Etude pour une meilleure intégration des données de conception dans les analyses de fiabilité

Pierre David, Vincent Idasiak, Frédéric Kratz

ETUDE POUR UNE MEILLEURE INTEGRATION DES DONNEES DE CONCEPTION DANS LES ANALYSES DE FIABILITE
STUDY FOR A BETTER INTEGRATION OF DESIGN DATA IN RELIABILITY ANALYSIS

Pierre DAVID, Vincent IDASIAK et Frédéric KRATZ
Institut PRISME – équipe MCDS / Ecole Nationale Supérieure d'Ingénieurs de Bourges
88 Boulevard Lahitolle
18020 Bourges

Résumé
De nombreux travaux ont montré, que la communication entre les experts de chaque domaine est l'un des points clef du développement des nouveaux systèmes complexes. Pour relever ce défi dans l'industrie du logiciel, l'OMG (Object Management Group) a développé UML (Unified Modelling Language). Depuis plusieurs années, de nombreux chercheurs travaillent sur l'adaptation de UML à l'ingénierie système. Ces efforts ont conduit depuis peu à la création de SysML, un nouveau langage pour la spécification, l'analyse la conception et la validation de systèmes complexes multi-technologiques. Dans cet article, nous présentons comment ces langages peuvent être intégrés pour combiner conception et analyse de sûreté de fonctionnement (SdF), dans le but de maîtriser les impératifs de SdF tout au long du processus de création. Pour cela, nous présentons une méthode d'analyse des critères de SdF s'appuyant sur les procédures classiques de conception, ceci dans le but de ne pas venir perturber le travail des concepteurs. Notre objectif est d'une part de faciliter la communication entre les concepteurs et les analystes SdF, et d'autre part d'accélérer et simplifier l'analyse SdF. Nous nous baserons sur l'emploi d'AMDEC pour la déduction des comportements dysfonctionnels. Nous présenterons notre solution pour la synthèse automatique de tableau AMDEC à partir de modèle UML/SysML décrivant uniquement le comportement fonctionnel du système. Nous insisterons en particulier sur l'apport de l'utilisation d'une base de données structurée pour la gestion du retour d'expérience. Le but de cet article est donc de décrire les premières étapes d'une nouvelle approche pour accélérer et simplifier l'analyse SdF de systèmes complexes. Cette approche s'appuie sur les langages classiquement utilisés en conception, elle propose d'automatiser au maximum les différentes étapes de création et d'analyse de modèles.

Summary
It is commonly admitted that one of the crux, during the design process of new complex systems, is the efficient communication between experts of different domains. The OMG developed UML to tackle this problem in software design. For several years, many researchers have been working on adapting and reusing UML for whole type of systems. These efforts led to the creation of SysML, a new modeling language for specifying, analyzing, designing and verifying complex multi-disciplinary systems. In this paper, we present how those modeling languages can be integrated to combine reliability study with the design process in order to compose a conception process driven by dependability priorities. We have set up a method to conduct reliability studies without perturbing the classic procedures of designers. Our objectives were on the one hand to facilitate communication between functional and dysfunctional analyzers and to simplify the execution of reliability studies on the other hand. In this article, we raise the utility of FMEA for a non-intrusive modeling of dysfunctional aspects. We thus give our solution for an automatic synthesis of FMEA from UML/SysML models, which only describe the nominal behavior. We insist on the keystone of this automatic synthesis, namely the use and management of a database constructed from the information of the return on experience. The purpose of this article is thus to describe the first steps of a new approach to enhance the realization of dependability studies by automating steps and using easily understandable models.

1. Introduction

La complexité des nouveaux systèmes ne cesse de grandir, il devient évident que la complexité de la phase de développement est liée au nombre de technologies différentes utilisées dans le système. Par exemple, la nécessité d'intégrer des composants logiciels a soulevé de nombreux problèmes auxquels se sont intéressés de nombreuses équipes de recherche. L'origine de ces problèmes d'intégration est bien souvent la difficulté de communiquer entre experts de domaines différents. Les vocabulaires employés par les spécialistes ne sont pas homogènes, ce qui conduit à des erreurs de compréhension et d'interprétation pouvant se révéler source d'erreurs. Le problème est le même lorsqu'il s'agit de considérer la réalisation de modèles. Il n'est pas toujours aisé de synchroniser des modèles spécifiques à chaque technologie employée. De même, les modèles employés en SdF pour qualifier et quantifier le comportement dysfonctionnel des systèmes sont trop souvent difficiles à comprendre et à construire par les ingénieurs de conception. C'est pourquoi les études SdF et les démarches de conception sont souvent perçues comme deux tâches bien distinctes du processus de développement. Ceci réduit fortement l'intérêt des analyses SdF, qui sont alors cantonnées à un rôle de validation plus qu'à un rôle d'aiguillage dans le design. En résumé, nous pouvons dire que les concepteurs et analystes sont confrontés à deux problèmes majeurs que sont, la communication inter-discipline et l'intégration multi-technologique. Le défi pour une étude efficace de SdF est donc de pouvoir combiner les modèles utilisés en conception et les méthodes d'analyse spécifiques au domaine de la SdF. Dans cet article, nous présentons la démarche et les outils développés dans le but de relier le plus rapidement possible les choix de conception et leurs impacts sur la SdF globale du système. Cette démarche a pour ambition de rendre à l'analyse SdF un rôle moteur dans le processus de développement. Dans la suite de ce papier, nous soulignerons l'intérêt de l'utilisation d'UML et SysML pour le design de système complexe. Nous présenterons ensuite l'utilisation de ces langages dans le cadre d'analyse SdF. Puis, nous introduirons notre méthode utilisant les diagrammes UML/SysML, avant de décrire plus en détail la première étape. Cette dernière permet de déduire le caractère dysfonctionnel d'un système à partir de son comportement fonctionnel, par le biais de la génération automatisée d'AMDEC. Nous présenterons un exemple d'utilisation de cette méthode sur un système de contrôle d'une cuve.

2. L'analyse SdF de système à partir de modèles UML et SysML

2.1 Des langages pour la conception de systèmes multi-technologiques

Comme nous l'avons mentionné en introduction, la bonne communication entre les acteurs d'un projet est un point essentiel dans le développement des systèmes complexes. Cette problématique est très présente dans le domaine de l'ingénierie logicielle, où le client final a souvent de grosses difficultés pour appréhender le vocabulaire très spécifique et précis du monde de l'informatique. L'expression des besoins dans ce domaine est devenue délicate et source d'importantes erreurs. Pour tenter de mettre un terme à ces difficultés, la communauté scientifique a mis au point différents langages synthétisés de nos jours en une unique spécification : UML. UML a été spécifié dans le but de modéliser les projets informatiques ou les systèmes incluant du logiciel sous une forme compréhensible par tous. Ce langage a largement atteint ses objectifs et est actuellement très employé dans le domaine du développement logiciel.

Depuis la spécification 2.0, UML n'est plus réservé à la programmation, grâce à son approche orientée-objet (OO), il devient exploitable pour la description de tous types de systèmes. De nombreuses publications mentionnent les avantages de l'approche OO pour l'ingénierie de systèmes complexes et conseillent donc l'emploi de l'UML 2.0. Néanmoins, en raison de sa sémantique trop orientée logicielle, il n'est encore que peu utilisé pour la conception de systèmes n'incluant pas une part significative de logiciel. Par conséquent une attente forte existe toujours pour un langage permettant la description de système où le caractère pluridisciplinaire est prédominant.
C'est pourquoi l'OMG a créé SysML [1] pour spécifier, analyser, concevoir et valider les systèmes complexes. SysML a été construit depuis avril 2003 comme une extension d'UML adaptée au vocabulaire de l'ingénierie système et apportant de nouveaux moyens pour la modélisation et la spécification. Dans [2], l'auteur souligne l'héritage provenant d'UML 2.0 et introduit les nouvelles possibilités de SysML en matière d'accessibilité et d'adaptabilité à des domaines variés. Dans [3], le lecteur pourra trouver une bonne introduction à SysML et à ses nouveaux diagrammes que sont les diagrammes de spécifications et les diagrammes paramétriques. SysML permet de modéliser aussi bien l'architecture du système que son comportement, il réutilise les diagrammes de classe et d'objet d'UML, tout en adaptant la sémantique en utilisant la dénomination de « bloc » pour tout type d'entité (abstraite ou instanciée). Les nouveaux diagrammes ajoutent de nouvelles possibilités comme la captation des exigences ou l'ouverture à une structuration proche des EFFBD (Extended Functional Flow Block Diagram). En outre les diagrammes paramétriques permettent de recenser et structurer les équations reliant les paramètres physiques du système. Dans [4], les auteurs décrivent les origines de ces diagrammes et leurs liens avec les COB (Composable OBject), ils démontrent alors comment SysML pourra être utilisé pour nourrir des moteurs de simulation. En particulier, ils présentent une application à l'outil d'analyse et de résolution d'équation XaiTools (X-Analysis Integration Toolkit du Georgia Institute of Technology).
UML et SysML sont exploitables pour la conception de système. L'approche OO utilisée ainsi que la modélisation hiérarchique et les possibilités de composition semblent être idéales pour traiter des problèmes pluri-technologiques. De plus, la notation graphique de ces langages permet d'accélérer la compréhension des modèles et permet de les rendre rapidement accessible à des néophytes. Enfin le dernier avantage que nous pouvons mentionner pour ces langages est la présence sur le marché de nombreux outils logiciels intégrant ces spécifications. En particulier, bien que SysML soit un langage finalisé récemment, de nombreux outils UML l'implémentent déjà.

2.2 UML et SysML pour l'analyse de la SdF

Devant l'intérêt croissant des ingénieurs système pour UML, les analystes de la SdF ont à leur tour cherché à exploiter ce langage. Ces recherches ont mené aux résultats suivants dont l'adaptation à SysML est un de nos axes de réflexion.
L'intégration d'exigences de SdF ou l'analyse des critères SdF des systèmes à partir de modèles UML posent de nombreux problèmes de construction et de modélisation d'informations, [5] décrit la problématique de la modélisation de la SdF dans le domaine informatique. Les auteurs font la remarque que les programmeurs ne maîtrisent pas les techniques d'estimation de la SdF. Par conséquent, ils ont imaginé une méthode pour réduire le fossé entre les méthodes des programmeurs et celle conçues pour l'analyse. Ils proposent de définir un nouvel ensemble de stéréotypes UML, permettant de décrire le comportement dysfonctionnel du système. Ce comportement est ajouté au modèle fonctionnel du système de sorte que le modèle résultant présente conjointement les deux facettes du système. Pour exploiter ce modèle construit manuellement, les auteurs proposent un outils utilisant le langage XML pour extraire automatiquement les paramètres nécessaires afin de lancer une analyse du système avec l'outil SHARPE (Symbolic Hierarchical Automated Reliability and Performance Evaluator développé par Duke University). Ceci leur permet par la suite d'obtenir et d'analyser de façon classique des arbres de défaillances et des modèles utilisant les chaînes de Markov. Pour illustrer leur propos, leur méthode est testée sur une architecture client/serveur de type 2/3.
Dans [6], Zarras et Issarny avaient déjà fait utilisation de stéréotypes pour la saisie formatée de données dysfonctionnelles. De plus, ils ont utilisé le langage OCL (Object Constraint Language) pour saisir les spécifications fonctionnelles du système. Ces deux travaux ([5],[6]) montrent qu'UML peut être utilisé afin de considérer les aspects dysfonctionnels et les exigences de SdF. Une spécification et une étude en UML peuvent donc servir de point de départ pour l'analyse SdF d'un système. On voit également que les résultats de cette analyse pourront être répercutés et exprimés dans le même modèle que celui utilisé par le concepteur initial. Cependant, les deux méthodes présentées dans ces publications sont dédiées chacune à un outil d'analyse particulier. De plus, elles font chacune appel à un ensemble de stéréotypes UML qui leur est propre. Ceci rend ces solutions très peu généralisables et exploitables telles quelles par le reste de la communauté. Enfin, on peut noter que la construction du modèle dysfonctionnel est réalisée de façon empirique par l'utilisateur et ne découle absolument pas de l'analyse du premier modèle UML.
Dans [7], Zarras emploie une approche légèrement différente. Le modèle dysfonctionnel est construit de la même façon cependant son exploitation est différente. Il ne s'agit plus de simplement mettre en forme les données pour un outil spécifique, mais de générer automatiquement des modèles de SdF, tels que des arbres de défaillances et des chaînes de Markov. Néanmoins, les applications considérées restent uniquement tournées vers des systèmes informatiques et supposent que l'utilisateur est capable de décrire lui-même l'ensemble du comportement dysfonctionnel du système. A l'inverse, d'autres travaux exploitent les modèles UML pour déduire ce comportement. Dans [8], l'approche est fondamentalement différente. L'analyse d'un robot de télé-échographie y est présentée. Dans ce cas, le modèle n'est pas construit pour représenter les défaillances du système mais pour les déduire grâce à l'emploi d'une méthode d'analyse des risques. Pour cela, les auteurs ont défini un ensemble de modèles d'erreurs pouvant affecter les messages rencontrés dans les diagrammes de séquences UML (présent sous la même forme en SysML). La méthode proposée se compose de deux phases, la première est de modéliser la tâche ou la mission que l'on souhaite analyser. Il est à noter que les interventions humaines peuvent être prises en compte. La deuxième étape consiste à utiliser les modèles d'erreur dans une analyse de type AMDEC pour ainsi déduire les risques encourus.
Ces multiples approches prouvent que UML est adapté pour l'expression ou la déduction d'informations sur le comportement dysfonctionnel des systèmes. Néanmoins l'utilisation massive de stéréotypes rend ces travaux peu réutilisables, car dirigés par des besoins spécifiques et souvent particuliers aux cas d'étude ou à des habitudes personnelles. En effet, les principes de construction du modèle et les informations modélisées sont conditionnés par l'analyse que souhaite faire l'utilisateur. De plus, pour être exploitables les modèles utilisés sont souvent régis par des règles strictes et difficiles à se réapproprier. Pour se montrer plus universel, on doit chercher à raisonner à partir de modèles directement issus de la modélisation de l'analyse fonctionnelle. Dans de nombreux cas, l'ajout direct de la partie dysfonctionnelle au modèle peut rendre ce dernier plus difficile à appréhender et améliorer. Dans tous ces travaux précédents, on remarque que malheureusement les modèles UML sont construits spécialement pour la réalisation d'analyses SdF. Il s'agit dans ces travaux non pas de réutiliser les modèles des concepteurs, mais de se servir du même langage de modélisation. La simple utilisation d'UML ne résout donc pas le problème de l'intégration des analyses SdF au processus de conception.
L'utilisation de SysML permet de réduire certains des inconvénients rencontrés dans les travaux précédents par le biais, entre autres, de l'utilisation des nouveaux diagrammes. Par exemple, les diagrammes de spécification permettent d'exprimer naturellement les exigences et de les allouer aux différents composants sans utiliser des règles OCL comme en [6]. De plus la modélisation des aspects de fiabilité est un objectif déclaré de SysML. Les diagrammes paramétriques peuvent par exemple correspondre à des calculs de taux de défaillances pour un composant considéré. En combinant ces diagrammes avec un solveur, comme décrit dans [4], il deviendra possible de déduire des indicateurs SdF comme un taux de défaillance global du système. Dans la section suivante, nous allons décrire notre solution pour la réalisation d'une analyse SdF s'interfacant au mieux avec le processus de conception.

3. Intégrer l'analyse de la Sûreté de Fonctionnement au processus de conception

Comme nous venons de le voir, les travaux précédents concernant l'utilisation d'UML pour réaliser l'analyse de la SdF au plus près des préoccupations des concepteurs, sont certes perfectibles mais démontrent un fort potentiel pour l'utilisation d'UML ou SysML en SdF. UML, et à n'en pas douter dans un futur proche SysML, seront les langages de modélisation bénéficiant de la plus large utilisation dans le domaine du développement de systèmes complexes.
Le but de nos travaux n'est pas de trouver comment modéliser les défaillances en UML et SysML, mais de trouver la façon la plus générale possible pour exploiter ces modèles, afin d'obtenir de nouvelles informations sur la fiabilité du systèmes. Au final, on souhaite obtenir le plus rapidement possible des critères caractérisant la SdF du système, permettant de prendre une décision au niveau de la conception. Un des désavantages majeurs des analyses de SdF est qu'elles compliquent et ralentissent le processus de conception. C'est pourquoi notre ambition est de créer un logiciel capable de fournir au designer un maximum d'informations sur la fiabilité du système sans qu'il soit un expert de SdF. Nous proposons pour cela de suivre une méthode déductive et itérative en trois étapes :

- Déduction du comportement dysfonctionnel par analyse du modèle du concepteur
- Construction du modèle dysfonctionnel
- Analyse qualitative et quantitative du modèle obtenu.

Chacune de ces étapes nécessite des méthodes ou modèles différents. Notre rôle est donc d'optimiser les liens qui existent entre ces différentes activités et modèles. C'est pourquoi, nous développons actuellement des outils pour conduire automatiquement des analyses SdF à partir de modèles créés par les concepteurs en UML ou SysML. Le but de chaque étape est d'apporter de nouvelles connaissances sur le comportement du système ou de préparer automatiquement des modèles pour les outils d'analyses existant. La figure 1 représente un exemple possible d'enchaînement des étapes de la méthode.

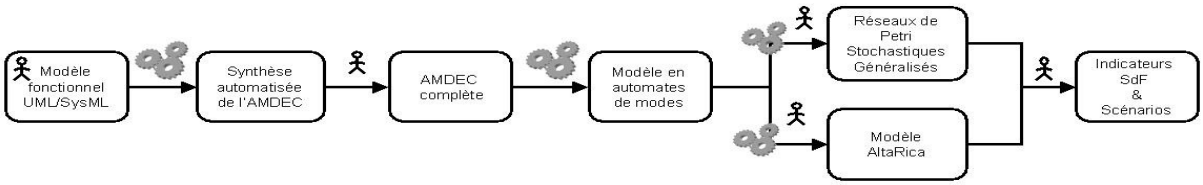

Figure 1. Du modèle fonctionnel aux indicateurs de SdF

A partir du modèle fonctionnel rédigé par le concepteur, la première étape consiste à l'établissement de l'AMDEC assistée par ordinateur. Cette AMDEC est réalisée sur la base de l'analyse des modèles UML ou SysML, en s'appuyant plus particulièrement sur les diagrammes de séquences. A l'inverse de [5], [6] et [7], nous ne supposons pas connaître le comportement dysfonctionnel des composants. Nous nous plaçons donc dans le cas où l'analyse SdF du système doit être réalisée entièrement. On suppose que l'utilisateur possède au mieux une base de données constituée à partir du retour d'expérience sur la vie de tout ou partie des composants employés. Le rôle de cette étape est donc de recenser les défaillances pouvant toucher le système. Ce point de l'étude est donc critique pour le reste du déroulement de l'analyse SdF, car il fournira les éléments de base du raisonnement à l'échelle du système. Dans la section 4, nous présentons comment accélérer la réalisation de l'AMDEC et rendre cette analyse exploitable pour accélérer et intégrer l'étude de la SdF.
L'obtention de scénarios de défaillance ainsi que le calcul d'indicateurs de fiabilité nécessite l'utilisation de modèles et langages formels (RDPSG, AltaRica, Figaro, BDMP). La construction de ces derniers nécessite une expertise. Nous envisageons donc de réaliser des outils d'aide à la création de ces modèles s'interfaçant avec les outils de traitement actuels, a l'instar de l'outil OPALE permettant de générer le modèle de fiabilité et de disponibilité de réseaux électriques [9]. Dans l'exemple de la Figure 1, nous proposons de construire un modèle en automate de mode [10] à partir des informations recueillies dans l'AMDEC. Nous envisageons l'utilisation de ce formalisme pour sa structure organisée autour d'états gérant les flux du système, avec laquelle il est facile de représenter le comportement dysfonctionnel que va prendre un composant. Pour chaque composant identifié dans l'AMDEC il est possible d'exprimer ses états fonctionnels et dysfonctionnels ainsi que le traitement des flux reçus et émis (flux d'énergie, de commandes, de contrôle) dans ces états donnés. Le modèle obtenu permet de mettre en évidence la propagation des flux et donc potentiellement des erreurs portées par ces derniers. Ce modèle a ensuite l'avantage d'être aisément transposable vers des modèles bénéficiant de plate-formes outillées tels que les modèles en AltaRica et les RDPSG. En effet, dans ces deux formalismes on peut retrouver ou adopter une représentation état/flux mettant en avant la transmission des informations et de l'énergie à travers le système. Cette partie de la méthode est encore à développer.
La dernière étape de la démarche consiste simplement à utiliser les outils existants pour réaliser l'analyse des modèles obtenus. On peut citer pour l'emploi d'AltaRica le logiciel OCAS module de Cecilia Workshop, pour les RDPSG, Jagrif équipé de Moca RP et pour Figaro le logiciel KB3. Il est possible alors d'obtenir des indicateurs SdF tels qu'un taux de défaillance global ou un temps moyen entre défaillances. Il est également possible d'obtenir des arbres de défaillances donnant les scénarios de défaillance et permettant également le calcul de taux de défaillance.

4. Synthèse automatique d'AMDEC à partie de modèles fonctionnels en UML/SysML

L'AMDEC est une méthode standard pour l'analyse des risques et la détection de défaillances potentielles, utilisable très tôt dans le processus de développement [11]. C'est une méthode inductive et qualitative bien connue, qui propose de détailler le système composant par composant. Pour chacun d'entre eux, les analystes recherchent leurs modes de défaillances, les effets de ces derniers ainsi que leurs causes. Ensuite dans le but de repérer les points faibles de la structure, on affecte une cotation en terme de probabilité d'apparition et de gravité à chacun des risques identifiés. Dans cette partie nous allons présenter les travaux existants pour la gestion des AMDEC. Ensuite, nous fournirons notre solution exploitant les diagrammes UML/SysML ainsi qu'une base de données regroupant le retour d'expérience. Nous montrerons, à travers ces deux sections, en quelle mesure notre solution est novatrice.

4.1 Etat de l'art sur la gestion automatisée des AMDEC

L'AMDEC est une méthode d'analyse qui bénéficie de nos jours d'un important retour sur utilisation. Les nombreuses années de pratique de cette méthode ont permis à de très nombreux chercheurs et utilisateurs d'identifier les entraves à sa mise en œuvre efficace. De nombreuses contraintes organisationnelles et opérationnelles éloignent bien souvent les ingénieurs des bénéfices qu'ils souhaitaient tirer de l'étude. L'AMDEC est en effet perçue par la majorité de ses utilisateurs comme une analyse longue à mettre en œuvre, peu flexible, fastidieuse et souvent très propice aux erreurs. Pour de très nombreuses personnes la réalisation d'AMDEC relève uniquement d'un impératif administratif et contractuel visant à satisfaire la hiérarchie ou le client. Les industriels expriment également leurs difficultés à rendre l'AMDEC opérationnelle, c'est à dire à lier les résultats de l'analyse au déclenchement d'actions correctives et de reprise de la conception. L'AMDEC perd ainsi son pouvoir d'aiguillage dans le processus de conception. La lourdeur de la méthode semble avoir plusieurs origines, ce sont d'une part les difficultés à réunir régulièrement les personnes compétentes pour la réalisation et d'autre par le fort volume d'information à produire, présen-

ter et comprendre. En effet, l'établissement d'une AMDEC doit se faire en présence des experts de chaque domaine ou partie du système ou processus étudié. L'étude a pour résultat un long tableau sous forme papier, qu'il est bien souvent difficile d'appréhender eu égard à l'importance du nombre de lignes crées dans une étude complète. Enfin, ces différentes contraintes ont pour effet de retarder les AMDEC de sorte qu'elles se trouvent réalisées bien trop tard dans le processus de conception, annihilant ainsi une des principales missions de l'AMDEC : identifier au plus tôt dans le cycle de développement les risques et défaillances probables du système. Une dernière critique relevée sur l'AMDEC est l'impossibilité de mettre en évidence les effets de défaillances multiples. Il a été démontré à de nombreuses reprises que les pannes combinées ont bien souvent un comportement bien différent de la simple addition des effets des défaillances particulières.
Toutes ces difficultés entrent en contradiction avec les bénéfices attendus des AMDEC. Cette méthode est née d'attentes très précises en matière d'analyse et de conception. Il a été vérifié qu'elle permet une recherche exhaustive des risques encourus par le système. Elle permet alors de détecter très tôt dans la conception les points sensibles sur lesquels focaliser toutes les attentions. Son système de cotations fait ressortir des priorités entre les risques, ce qui permettra aux concepteurs de justifier la non-correction de défaillances jugées mineures par leur faible dangerosité ou par leur faible probabilité d'apparition. Le traitement systématique des risques liés au système peut alors être envisagé, ceci représente une étape importante vers une conception de systèmes sûrs de fonctionnement et non dangereux pour l'utilisateur final. Le principal atout de l'AMDEC est alors de pouvoir déduire très tôt le comportement dysfonctionnel du système à partir de sa description fonctionnelle. La seule étape nécessaire qui précède l'AMDEC étant une simple analyse fonctionnelle. L'AMDEC est aussi très appréciée pour la constitution de base de données concernant l'ensemble des composants rencontrés au cours des études. Le recensement systématique des modes de défaillance et de leurs cotations permet de constituer une base importante pour utiliser le retour d'expérience dans les nouvelles créations. Le format de présentation est très facile à réorganiser en structure de données accélérant les travaux futurs.
Pour toutes ces raisons les travaux concernant les AMDEC restent très nombreux. Le but de ces travaux est de trouver les moyens de combattre les entraves à l'établissement efficace d'AMDEC. Les apports de ce type d'analyse sont en effet suffisamment importants pour justifier l'emploi de nouvelles techniques pour accélérer et rendre plus robuste la réalisation d'AMDEC. Pour cela les chercheurs disposent de deux leviers pour rendre optimum ce travail. On peut agir sur l'aspect organisationnel et sur l'ajout d'un support logiciel dans la conduite des analyses. Bassetto [12] fort d'une expérience menée à grande échelle fournit quelques conseils d'organisation pour impliquer les ingénieurs responsables des AMDEC et améliorer leur perception sur l'efficacité de la méthode:

- Définir l'objet et les limites de l'analyse
- Constituer une équipe réunissant spécialistes et utilisateurs du système analyser
- Formation des participants à la méthode
- Régularité des rencontres
- Obtenir l'adhésion du management et des équipes opérationnelles en faisant valoir leur intérêt dans la méthode.

De plus, l'intégration de l'AMDEC ne peut se faire sans un support logiciel adapté. Dans ce sens de nombreux chercheurs se sont penchés sur l'aide que pouvait apporter l'outil informatique à la création d'AMDEC. L'utilisation de logiciel peut être variée dans le cas de l'AMDEC :

- Maintien de la base de données [12]
- Aide à la saisie du tableau
- Synthèse automatique de parties du tableau [13], [14]
- Réorganisation et mise en évidence des éléments importants [15]
- Aide à l'utilisation et à la création de typologies [11], [12]
- Génération de modèle à partir de l'AMDEC [16].

La génération automatique, partielle ou complète, est donc l'objet de multiples travaux. Certains se focalisent sur l'exploitation d'une Base de Données et la pérennisation des résultats, quand d'autres mettent l'accent sur la génération automatique d'information par des simulations de modèles. Dans [17], les auteurs présentent l'outil AutoSteve d'aide à la conception de systèmes électroniques complexes à destination de l'automobile. L'outil réalise principalement une création automatisée d'AMDEC. Les auteurs décrivent une méthode de génération d'AMDEC de premier niveau (mode de défaillance simple), puis une méthode pour la recherche des défaillances multiples dont le souci est de contenir l'explosion combinatoire par des moyens « d'élagage » de l'arbre des scenarii de défaillance. Ainsi une méthode et des règles de sélection des risques significatifs sont utilisées. La génération d'AMDEC simple est réalisée dans AutoSteve par simulation qualitative d'un modèle de la carte électronique, réunissant la partie fonctionnelle et la partie dysfonctionnelle de chaque composant. La simulation peut donc être déroulée sur le modèle sans connaître précisément chaque composant mais en se cantonnant à la connaissance de sa fonction.
Dans [16], les auteurs développent une approche cherchant à rester généraliste et applicable à tout type de technologie. Le modèle fonctionnel du système est saisi sous Matlab Simulink, puis enrichi par une série d'arbre de défaillances créés automatiquement à partir de la saisie du caractère dysfonctionnelle des composants. L'AMDEC est ensuite réalisée par l'étude des arbres de défaillances. Cette méthode permet à partir de la connaissance des défaillances particulières de chacun des composants de déduire l'effet des défaillances à l'échelle du système et de considérer les défaillances simultanées.
Pour ces deux précédents travaux, le pré-requis à la création automatisée de L'AMDEC est la connaissance et la modélisation des défaillances particulières de chaque composant. L'apport réel de telles méthodes dans la mise en place de l'AMDEC est donc le calcul des effets des défaillances à l'échelle globale du système. L'intérêt d'une telle AMDEC n'est alors plus la découverte et le recensement des défaillances individuelles, mais leur sélection au regard de leurs conséquences ainsi que la recherche de défaillances multiples significatives.
Enfin pour clore cet état de l'art on peut mentionner les travaux cherchant à réduire l'ambiguïté des cotations et du vocabulaire employé dans les AMDEC. [18] propose un emploi de la logique floue permettant de transcrire de façon plus explicite et en langage naturel les cotations des risques. La méthode employée cherche, au moyen de règles d'inférences, à être plus précise sur la qualification de risques ayant un indicateur RPN (Risk Priority Number) similaire mais un vecteur de risques différents. Dans ce travail les cotations des paramètres de fréquence, gravité et détectabilité sont notés de façon classique, l'emploi de la logique floue permet en sortie de caractériser le risque dans le langage naturel. La méthode a été utilisée sur de nombreux problèmes, comme explicité dans [19] qui applique la méthode à un moteur diesel. Une variante est également disponible, qui permet de s'affranchir des cotations chiffrées et de procéder dès le début en langage naturel.

4.2 les points sensibles pour l'automatisation de la création d'AMDEC

L'étude proposée au paragraphe précédent nous a permis d'identifier les points cruciaux de la génération automatique d'AMDEC. Nous en avons dégagé deux principaux. Ce sont d'une part le modèle à partir duquel est réalisée l'analyse et d'autre part la base de retour d'expérience concernant les comportements dysfonctionnels.
En premier lieu l'AMDEC doit se « nourrir » d'une bonne analyse fonctionnelle du système. En effet il est indispensable de connaître parfaitement chaque fonction ainsi que l'ensemble des composants qui les réalisent. Le mode de fonctionnement de chacun d'entre eux doit être identifié et déclaré pour permettre la réalisation de l'AMDEC. Il faut connaître le mieux possible les relations entre chacun d'entre eux afin d'être capable de tirer les conclusions nécessaires pour la connaissance des effets de défaillance. Pour capturer ce comportement fonctionnel un très large éventail de méthodes, souvent spécialisées par technologie, existe. Actuellement les méthodes utilisées pour la gestion automatique d'AMDEC s'appuient sur des modèles fonctionnels enrichis par des informations sur le caractère dysfonctionnel des composants [13], [17]. Cet état de fait nous semble dans un sens paradoxal avec le but premier des AMDECs, qui est d'isoler et de détecter les risques et comportements dysfonctionnels, avec comme unique connaissance le comportement fonctionnel du système. Ces langages qu'ils

soient naturels ou techniques ont en commun la représentation des composants et leur influence les uns sur les autres jusqu'à l'obtention d'une fonction finale. Ainsi un modèle est utilisable selon nous pour la création d'AMDEC s'il permet d'isoler l'architecture du système (agencement entre composants et sous systèmes), ainsi que les flux d'énergie et d'informations entre les composants. Il est en effet utile dans le but de réaliser une étude déductive, cherchant par définition à déterminer un état final en fonction d'une situation donnée, d'utiliser une représentation permettant d'isoler facilement la propagation des flux trahissant la réalisation des fonctions. Pour nous, les caractéristiques minimales du langage de modélisation fonctionnel doivent donc être les suivantes :

- représentation architecturale
- représentation hiérarchique
- représentation comportementale.

Ces aspects nous pouvons les retrouver dans le langage UML/SysML, dont les diagrammes permettent une représentation claire des architectures et composition des systèmes par les diagrammes d'objet et de déploiement. UML/SysML met à disposition une vue des flux transmis au travers du système en identifiant via les messages des diagrammes de séquences toutes les interactions existantes entre les objets. Enfin la possibilité de classifier les objets donne à ce langage une grande capacité pour fournir une information facilement réutilisable et dont le contenu sera uniforme. Ce dernier ouvre une perspective vers la classification et l'enregistrement des composants dans des structures de base de données particulièrement adaptées à la conduite et à la pérennisation des AMDEC.
L'emploi d'une Base de Données (BdD) est le second point clef pour une AMDEC. Pour [12] la constitution d'une BdD est indispensable pour espérer créer automatiquement une AMDEC : « si chaque composante des risques possède ses typologies, la génération automatique de risques peut être envisagée ». La constitution de la BdD intervient en amont et en aval de l'AMDEC. La condition sine qua non pour l'automatisation d'une AMDEC est la connaissance pour chaque composant étudié de ses modes de défaillance, de son comportement fonctionnel et dysfonctionnel. Une fois l'AMDEC d'un système réalisée et enrichie par ses utilisateurs, la BdD d'origine peut être enrichie à son tour à partir de la nouvelle expérience obtenue de l'AMDEC. La constitution et l'exploitation d'une BdD dans le cadre d'une AMDEC nécessite la mise en place d'un vocabulaire fixé, spécifié et diffusé. Ce point délicat revient à fixer une typologie dans la création des AMDEC. Cet aspect a été ciblé par de nombreux auteurs comme [12] et a fait l'objet de développement particuliers comme [11]. Les typologies ont pour but de décrire dans un vocabulaire spécifié les différents éléments présents dans les AMDEC. Ces éléments sont dépendants du type de technologie étudiée, et chaque « grand domaine » technologique doit pouvoir se doter de ses typologies propres. Les éléments devant bénéficier d'une typologie sont a priori les composants, les fonctions, les modes de défaillances, les effets, les causes, les moyens de détection et les actions correctives.
La difficulté, pour être capable d'interpréter le modèle du concepteur, est de pourvoir faire concorder le vocabulaire du concepteur et la base de typologie. Il est donc préférable d'établir cette typologie en fonction des habitudes des développeurs, ainsi que de leur offrir la possibilité de faire évoluer et de compléter la BdD qui sera utilisée pour l'analyse SdF.

4.3 Utiliser UML/SysML pour la synthèse automatique d'AMDEC

Nous présentons dans cette section notre solution pour la création automatisée d'AMDEC. Cette solution s'attache à respecter nos remarques exposées précédemment sur le type de modèle analysable et l'apport d'une BdD bien formée. Notre but est donc de rendre opérationnelle l'AMDEC, avec pour ambition de se servir de cette méthode pour initier une démarche globale d'analyse de la SdF. L'utilisation d'une AMDEC automatisée est dans notre cas novatrice car l'automatisation porte sur la déduction des défaillances élémentaires. Ceci permet de réaliser une utilisation également nouvelle des diagrammes UML et SysML où il ne s'agit plus de simplement saisir un comportement dysfonctionnel, mais de pouvoir le déduire du modèle originel.

4.3.1 Modèle exploité et première solution réalisée

Comme décrit dans la seconde section, l'utilisation d'UML et SysML semble être la meilleure approche pour définir une méthode s'interfacant au mieux avec les problématiques de conception. De plus, ces langages possèdent les caractéristiques requises pour être analysés par une AMDEC, que nous avons énoncées dans le paragraphe 4.2.1. La première solution de constitution automatique d'AMDEC que nous avons réalisée exploite principalement les diagrammes de séquences, que l'on retrouve de façon identique dans les deux spécifications de l'OMG. Traditionnellement les AMDEC sont construites à partir des informations issues des analyses fonctionnelles. Dans une analyse fonctionnelle on isole différents éléments, on recherche notamment les différentes phases de vie du système. Celles-ci se retrouvent dans les diagrammes de séquences, car ces derniers décrivent les cas d'utilisation dans toutes les phases de vie. On recherche également les éléments extérieurs et leurs relations avec le système. Ces éléments extérieurs seront identifiés dans les diagrammes de séquences via les « acteurs ». Enfin l'analyse fonctionnelle permet d'identifier toutes les fonctions du système, or ce sont ces fonctions qui sont décrites par chaque cas d'utilisation, que représentent les diagrammes de séquences.
Dans la première version de l'algorithme de création d'AMDEC, on ne fait pas encore appel à la BdD des comportements dysfonctionnels. Cet algorithme permet de garantir l'exhaustivité de la recherche de composants et de défaillances. Le résultat de l'algorithme est un tableau AMDEC dans lequel figurent tous les systèmes à étudier et tous les modes de défaillances à envisager pour eux. Ces modes de défaillances sont les quatre modes de défaillance classiques, à savoir : la fonction dégradée, l'absence de fonction, la fonction intempestive et la perte de fonction. De plus, on suggère pour chaque risque (couple composant/mode de défaillance) de ce tableau AMDEC, l'ensemble des composants ou éléments de l'environnement pouvant être cause ou subir les effets de chaque défaillance. Cet algorithme est présenté dans les lignes qui suivent :
Entrées de l'algorithme :

- Ensemble des Acteurs nommé A
- Ensemble des Objets nommé O. Remarque : $A \cap O = \varnothing$
- Ensemble des Messages nommé M.

Les éléments de M sont des couples (x,y) où x, $y \in A \cup O$, x est l'émetteur du message et y le récepteur.

- C et E deux ensembles vides (ils recevront respectivement les entités pouvant être cause ou subir les effets de défaillances)
- L'ensemble des Modes de défaillance nommé MD.

MD = {fonction dégradée, pas de fonction, fonction intempestive, perte de fonction}
Sorties de l'algorithme :

- Tableau représentant l'AMDEC

Opérateurs :

- + est ici l'opérateur de concaténation à droite
- pour $m \in M$, m= (m[1] , m[2]).

Début de l'algorithme :
Création d'un tableau dont la première ligne comprend 9 cases indiquant, nom de l'objet ou de l'acteur, mode de défaillance, entités cause, entités affectées, gravité, occurrence, risque, moyens de détection possibles, solutions possibles.
$\forall i \in A \cup O$ et $\forall md \in MD$
$\forall m \in M$, si m[2] = i, C = C + m[1]
$\forall m \in M$, si m[1] = i, E = E + m[2]

créer une ligne = {i, md, C, E, ∅, ∅, ∅, ∅, ∅}.
Cette solution a été testée sur des diagrammes issus de [4]. Des résultats et un retour d'expérience sur son utilisation pour l'étude d'un microprocesseur sont également exposés dans [20]. Dans cet article les auteurs précisent que l'emploi de ce premier algorithme leur a permis de retrouver toutes les informations qui figuraient dans une première AMDEC faite entièrement manuellement. De plus, outre le gain de temps observé lors de la création de l'AMDEC, les suggestions apportées par la partie créée automatiquement leur a permis d'identifier des risques qui n'avaient pas été perçus lors de la première analyse. Malgré les avantages cités précédemment, cette version souffre du problème de l'explosion combinatoire du nombre de lignes du tableau fournit en sortie. De plus, les solutions proposées restent très généralistes dans l'étude du système. C'est pourquoi, nous avons décidé d'apporter une modification en réalisant une connexion de l'algorithme à une BdD. Ce travail nous permet d'obtenir une AMDEC à la fois plus précise et concise. La connexion avec la BdD intervient à l'étape de recensement des modes de défaillances envisagés pour le composant. En réalisant une reconnaissance de la typologie des types des composants, on peut automatiquement rechercher dans la base de données les modes de défaillances particuliers applicables aux composants étudiés. La partie suivante expose l'utilisation de la classification d'éléments UML et SysML pour la réalisation de la BdD.

4.3.2 Organisation et exploitation d'une base de données des comportements dysfonctionnels pour la synthèse d'AMDEC

Comme il a été indiqué en 4.2 l'utilisation d'une BdD des comportements dysfonctionnels est cruciale, car elle permet d'impliquer le retour d'expérience dans l'identification des risques. On assure ainsi une cohérence entre les différentes études et on obtient automatiquement des résultats beaucoup plus précis. La BdD contient les modes de défaillance connus pour les composants que nous utilisons. L'algorithme indique donc les défaillances connues dans la base, mais propose néanmoins les modes classiques si un composant n'est pas connu. Une fois l'AMDEC reprise par un opérateur, on pourra réintégrer dans la BdD les modes de défaillance identifiés manuellement. De même il est important de maintenir à jour la base en la modifiant si de nouveaux modes de défaillance sont identifiés en reprenant les AMDEC passées. Nous avons conçu notre base de données en nous appuyant sur le concept de classes présent en UML et extensible à SysML par les diagrammes de définition de blocs. Ce concept nous permet de travailler à de très nombreux niveaux de détail dans la définition du système. En effet, il est possible de représenter les composants avec des relations de hiérarchie au sens généralisation. Cette hiérarchie peut également être définie entre les modes de défaillance, du plus large (modes de défaillance des composants électriques) au plus précis (modes de défaillance d'un condensateur céramique de marque donnée). La figure 2 illustre l'exploitation de la base de donnée pour la création d'AMDEC.

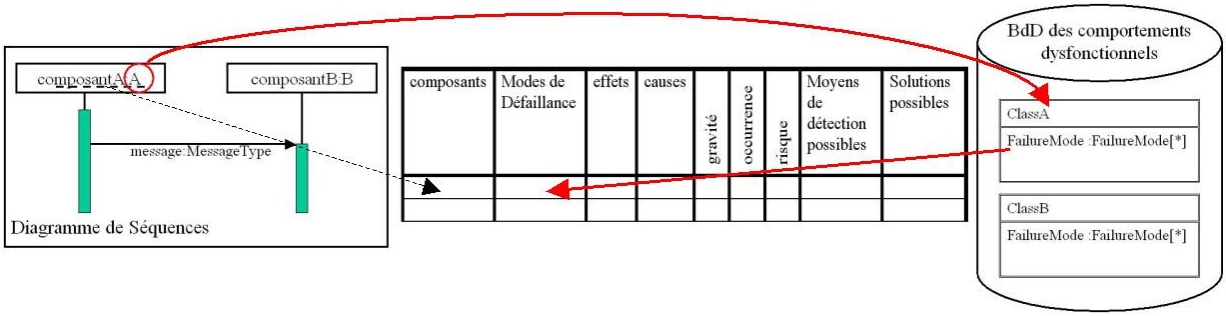

Figure 2. Utilisation de la BdD pour la création d'AMDEC

La base de données est construite par l'intermédiaire de diagrammes de définition de blocs (bdd : Block Definition Diagram). Elle est structurée autour des composants physiques. C'est à dire que les relations hiérarchiques concernent les composants eux-mêmes, on crée ainsi une arborescence de la définition la plus générale (ex : composant électronique) vers plus de spécialisation (ex : résistance). Pour chaque élément de cet arbre, on fixe dans leurs attributs les modes de défaillance auxquels ils sont sujets. Leur nom peut ainsi être récupéré pour construire la colonne modes de défaillance de l'AMDEC.
Cette technique nous permet de gagner beaucoup de temps dans l'établissement des AMDEC. On peut ainsi créer une majeure partie de l'AMDEC de façon précise tout en assurant l'exhaustivité dans la prise en compte des composants. Les figures 3,4 et 5 présentent un exemple type d'utilisation de cet algorithme sur un système anti-débordement (SAD) de cuve. L'AMDEC résultante est automatiquement partiellement remplie, on y retrouve les modes de défaillance précis de chaque composant ainsi que des suggestions sur le déroulement de la défaillance (causes, effets). Le système que nous considérons est décrit sur le bdd de la figure 3. Il est principalement constitué d'un système de contrôle du volume de la cuve. Ce système est le sujet de l'étude réalisée. Il est constitué de 2 capteurs de niveau, d'une alarme, de 2 électrovannes, 2 vannes manuelles et d'un système d'alimentation électrique.
Le comportement du système de contrôle a été décrit au travers de différents diagrammes de séquences. Ces diagrammes nous permettent d'utiliser notre technique pour la génération automatique d'AMDEC partielle. La figure 4 présente un de ces diagrammes de séquences. Le scénario décrit tous les mécanismes mis en place pour éviter le débordement de la cuve. En cas d'accident tout ou partie de ce scénario se déroule jusqu'à un retour en état sûr de l'installation.
Nous avons ensuite utilisé les algorithmes précédemment décrit, ainsi qu'une BdD décrivant les caractères dysfonctionnels connus pour des composants classiques tels que les vannes manuelles et les électrovannes. Nous avons ainsi créé automatiquement une partie importante de l'AMDEC du système de contrôle. La figure 5 présente un extrait de ce que nous générons de façon systématique. Il s'agit de la partie concernant l'étude de l'électrovanne *va*.
Cette étude nous a permis de mettre en évidence l'efficacité de la méthode développée. En effet nous avons constaté que nous retrouvions les même avantages qu'avec la première version de l'algorithme et que nous fournissions des informations beaucoup plus riches sur le système. L'utilisation de SysML pour la modélisation de ce système nous permet également de mettre en évidence certaines voies d'amélioration de notre méthode. En effet nous nous apercevons que l'utilisation d'informations complémentaires présentes dans les modèles SysML, va nous permettre d'aller encore plus loin dans la mise en évidence des points faibles et risques liés au système. En SysML les IBD (Internal Block Diagram) décrivent la structure interne des composants en termes de connections entre leurs propriétés ou entre les ports de leurs sous-composants. Ces diagrammes vont nous permettre d'analyser plus précisément les causes et effets des Modes de Défaillance. En effet si nous considérons l'IBD du système de contrôle de la figure 6, nous pouvons compléter l'AMDEC de la figure 5 en ajoutant dans la liste des composants causes de défaillances l'alimentation électrique *ae* qui fournit le courant à *va*.
L'utilisation des IBD nous semble indispensable pour poursuivre cette démarche de création d'AMDEC. En effet il apparaît qu'il n'est pas suffisant de se cantonner à la description comportementale inter composant du système pour mettre en évidence toutes les facettes du comportement dysfonctionnel. L'étude de la structure et plus particulièrement des points d'échanges de flux, d'énergie et de données est primordiale pour évaluer dans leur globalité les risques qui pèsent sur le système. Les IBD vous donc nous permettre de franchir ce pas dans l'analyse des systèmes. En effet l'étude des entités liées par les flow ports ainsi que la définition du flux passant par ces ports va nous permettre de relever de nouveaux éléments comme les problèmes d'alimentation énergétique ou le passage de fluides. Ces données très importantes ne sont pas représentées au sein des diagrammes de séquences, plus à même de représenter la partie commande des systèmes. La spécification de SysML offre la possibilité au concepteur de préciser la nature du flux circulant par les ports des composants.

L'analyse de ces flux peut également donner des indications sur le comportement futur du système. En particulier pour le système que nous étudions, nous pouvons étudier le fluide stocké dans la cuve et passant dans le système de contrôle. S'il s'agit d'eau nous pouvons dire que ce type de fluide peut geler. Ceci peut avoir un impact sur le fonctionnement du système. Une étude complète menée grâce à ce modèle SysML doit prendre en compte les états possibles des flux identifiés et doit pouvoir conduire à préciser les conditions d'emploi dans lesquelles le système n'est plus sûr, dans notre cas cela peut être les conditions de température supportables.

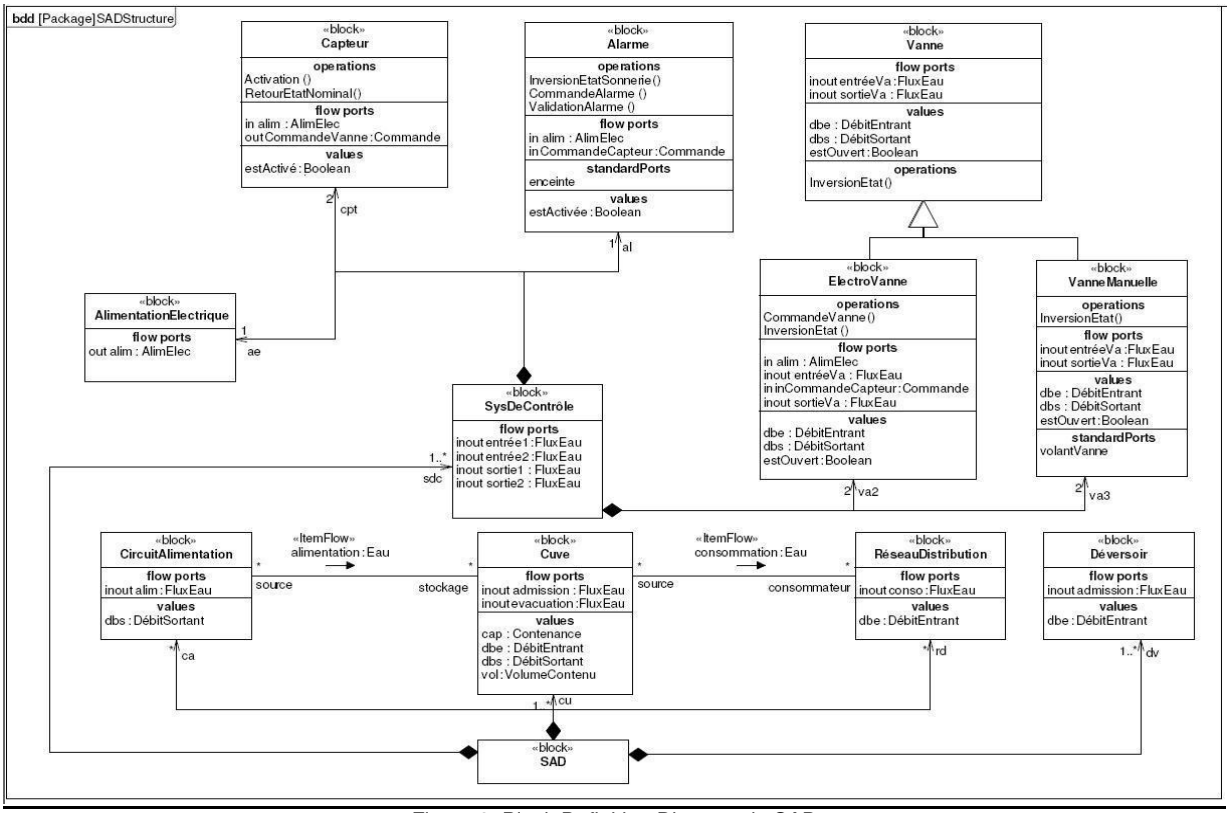

Figure 3. Block Definition Diagram du SAD

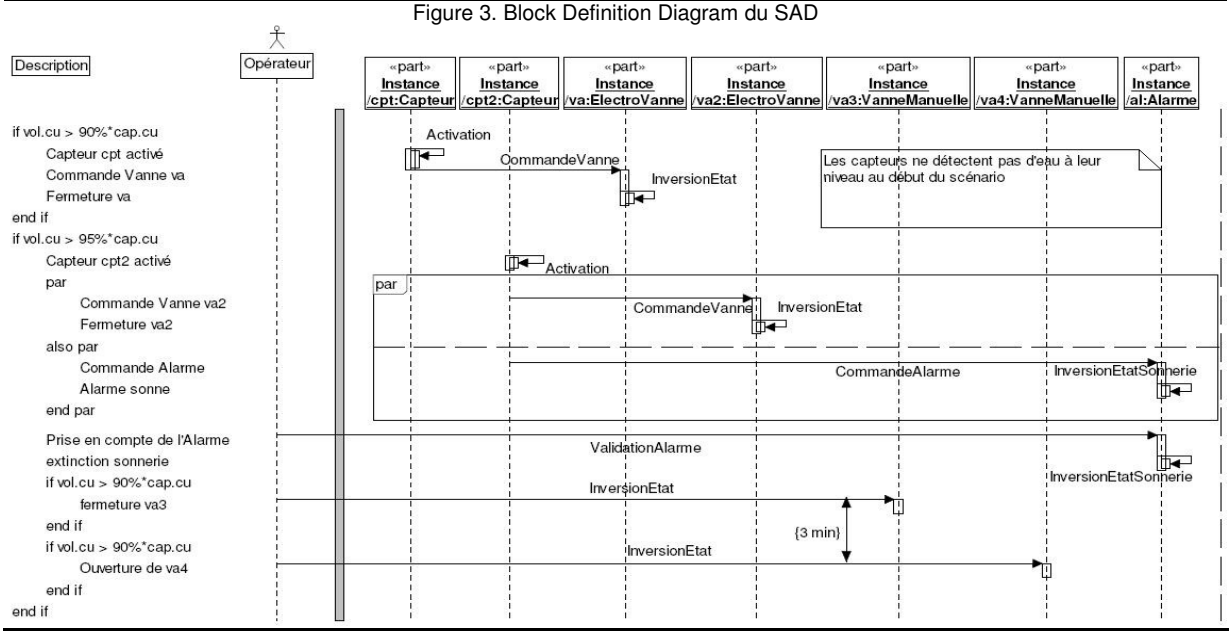

Figure 4. Diagramme de séquence du fonctionnement du système de contrôle

Composants	Modes de défaillances	Composants potentiellement affectés	Effets	Composants potentiellement source de la défaillance
va:Electrovanne	Axe bloqué ouvert	va (message : va.InversionEtat)	Fluide passe	cpt (message : cpt.CommandeVanne) va (message : va.InversionEtat)
	Axe bloqué fermé	va (message : va.InversionEtat)	Fluide bloqué	cpt (message : cpt.CommandeVanne) va (message : va.InversionEtat)
	Fuite	va (message : va.InversionEtat)	Chute de pression (vanne ouverte) Passage de fluide (vanne fermée)	cpt (message : cpt.CommandeVanne) va (message : va.InversionEtat)
	Panne Moteur	va (message : va.InversionEtat)	Vanne bloquée	cpt (message : cpt.CommandeVanne) va (message : va.InversionEtat)

Figure 5. Extrait de la partie automatiquement créée de l'AMDEC du système de contrôle

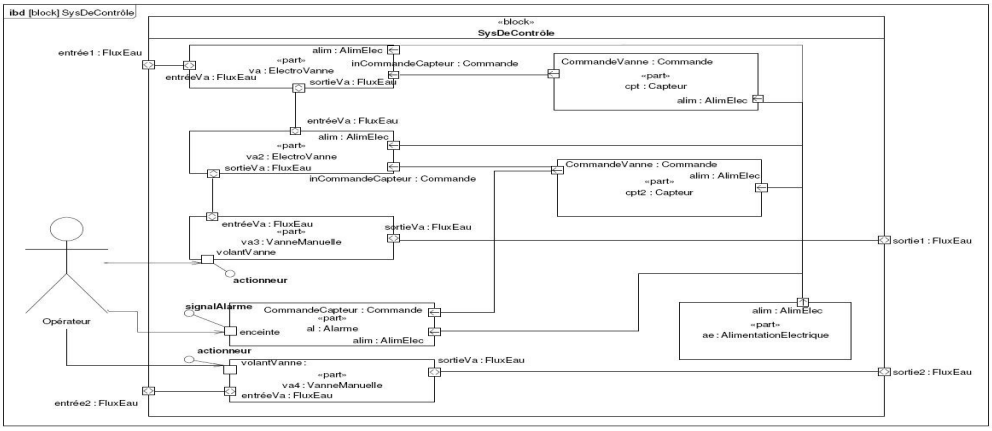

figure 6. Internal Block Diagram du système de contrôle

5. Conclusion

Dans cet article nous venons de présenter la légitimité de l'utilisation d'UML et SysML dans le cadre d'analyses de SdF. Nous avons introduit notre propre approche pour l'enchaînement des étapes nécessaires à l'analyse d'un système en cours de conception. Mettant en avant l'utilité de la réalisation d'une AMDEC, nous avons souligné les points cruciaux pour l'automatisation de cette analyse. Puis nous avons présenté notre solution pour la synthèse d'AMDEC et ses principaux mécanismes. Nous sommes en cours de développement du logiciel devant intégrer chacun des concepts présentés dans cet article. L'algorithme de création de l'AMDEC exploite les fichiers XML créés par les outils UML/SysML et normalisé par l'OMG. L'analyse se fait avec un parseur permettant d'isoler dans les fichiers du modèle et de la base de données les informations nécessaires à la création de l'AMDEC. L'ambition est d'intégrer tout le traitement de l'AMDEC dans un seul logiciel. Ce logiciel doit permettre de charger le modèle décrit dans un fichier XMI et de dérouler l'algorithme en relation avec la BdD. Nous souhaitons également intégrer la gestion de la base à cette solution. Dans la suite de nos travaux nous nous attacherons à développer des parseurs permettant de créer les modèles en automates de modes et RDPSG. De plus, les nouveautés de SysML laissent entrevoir de grandes possibilités pour la réalisation d'analyse SdF. Nous spécifions donc actuellement une nouvelle version de la construction d'AMDEC exploitant les nouveaux diagrammes SysML. Nous espérons ensuite réaliser l'étude complète grâce à des modèles SysML et quelques outils de simulation. Nous souhaitons par exemple lier le calcul du taux de défaillance de chaque mode, à sa déclaration dans la BdD des composants. Pour cela il est possible d'associer un diagramme paramétrique à chaque mode de défaillance déclaré. L'équation pour le calcul de ce taux peut être tirée des normes applicables à la technologie étudiée. Les diagrammes paramétriques indiquent la formule du calcul et indiquent quels paramètres structuraux et environnementaux sont utilisés pour le calcul.

Remerciements

Nous tenons à remercier les partenaires du projet Capthom. Ce travail a été réalisé avec le soutien financier de la Région Centre et du Ministère de l'Industrie dans le cadre du projet Capthom du pole S^2E^2, www.s2e2.fr.

Références

[1] OMG, OMG systems modeling language (OMG SysML) V1.0. 1er septembre 2007
[2] Willard, B., UML for systems engineering. Computer standard and interfaces, 2006, 29:69-81
[3] Hause, M., The SysML modeling language. 15th European systems engineering conference, septembre 2006.
[4] Peak, R.S. Burkhart, R.M. Friedenthal, S.A. Wilson, M.W. Bajaj, M. Kim, I., Simulation-based design using SysML Part 1: a parametric primer & Simulation-based design using SysML Part 2: celebrating diversity by example, INCOSE intern. Symp., San Diego, 4 mai 2007
[5] Leangsukun, C. Song, H. Shen, L., Reliability modeling using UML. International conference on software engineering research and practise, Las Vegas, Juin 2003.
[6] Zarras, A. & Issarny, V., A UML-based framework for assessing the reliability of software systems. Proc. Intern. Symp. ICSE worshop on describing software architecture with UML, Toronto, mat 2001. 36-46.
[7] Zarras, A. Vassiliadis, P. Issarny, V., Model-driven dependability analysis of web services. Proc. Intern. Conference on distributed objects and applications, Octobre 2004
[8] Guiochet, J. Motet, G. Baron, C. Boy, G., Toward a human-centered UML for risk analysis. In Jonhson, C.W. & Palanque, P. (eds), IFIP world computer congress, Human error, Safety and Systems Development; Proc. Intern. Symp., 2004, 177-191.
[9] Bouissou, M., Automated Dependability Analysis of Complex Systems with the KB3 Workbench: the Experience of EDF R&D, The International Conference on ENERGY and ENVIRONMENT, CIEM 2005, Bucharest, 2005.
[10] Rauzy, A., Mode Automata and their compilation into Fault tree. Reliability Engineering and System Safety, 78: 1-12.
[11] Tumer, I.Y. Stone, R.B. Bell, D.G., Requirement for a failure mode taxonomy for use in conceptual design. ICED, Stockholm, 19-21 août 2003.
[12] Basetto, S., Contribution à la qualification et à l'amélioration des moyens de production ~ Application à une usine de recherché et production de semi-conducteurs. Thèse présentée à l'Ecole Nationale Supérieure des Arts et Métiers, 28 juin 2005.
[13] Papadopoulos, Y. Parker, D. Grante, C., Automating the failure modes end effects analysis of safety critical systems. IEEE Hase 2004.
[14] Bull, D.R. Burrows, C.R. Edge, K.A. Hawkins, P.G. Woolons, D.J., A computational tool for failure modes and effects analysis of hydraulic systems. ASME annual winter meeting, Atlanta, 17-22 novembre 1996.
[15] Price, C.J., AutoSteve: automated electrical design analysis. ECAI proc. Int. symp., 2000, 721-725.
[16] Papadopoulos, Y. Parker, D. Grante, C., A method and tool support for model-based semi-automated FMEA of engineering designs. Cant, T. (ed.). 9th Australian workshop on safety related programmable systems, 2004, 47.
[17] Price, C.J. & Taylor, N.S., Automated multiple FMEA. Reliability engineering and system safety, 2002, 76: 1-10.
[18] Bowles, J.B. & Pelaez, C.E., Fuzzy logic priorization of failures in a system failure mode, effects and criticality analysis. Reliability engineering and system safety, 1995, 50:203-213.
[19] Yeh, R.H. & Hsieh M.-H., Fuzzy assessment of FMEA for sewage plant. Journal of the Chinese institute of industrial engineers,2007, 24:505-512.
[20] Belhadaoui, H. Cassier, C. Idasiak, V. Malassé, O. Aubry, J.F., Outil d'aide a la conception des systèmes mécatroniques sûrs de fonctionnement. Qualita 2007,Tanger, 20-22 mars 2007. 426-43