submitted to Journal of Manufacturing Technology management* (en révision mineure)

Ruin, T., Levrat, E., Iung, B., 2012. Model-driven framework based on SysML and AltaRica Data Flow languages for complex maintenance program quantification*; A-MEST 2012 (2nd IFAC Workshop on Advanced Maintenance Engineering, Services and Technology)* November 22-23, 2012, Universidad de Sevilla, Sevilla, Spain.

Ruin, T., Levrat, E., Iung, B., Despujols, A., 2012. Using SySML language for maintenance decision-making model development to support complex maintenance program quantification*; PSAM 2011 & ESREL 2012*, Conference, 25 - 29 June 2012, Helsinki, Finland.

Medina-Oliva, G., Iung, B., Barberá, L., Viveros, P., Ruin, T., 2012. Root cause analysis to identify physical causes*; PSAM11 & ESREL 2012 Conference*, 25 - 29 June 2012, Helsinki, Finland.

Ruin, T.,. Cocheteux, P., Levrat, E., 2009. Formalisation de la décision basée sur un pronostic pour la maintenance prévisionnelle ; *PENTOM 2009*, 7 - 8 décembre 2009, Autrans.

Bibliographie

Adeline, R., 2011. *Méthodes pour la validation de modèles formels pour la sûreté de fonctionnement et extension aux problèmes multi-physiques*. Thèse de doctorat, Université de Toulouse.

AFNOR, 2001. *NF EN 13306 Terminologie de la maintenance*,

Alsyouf, I., 2007. The role of maintenance in improving companies' productivity and profitability. *International Journal of Production Economics*, 105(1), pp.70–78.

Arbaretier, E. Brik, Z., Brindejonc, V., Desmarquest, B, 2010. ration des modèles de SDF de conception systèmes en niveaux d'abstraction. In *Actes du congrès Lambda-Mu 17*. La Rochelle (France).

Aven, T., Sklet, S. & Vinnem, J.E., 2006. Barrier and operational risk analysis of hydrocarbon releases (BORA-Release). Part I. Method description. *Journal of hazardous materials*, 137(2), pp.681–91.

Baraldi, P. et al., 2011. Modeling the effects of maintenance on the degradation of a water-feeding turbo-pump of nuclear power plant. In *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*. pp. 169–183.

Barros, A., Berenguer, C. & Grall, A., 2006. A maintenance policy for two-unit parallel systems based on imperfect monitoring information. *Reliability Engineering & System Safety*, 91(2), pp.131–136.

Ben Salem, A., 2008. *Modèles probabilistes de séquences temporelles et fusion. application à la classification de défauts de rails et à leur Maintenance*. Thèse de doctorat, Université Henri Poincaré, Nancy I.

Bernardi, S., Donatelli, S. & Merseguer, J., 2002. From UML Sequence Diagrams and Statecharts to analysable Petri Net models. In *In Proceedings of the 3rd Int. workshop on software on performance*. Rome, Italy.

Bernardi, S., Merseguer, J. & Petriu, D.C., 2008. Adding Dependability Analysis Capabilities to the MARTE Profile. In *Proc. of MoDELS '08*. Springer-Verlag, p. pages 736–750.

Berrade, M.D., 2012. A two-phase inspection policy with imperfect testing. *Applied Mathematical Modelling*, 36(1), pp.108–114.

Betous-Almeida, C. & Kanoun, K., 2004a. Construction and stepwise refinement of dependability models. *Performance Evaluation*, 56(1-4), pp.277–306.

Betous-Almeida, C. & Kanoun, K., 2004b. Dependability modelling of instrumentation and control systems. *Safety Science*, 42(5), pp.457–480.

Bevilacqua, M. & Braglia, M., 2000. The analytic hierarchy process applied to maintenance strategy selection. *Reliability Engineering & System Safety*, 70(1), pp.71–83.

Biswas, A. & Sarkar, J., 2000. Availability of a system maintained through several imperfect repairs before a replacement or a perfect repair. *Statist. Probab. Lett.*,, 50, pp.105–114.

Boiteau, M. et al., 2006. The AltaRica data-flow language in use: modeling of production availability of a multi-state system. *Reliability Engineering & System Safety*, 91(7), pp.747–755.

Bottelberghs, P.H., 2000. Risk analysis and safety policy developments in the Netherlands. *Journal of hazardous materials*, 71(1-3), pp.59–84.

Bouissou, M., 2005. Dix petits problèmes de modélisation (en sûreté de fonctionnement des systèmes). *Revue Phoebus*, 34.

Bouissou, M. et al., 2002. KB3 tool: feedback on knowledge bases. In *ESREL*. Lyon (France).

Bouissou, M. & Seguin, C., 2006. Comparaison des langages de modélisation AltaRica et Figaro. In *Actes du congrès Lambda Mu'15.* Lille (France).

Branscomb, J.M. et al., 2013. Supporting Multidisciplinary Vehicle Analysis Using a Vehicle Reference Architecture Model in SysML. *Procedia Computer Science*, 16(0), pp.79–88.

Brik, Z. & Secher, P., 2010. Optimisation de la disponibilite d'un système de production intégrant l'aspect environnement. In *JIRITE*. Oran.

Brissaud, F. et al., 2010. Failure rate evaluation with influencing factors. *Journal of Loss Prevention in the Process Industries*, 23(2), pp.187–193.

Carnero, M.C., 2005. Selection of diagnostic techniques and instrumentation in a predictive maintenance program. A case study. *Decision Support Systems*, 38(4), pp.539–555.

Cha, J.H. & Kim, J.J., 2001. On availability of Bayesian imperfect repair model. *Statist. Probab. Lett.*, 53(February), pp.181–187.

Chavanis, P.-H., 2008. Hamiltonian and Brownian systems with long-range interactions: V. Stochastic kinetic equations and theory of fluctuations. *Physica A: Statistical Mechanics and its Applications*, 387(23), pp.5716–5740.

Chouiki, H., Khatab, A. & Rezg, N., 2012. Condition-based maintenance for availability optimization of production system under environment constraints. In *9th Intrernational Conference of Modeling, Optimization and Simulation - MOSIM'12*. Bordeaux - France.

Clavareau, J. & Labeau, P.-E., 2009a. A Petri net-based modelling of replacement strategies under technological obsolescence. *Reliability Engineering & System Safety*, 94(2), pp.357–369.

Clavareau, J. & Labeau, P.-E., 2009b. Maintenance and replacement policies under technological obsolescence. *Reliability Engineering & System Safety*, 94(2), pp.370–381.

Cocheteux, P., 2010. *Contribution à la maintenance proactive par la formalisation du processus de pronostic des performances de systèmes industriels*. Thèse de doctorat, Université Henri Poincaré Nancy I.

Cocheteux, P., Voisin, A., Levrat, E., Iung, B., 2009. Methodology for assessing system performance loss within a proactive maintenance framework. In *13th IFAC Symposium on Information Control Problems in Manufacturing, INCOM'09*. Moscow.

Crane, M.L., 2006. *On the Syntax and Semantics of State Machines*. PhD dissertation, Queen's University.

Czarnecki, 2006. Feature-based survey of model transformation approaches. *IBM Systems Journa*.

Cressent, R., David, P., Idasiak, F. & Kratz, F., 2013. Desining the database for a reliability aware Model-Based System Engineering Process. *Reliability Engineering & System Safety*, 111, pp.171-182.

David, P., 2009. *Contribution à l ' analyse de sûreté de fonc- tionnement des systèmes complexes en phase de conception : application à l ' évaluation des missions d ' un réseau de capteurs de présence humaine*. Thèse de doctorat, Université d'Orléans.

David, P., Idasiak, V. & Kratz, F., 2010. Reliability study of complex physical systems using SysML. *Reliability Engineering & System Safety*, 95(4), pp.431–450.].

Dekker, R., 1996. Applications of maintenance optimization models: a review and analysis. *Reliability Engineering & System Safety*, 51(3), pp.229–240.

Delia, M.-C. & Rafael, P.-O., 2008. A maintenance model with failures and inspection following Markovian arrival processes and two repair modes. *European Journal of Operational Research*, 186(2), pp.694–707.

Despujols, A., 2004. *Optimisation de la maintenance par la fiabilité (OMF)*, Techniques de l'ingénieur, dossier MT9310.

Dobre, D., 2010. *Contribution à la modélisation d'un système interactif d'aide à la conduite d'un procédé industriel*. Thèse de doctorat, Université Henri Poincaré Nancy I.

Doyen, L. & Gaudoin, O., 2004. Classes of imperfect repair models based on reduction of failure intensity or virtual age. *Reliability Engineering & System Safety*, 84(1), pp.45–56.

Dumas, X. et al., 2008. Vers la génération de modèles de sûreté de fonctionnement. In *Proceedings of Conférence sur les Architectures Logicielles*. Montreal (Canada).

Dutuit, Y. et al., 1997. Dependability modelling and evaluation by using stochastic Petri nets: application to two test cases. *Reliability Engineering & System Safety*, 55(2), pp.117–124.

Estefan, J.A., 2008. Survey of Model-Based Systems Engineering (MBSE) Methodologies. In *Version B, INCOSE*.

Eti, M.C., Ogaji, S.O.T. & Probert, S.D., 2006. Reducing the cost of preventive maintenance (PM) through adopting a proactive reliability-focused culture. *Applied Energy*, 83(11), pp.1235–1248.

Harel, D., 1987. Statecharts: a visual formalism for complex systems. *Science of Computer Programming*, 8, pp.231–274.

Hoepfer, V.M., Saleh, J.H. & Marais, K.B., 2009. On the value of redundancy subject to common-cause failures: Toward the resolution of an on-going debate. *Reliability Engineering & System Safety*, 94(12), pp.1904–1916.

Hoffmann, 2011. *Systems Engineering Best Practices with the Rational Solution for Systems and Software Engineering Deskbook*, IBM Software Group.

Hua, C. et al., 2012. Long-term potential performance degradation analysis method based on dynamical probability model. *Expert Systems with Applications*, 39(4), pp.4410–4417.

Huynh, K.T. et al., 2011. A periodic inspection and replacement policy for systems subject to competing failure modes due to degradation and traumatic events. *Reliability Engineering & System Safety*, 96(4), pp.497–508.

Huynh, K.T. et al., 2012. Modeling age-based maintenance strategies with minimal repairs for systems subject to competing failure modes due to degradation and shocks. *European Journal of Operational Research*, 218(1), pp.140–151.

Jardine, A.K.S., Lin, D. & Banjevic, D., 2006. A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7), pp.1483–1510.

Jeang, A., 1998. Reliable tool replacement policy for quality and cost. *European Journal of Operational Research*, 108(2), pp.334–344.

Jin, G. et al., 2013. Physics of failure-based degradation modeling and lifetime prediction of the momentum wheel in a dynamic covariate environment. *Engineering Failure Analysis*, 28, pp.222–240.

Kallen, M.J. & van Noortwijk, J.M., 2005. Optimal maintenance decisions under imperfect inspection. *Reliability Engineering & System Safety*, 90(2-3), pp.177–185.

Kehren, C., 2005. *Motifs formels d'architectures de systèmes pour la sûreté de fonctionnement*. Thèse de doctorat, ENSAE.

Kogelnik, H., Jopson, R.M. & Nelson, L.E., 2002. Chapter 15 - Polarization-Mode Dispersion. In I. P. Kaminow & T. Li, eds. *Optical Fiber Telecommunications IV-B (Fourth Edition)*. Burlington: Academic Press, pp. 725–861.

Lair, W., 2012. Quantification de la maintenance du système AREH-T58-2012-01757-FR. Technical report, EDF.

Lee, J., 1995. Modern computer-aided maintenance of manufacturing equipment and systems: review and perspective. *Computers & industrial engineering*, 28(4).

Lorino, P., 2001. Le balanced scorecard revisité: dynamique stratégique et pilotage de performance, exemple d'ue entreprise énergétique. *In: 22ème congrès de l'AFC.*

Marseguerra, M. et al., 2003. Fuzzy logic for signal prediction in nuclear systems. *Progress in Nuclear Energy*, 43(1-4), pp.373–380.

Marseguerra, Marzio, Zio, Enrico & Podofillini, L., 2002. Condition-based maintenance optimization by means of genetic algorithms and Monte Carlo simulation. *Reliability Engineering & System Safety*, 77(2), pp.151–165.

Medina Oliva, G., 2011. *Modélisation conjointe des connaissance multi-points de vue d'un système industriel et de son système de soutien pour l'évaluation des stratégie de maintenance*. Thèse de doctorat, Université Henri Poincaré Nancy I.

Mens, & Van Gorp, 2006. A Taxonomy of Model Transformation. *Electr. Notes Theor. Comput. Sci.*, 152: 125-142.

Moghaddam, K.S. & Usher, J.S., 2011. Preventive maintenance and replacement scheduling for repairable and maintainable systems using dynamic programming. *Computers & Industrial Engineering*, 60(4), pp.654–665.

Monnin, M., 2007. *Approche unifiée défaillance/dommage dans la sûreté de fonctionnement pour la régénération des matériels de combat*. Thèse de doctorat, Université de Valenciennes.

Monnin, M., Iung, B. & Se, O., 2011. Dynamic behavioural model for assessing impact of regeneration actions on system availability : Application to weapon systems. , 96, pp.410–424.

Moore, W.J. & Starr, a. G., 2006. An intelligent maintenance system for continuous cost-based prioritisation of maintenance activities. *Computers in Industry*, 57(6), pp.595–606.

Nikolaidou, M., Dalakas, V. and Anagnostopoulos, D., 2010. Integrating simulation capabilities in SysML using DEVS. In Proceedings of IEEE Systems Conference 2010, San Diego, California, USA.

Nottage, D. & Corns, S., 2011. SysML profiling for handling army base camp planning. *Procedia Computer Science*, 6, pp.63–68.

Nowakowski, T. & Werbińka, S., 2009. On problems of multicomponent system maintenance modelling. *International Journal of Automation and Computing*, 6(4), pp.364–378.

OMG, 2010. *Systems Modeling Language (OMG SysML)*, Objecti management group.

OMG, 2011. *Unified Modeling Language (OMG UML), Superstructure*, Object Management Group.

Paredis, C.J.J. et al., 2010. An Overview of the SysML-Modelica Transformation Specification. In *INCOSE International Symposium*.

Parida, A., 2006. *Development of a Multi-criteria Hierarchical Framework for Maintenance Performance Measurement Concepts , Issues and Challenges*. Doctoral thesis, Lulea University of Techonlogy.

Pennington, M. & Strùmme, T., 1998. Surveys as a research tool for managing dynamic stocks. *Fisheries Research*, 37, pp.97–106.

Perisse, F., 2003. *Etude et analyse des modes de défaillances des condensateurs électrolytiques à l'aluminium et des thyristors, appliquées au système de protection du LHC (Large Hadron Collider)*. Thèse de doctorat, Université de Lyon.

Proverbs, D.G. & Holt, G.D., 2000. Reducing construction costs : European best practice supply chain implications. *European Journal of Purchasing & Supply Management*, 6.

Pérès, F. & Noyes, D., 2003. Evaluation of a maintenance strategy by the analysis of the rate of repair. *Quality and Reliability Engineering International*, 19(2), pp.129–148.

Ramesh, a. . et al., 1999. An integrated reliability modeling environment. *Reliability Engineering & System Safety*, 65(1), pp.65–75.

Rauzy, A., 2002. Mode automata and their compilation into fault trees. *Reliability Engineering & System Safety*, 78(1), pp.1–12.

Ravichandran, R., 1993. A decision support system for stochastic cost-volume-profit analysis. *Decision Support Systems*, 10(4), pp.379–399.

Reggio, G. & Wieringa, R.J., 1999. Thirty one Problems in the Semantics of UML 1 . 3 Dynamics. In *Rigourous Modelling and analysis of the UML: Challenges and limitations*.

Ribeiro, R.A., 1996. Fuzzy multiple attribute decision making : A review and new preference elicitation techniques. , 78, pp.155–181.

Rolland, J.-F., 2008. *Développement et validation d'architectures dynamiques*. Thèse de doctorat, Toulouse III - Paul Sabatier.

Sanajian, N. & Balcıog˘lu, B., 2009. The impact of production time variability on make-to-stock queue performance. *European Journal of Operational Research*, 194(3), pp.847–855.

Savsar, M., 2006. Effects of maintenance policies on the productivity of flexible manufacturing cells. *Omega*, 34(3), pp.274–282.

Sénéchal, O,. 2004. *Pilotage des systèmes de production vers la performance.* Rapport d'habilitation à diriger les recherches, UBHC-LAMIH.

Thomas, E., 2009. *Contribution à la prise de décision dynamique en maintenance prévisionnelle par formalisation d'un principe d'opportunité*. Thèse de doctoral, Université Henri Poincaré-Nancy I.

UTE, 2010. *Union technique de l'électricité, méthodologie de fiabilité pour les systèmes électroniques*,

Wang, W., 2012. An overview of the recent advances in delay-time-based maintenance modelling. *Reliability Engineering & System Safety*, 106, pp.165–178.

Weber, P. & Jouffe, L., 2006. Complex system reliability modelling with Dynamic Object Oriented Bayesian Networks (DOOBN). *Reliability Engineering & System Safety*, 91(2), pp.149–162.

Wieringa, R. (1998). A Survey of Structured and Object-Oriented Software Specification Methods and Techniques. *ACM Computing Surveys*, 30(4):459--527.

Yu, Q., Chen, P. & Wang, L., 2013. Degradation in mechanical and physical properties of carbon fiber/bismaleimide composite subjected to proton irradiation in a space environment. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 298, pp.42–46.

Zeigler, B.P. & Nk-, D.C., 1998. DEVS Theory of Quantized Systems Department of Electrical and Computer Engineering University of Arizona , Tucson , Arizona A Deliverable in Fulfillment of Advance Simulation Technology Thrust (ASTT). , (June).

Zhou, X., Xi, L. & Lee, J., 2007. Reliability-centered predictive maintenance scheduling for a continuously monitored system subject to degradation. *Reliability Engineering & System Safety*, 92(4), pp.530–534.

Zille, V., 2009. *Modélisation et évaluation des stratégies de maintenance complexes sur des systèmes multi-composants*. Thèse de doctorat, UTT.

Zio, E. & Baraldi, P., 2003. Sensitivity analysis and fuzzy modelling for passive systems reliability assessment. *Annals of Nuclear Energy*, 31(3), pp.277–301.