SÛRETE DE FONCTIONNEMENT ET ANALYSE DE PERFORMANCE

RAMS ANALYSIS TECHNICS AND PERFORMANCE ANALYSIS

Antoine RAUZY Institut de Mathématiques de Luminy 163 avenue de Luminy, Case 907 13288 Marseille CEDEX 9 téléphone +33 4 91 73 26 15 E-mail arauzy@iml.univ-mrs.fr	Zina BRIK APSYS 22 Quai Gallieni 92 158 Suresnes Cedex Téléphone +33 1 42048591 E-mail zina.brik@apsys.eads.net
Emmanuel ARBARETIER APSYS 22 Quai Gallieni 92 158 Suresnes Cedex Téléphone +33 1 42048585 E-mail emmanuel.arbaretier@apsys.eads.net	

Résumé

Cet article présente différents modèles simplifiés illustrant l'évolution progressive présentée par les disciplines de la Maîtrise du Risque, qui, peu à peu, tendent à passer d'un référentiel théorique classique dit "de la Sûreté de Fonctionnement" à la construction d'un cadre outillé et performant d' "Ingénierie Performantielle": un industriel constructeur de systèmes doit en effet de plus en plus être capable de s'engager sur la réalisation de performances, dont la spécification inclut l'ensemble des aléas susceptibles d'être rencontrés au cours du cycle de vie opérationnel du système (défaillances, accidents, sinistres, menaces diverses...).

Summary

This publication presents different simple models, which give a concrete example of progressive evolution occurring in RAMS technics and practices towards a Computer Aided framework consisting in Performance Engineering: a system procuror or manufacturer is more and more expected to give assurance of Performance satisfaction, the specification of which must absolutely include all possible risky or random events able to occure during the whole operational life of the system

1. Introduction

Les disciplines de la Sûreté de Fonctionnement ont toujours favorisé le développement de techniques consistant à évaluer des indicateurs probabilistes associés à des conditions de réalisation de mission, en insistant bien sur le fait que chaque mission évaluée correspond à un ensemble de niveaux de performance bien précis: elles peuvent être désignées sous des dénominations du genre "mission complète ou maximale" intégrant des niveaux de performances idéaux, "mission critique" correspondant au niveau le plus bas des performances acceptables, ou encore niveaux intermédiaires dégradés.

Cette approche de discrétisation assez limitée des niveaux de réalisation de mission, et plutôt conventionnelle, présente de nombreux désavantages: entre autres, ces niveaux de mission choisis arbitrairement au sein des multiples trajectoires possibles du système, rassemblent rarement le concensus de tous les acteurs industriels, et l'intégration des analyses de risque au processus de décision des concepteur semble plutôt reposer sur des considérations partielles.

Une autre possibilité envisagée, afin d'évaluer les risques associés à la réalisation des performances d'un système est:

- d'identifier toutes les occurrences possibles des risques (modes de défaillance, sinistres, avaries, ...),
- d'élaborer les modèles de simulation de performance en prenant en compte toutes les contributions et les influences possibles de ces risques sur les performances,
- de réaliser des simulations stochastiques pour obtenir les distributions statistiques des paramètres de performance de ces systèmes ainsi que les caractéristiques quantitatives associées.

Au cours de cet article, différents examples sont présentés, et utilisent le langage Altarica:

- distribution statistique de la productivité d'une plate forme offshore,
- influence des protocoles de partage de ressources,
- prise en compte des mécanismes de stockage,
- disponibilité de service d'un banc de test.

Pour chacun de ces examples, les aspects suivants sont abordés:

- méthodologie de construction du modèle: collecte des informations de conception et raffinements dysfonctionnels, afin d'adapter les équations de transfert des constituants aux mécanismes de dégradation et de dysfonctionnement,

- possibilités de réutilisation et archivage des modèles avec possibilité de constitution de bibliothèques pour faciliter l'initialisation et l'adaptation des modèles,
- typage des modèles: statique, dynamique, mécanismes de temporisation, invariants conservatifs ou répétitifs,
- capacités de traitement du langage: simulation par évènement, exploration des trajectoires de production de situations ou d'évènements redoutés,
- évaluation logique ou quantitative de prédicats grâce à des traitements du type "Monte Carlo"
- ...

Les avantages pratiques sont mis en évidence à partir de ces tâches de modélisation et de simulation:
- meilleur contrôle des avantages de confiance en relation avec la performance du système,
- identification des seuils acceptables et inacceptables,
- prévision des fonctions économiques objectifs, ou des pénalités dues à des bonus ou au contraire à des déficits de performances,
- analyses de sensibilité de ces performances, du fait des différents scénarios ou configurations...

2. Le langage Altarica

Tout d'abord, afin d'avoir une idée plus précise de ce qui est modélisé sur ces jeux d'essai, présentons le langage Altarica (Griffault 1998a & 1999b), qui supporte la modélisation de ces exemples.
Le langage Altarica convient à la description d'un très grand nombre de systèmes: il est surtout utilisé dans le cadre d'études de Fiabilité, et plus généralement de Sûreté de Fonctionnement.
Altarica permet d'analyser des systèmes critiques des points de vue fonctionnels et dysfonctionnels:
- Le point de vue fonctionnel recouvre toutes les techniques permettant de formaliser et de démontrer que le système est compatible avec ses spécifcations,
- Le point de vue dysfonctionnel inclut tous les points de vue consistant à analyser ce qui se passe lorsque se produi-t (sent) une ou plusieurs anomalies, et suivant quel processus, et avec quelle probabilité le système présentera une défaillance partielle ou globale, catalectique ou progressive.
Ce langage généralise des formalismes largement répandus comme les grafcets, les automates à états finis synchronisés, les réseaux de Pétri, ...
Il est caractérisé par:
- une description hiérarchique: le système est composé de differents sous systèmes, eux mêmes constitués d'entités plus élémentaires, pouvant aller jusqu'au composant,
- la possibilité d'associer des automates à états finis à chaque entité constitutive,
- des flux entrées et sorties,
- des évènements discrets,
- des variables associées aux entités et aux flux,
- des transitions: chaque composant possède différents états (bon fonctionnement, défaillance, état standbye...). Des occurrences d'évènements (défaillances, réparations, démarrage, reconfiguration,...) modifient ces états lorsque ces transitions sont tirées. Le temps d'activation de la transition peut être déterministe ou aléatoire, et suivre une loi de distribution donnée.

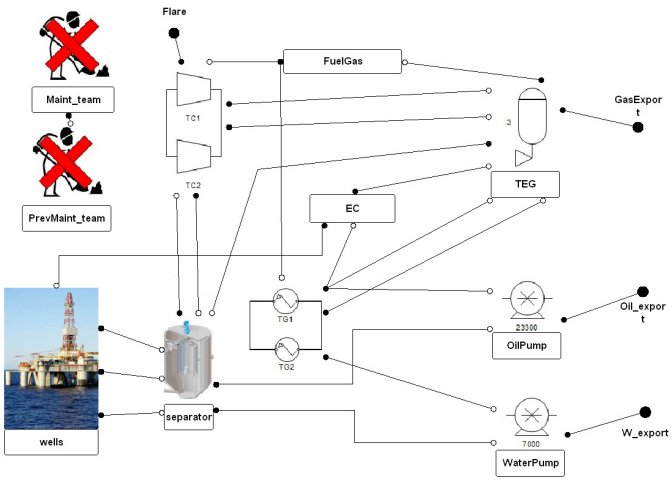
2. Les études de cas

2.1 Etudes de cas 1 : analyse de disponibilité de production d'une plateforme offshore

Cette étude de cas est particulièrement significative, dans la mesure où elle illustre parfaitement la capacité du langage à modéliser les aspects dynamiques, ainsi que la possibilité de faire des calculs prévisionnels de performance. La plateforme offshore est décrite par le schéma de la figure 1.
Ce modèle inclut également les processus de maintenance préventive et corrective représentés dans la partie supérieure gauche du schéma.
En se basant sur cette architecture et sur une compréhension des différentes fonctions de production de pétrole brut, on peut élaborer différents graphes logiques montrés en figure 2, concernant l'électro compresseur, en figure 3 concernant le turbocompresseur, et en figure 4 pour l'équipe de maintenance.
Il est intéressant d'insister sur le fait que le modèle intègre la prise en compte de l'équipe de maintenance, en incluant les délais d'intervention et de venue sur place, représentant le fait que l'équipe de maintenance n'est pas localisée sur la plateforme.

Figure 1. Modèle simplifié d'une plateforme offshore

.



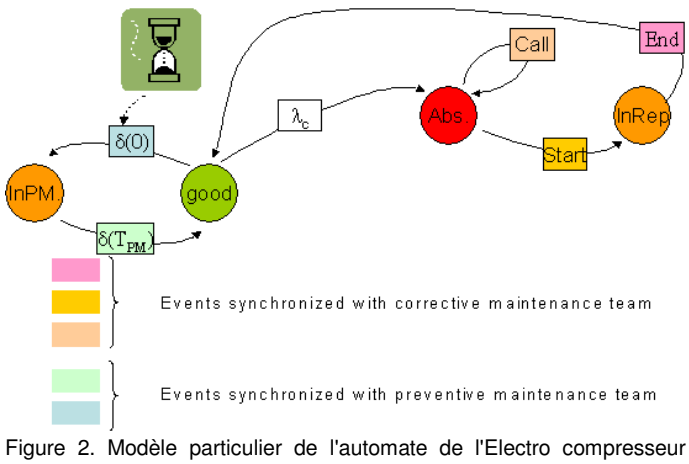


Figure 2. Modèle particulier de l'automate de l'Electro compresseur (EC).

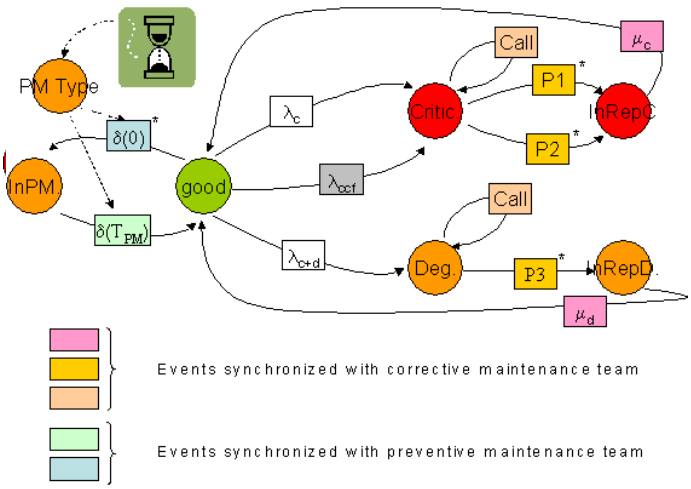


Figure 3. Modèle particulier de l'automate du Turbo compressor (TC).

Figure 4. Modèle particulier de l'équipe de maintenance

Cette représentation graphique est ensuite transformée automatiquement en langage Altarica, en fonction de l'outil utilisé. Après, il est possible de lancer une simulation de Montecarlo, afin de calculer les différentes performances de la plateforme offshore (Dutuit, 1997a & 1997b). Les résultats obtenus sont consignés dans le tableau suivant:

Table 1. performances of an oil platform (cubic meter)

Indicator	Mean	Std Dev.
Oil production	22564.3	129.8
Gas production	2.469	0.026
Water production	5221.3	73.9

Ce modèle montre bien qu'au lieu de fournir la probabilité de produire du pétrole avec un certain débit, ces simulations nous permettent de connaître les paramètres statistiques de dispersion de ce débit, compte tenu de tous les aléas.

2.2 Etudes de cas 2 : impact du stockage sur la fiabilité

Cette étude de cas analyse montre comment modéliser l'influence des phases de stockage sur la fiabilité d'un composant.

A titre d'exemple, on considère un composant, qui est stocké pendant une durée T1, avant d'être utlisé, étant données les variables suivantes:

- λs : taux de défaillance en phase de stockage
- λf : taux de défaillance en utilisation
- μ : taux de réparation en phase utilisation
- γ: probabilité de défaillance à la sollicitation.

Les tests périodiques sont réalisés avec une périodicité de (x) et satisfont un taux de couverture de (c%).

Le composant possède trois variables d'états différentes.

Les automates de la figure 5 montre les transitions possibles.

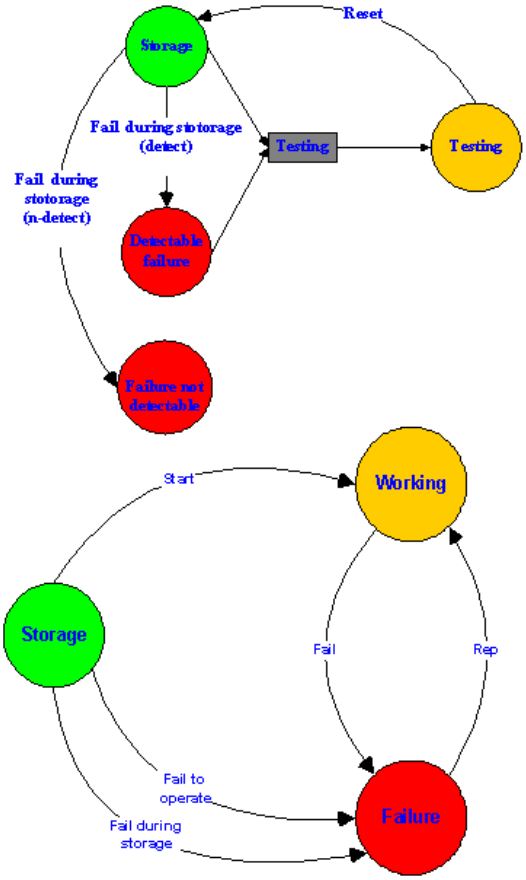


Figure 5. Transition modelling of a component's storage

.

Ces transitions peuvent être décrites par les équations suivantes:

- law <event fail_storage_detect> = exponential(c%*λs);
- law <event fail_storage_ndetect> = exponential[(100%-c%)*λs];
- law <event testing> = Dirac(x);
- law <event reset> = Dirac(ε);
- law <event start> = constant(1-γ);
- law <event fail to operate> = constant(γ);
- law <event fail during storage> = Dirac(0);
- law <event fail> =exponential(λf);
- law <event rep> =exponential(μ);
- law <event request> =Dirac(T1);

Basé sur le fait, que le modèle peut ensuite être configuré avec des données numériques, et effectuer le calcul de disponibilité de mission, en intégrant les phases de stockage, et en prenant en compte les tests périodiques pendant ce stockage.

Par exemple, avec un temps de mission de 10 000 h, on trouve une disponibilité de 98.18% avec un écart type de 0.13%.

Ce type de modèle peut être utilisé pour des composants multiples. Ainsi, il est possible de fournir une réponse rapide à la problématique concernant l'effet des phases de stockage, et des tests périodiques sur les performances d'un système.

2.3 Etudes de cas 3 : partage de ressource

Une question qui se pose souvent en maintenabilité consiste à se demander comment dimensionner les ressources nécessaires au soutien logistique d'un système, par exemple, les réparateurs ou les rechanges; pour un souci de simplicité, on se limite là à un réparateur et une pièce de rechange partagés par un seul système.

Tout d'abord, il est nécessaire de créer un modèle pour le composant: figure 6.

.

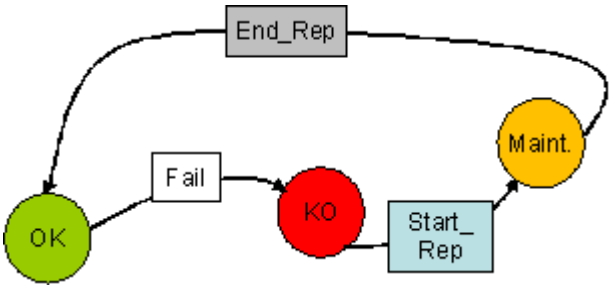


Figure 6: cycle d'évolution d'un composant

On doit créer un modèle de consommation des rechanges et de calcul de la disponibilité prenant en compte la mobilisation du personnel pour la maintenance.

Puis, on définit les synchronisations suivantes:

- Equipment_A.Start_Rep and Engineer.Start and Stock.Start
- Equipment _B.Start_Rep and Engineer.Start and Stock.Start
- Equipment _A.End_Rep and Engineer.End
- Equipment _B.End_Rep and Engineer.End

Tous ces modèles permettent de calculer la disponibilité opérationnelle exacte de chaque composant, tout en prenant en compte le fait que, de temps à autre, ils doivent attendre la disponibilité d'un ingénieur de maintenance ou l'arrivée d'une prièce de rechange.

Par exemple, avec 1 ingénieur de maintenance, et 5 pièces de rechange en début de mission, on a calculé les disponibilités suivantes:

- composant A: moyenne: 99.9937%, écart type: 0.0096%
- composant B: moyenne: 99.9886%, écart type: 0.0158%

2.4 Etudes de cas 4 : disponibilité de service d'un banc de test

The dernier exemple constitue la réponse à un problème classqiue: comment peut-on mesurer la disponibilité de service d'un banc de test ?

La disponibilité de service est le temps pendant lequel le système est disponible, pour prendre en charge une réparation d'un LRU, comparé au temps global de réparation au sein de la mission, impliquant l'utilisation du banc de test.

Cette évaluation conduit à obtenir un taux plus important que celui obtenu, lorsqu'on calcule une disponibilité classique caractérisant le software et le hardware constituant le banc de test: cette disponibilité implique que tous les éléments utilisés dans le cadre de la mission soient requis pour le calcul du chiffre de disponibilité du banc de test, ce qui est loin d'être le cas, lorsqu'il n'y a pas besoin d'utiliser le banc de test.

L'explication est que le banc de test peut être utilisé, malgré le fait que certains de ses modules soient en panne; en effet, tous ses modules ne sont pas nécessaires à la prise en charge de toutes les actions de maintenance.

Tout d'abord, les hypothèses du modèle sont à spécifier:

- Hypothèse 1: les quantités d'articles sont limitées à 5 types de composants pour le banc de test, 4 catégories d'équipements à tester avec ce banc, et 2 articles par catégorie. Dans la réalité, ce modèle a été réalisé avec 800 articles et 100 composants pour le banc de test.
- Hypothèse 2: on n'a pas modélisé de réelles files d'attente; peu importe si l'on teste A avant B ou B avant A.
- Hypothèse 3: aucune règle de priorité n'a été prise en compte, mais il serait possible de les rajouter.

Le banc de test est constitué de 5 composants: A, B, C, D et E. Pour chaque composant, les informations suivantes sont requises: MTBF, MTTR; on renseigne également le fait qu'il soit nécessaire de déconnecter le banc de test pour réparer les composants: oui, pour A,C, et E; non pour B et D.

Quatre catégories d'équipements sont à tester.

Pour chaque équipement, l'information suivante est disponible: MTBF, durée des tests, et composants du banc de test dont on a besoin pour effectuer ces tests:

- équipement 1: A, B, C
- équipement 2: A, B, D
- équipement 3: A, C, E
- équipement 4: A, D

La stratégie de test est la suivante: les équipements sont testés dans l'ordre où ils tombent en panne, mais si le test n'est pas possible du fait qu'un composant du banc de test n'est pas disponible, l'équipement est maintenu dans la file d'attente, et l'équipement suivant qui est à tester est pris en charge par le banc de test.

Cette stratégie est appliquée, jusqu'à ce qu'il n'y ait plus d'équipement testable, ou que le banc de test doive lui-même être envoyé en maintenance, et donc ne puisse pas être utilisé.

Pour pouvoir modéliser le système, on a du utiliser des fonctions de transfert virtuelles, permettant de paramétrer les combinaisons de composants nécessaires à la mise en oeuvre des tests.

La figure 7 montre une repésentation de l'architecture du système: ces opérateurs virtuels sont également trsè utiles dans le modèle pour paramétrer le type de disponibilité de service recherché dans le calcul.

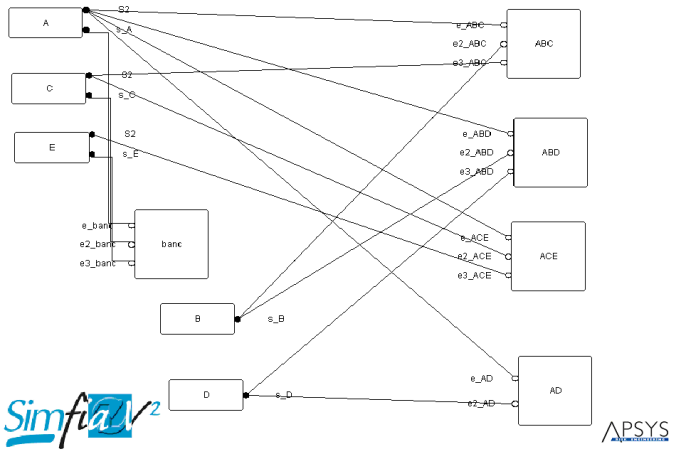


Figure 7. Architecture Banc de Test

On peut faire tourner le modèle avec la méthode simulation de Monte Carlo, et lancer le calcul de la disponibilité de service du banc de test, et comparer avec son chiffre de disponibilité intrinsèque classique.

On obtient par exemple les résultats suivants:

- disponibilité de service: espérance mathématique: 99.927%, écart type: 0.067%,
- disponibilité intrinsèque: espérance mathématique: 98.234%, écart type: 0.617%,

Dans ce cas, la possibilité de créer un modèle susceptible de décrire tous les différents scénarios, les reconfigurations et les transitions, permet de simuler une disponibilité plus proche de la réalité que la disponibilité intrinsèque correspondant au référentiel théorique de la Sûreté de Fonctionnement classique. Cela permet également d'avoir une meilleure compréhension de ce qui se passe du point de vue économique sur le projet. Dans notre exemple numérique, pour un même objectif de disponibilité de service de 99%, dans un cas, les amélioration se justifieraient, dans un autre cas, ces améliorations se ramèneraient à un gaspillage manifeste...

3. Conclusion

Ces quatre modèles simplifiés ont montré que grâce à des langages de description comportementale, la représentation de mécanismes sophistiqués est possible et ouvre la porte à tout un ensemble de questions sur l'évolution des disciplines de la maîtrise du risque.

Grâce à ces techniques de modélisation et de simulation performantes, des hypothèses plus complexes et moins classiques peuvent être prises en compte, et introduisent de nouvelles pratiques en Sûreté de Fonctionnement et par rapport aux techniques de validation de niveaux de risques et de performances de services dans l'ingénierie des systèmes. Les frontières entre les environnements de développement virtuel au service des concepteurs, et les référentiels de simulation accompagnant le travail des Fiabilistes, Logisticiens et Risk Managers tend de plus en plus à se dissoudre, pour laisser place à une discipline commune que l'on pourrait qualifier d'Assurance Performentielle...

4. Références

- M. Boiteau, Y. Dutuit, A. Rauzy & J.-P. Signoret. The altarica data-flow language in use: Assessment of production availability of a multistates system. *Reliability Engineering and System Safety*, 91:747-755, 2006.
- S. Epstein & A. Rauzy. Can We Trust PRA *Reliability Engineering and System Safety*, 88(3):195-205, 2005.
- A. Rauzy. An experimental study on six algorithms to compute transient solutions of large markov systems. *Reliability Engineering and System Safety*, 86(1):105-115, 2004.
- A. Rauzy, E. Châtelet, Y. Dutuit & C. Bérenguer. A practical comparison of methods to assess sum-of-products. *Reliability Engineering and System Safety*, 79:33-42, 2003.
- A. Rauzy. Modes automata and their compilation into fault trees. *Reliability Engineering and System Safety*, 78:1-12, 2002.
- A. Rauzy. A *m*log*m* algorithm to compute the most probable configurations of a system with multi-mode independent components. *IEEE Transactions on Reliability*, 00(00), 2001.
- A. Rauzy. A new methodology to handle boolean models with loops. *IEEE Transactions on Reliability*, 52(1):96-105, 2003.
- D. Bégay & A. Rauzy. A Realistic Involvement in Formal Methods. *Software Practice and Experience*, 31(2):191-208, 2000.
- P.F. Williams, M. Nikolskaïa & A. Rauzy. Bypassing BDD construction for reliability analysis. *Information Processing Letters*, 75:85-89, 2000.
- A. Rauzy. Mathematical Foundation of Minimal Cutsets. *IEEE Transactions on Reliability*, 50(4):389-396, december 2001.
- A. Arnold, A. Griffault, G. Point & A. Rauzy. The altarica language and its semantics. *Fundamenta Informaticae*, 34:109-124, 2000.