Projet ANR VACSIM

Validation de la commande des systèmes critiques par couplage simulation et méthodes d'analyse formelle

Réunion de lancement du projet

Présentation des tâches 1 et 2

30 septembre 2011

AGENCE NATIONALE DE LA RECHERCHE ANR

LURPA EDF INRIA RENNES - BRETAGNE ATLANTIQUE i3S sophia antipolis DS LaBRI

EDF CHANGER L'ÉNERGIE ENSEMBLE

Contexte des travaux - généralités

- La confiance dans la qualité des systèmes complexes comportant du logiciel ne repose pas uniquement sur les tests de validation
  - Nécessité de disposer de résultats des contrôles réalisés lors du développement (V validation -> V&V vérification et validation)
  - Note n°3 du tableau A.9 'Vérification du logiciel' de la norme CEI 61508-3 : « Au cours des premières phases du cycle de vie de sécurité du logiciel, la **vérification** est **statique**, par exemple inspection, revue, contrôle formel, etc. Dès que du code est produit, il est possible de réaliser un **test dynamique**. C'est la **combinaison des deux types d'informations** qui est **exigée** pour la vérification. Par exemple, la vérification du code d'un module logiciel par des moyens statiques comprend les techniques d'inspection du logiciel, de lecture croisée, d'analyse statique, de preuve formelle, etc. La vérification du code par des moyens dynamiques comprend le test fonctionnel, le test boite blanche et le test statistique. C'est la combinaison des deux types de preuves qui fournit l'assurance que chaque module logiciel satisfait à sa spécification. »

- La norme ne donne **malheureusement pas d'indication** sur la façon dont les **deux types de preuves** peuvent se **combiner**

Contexte des travaux - généralités

- Introduction au test statistique
  - Vise à estimer les probabilités de défaillance d'un système programmé
  - S'effectue sur un système complètement achevé
- Tester le système sans le modéliser en détail
  - soit avec des vecteurs de test d'une probabilité correspondant à celle des entrées du système en exploitation (test statistique par rapport à la demande)
  - soit avec des vecteurs de test d'une probabilité correspondant aux propriétés du programme (test statistique par rapport à la justesse)
- Exemple illustratif : (D'après la norme CEI 60880)
  - Supposons que n tests statistiques soient réalisés sur le système et qu'aucune défaillance ne soit observée. Alors, pour que la probabilité de détection de défaillance (pfd) soit inférieure ou égale à la valeur de pfd visée avec une probabilité α, il faut que la condition suivante soit satisfaite :
  - **Pfd <= - ln (1- α) / n**
  - Ainsi pour α = 0,95, on obtient approximativement après n tests statistiques sans défaillance :
  - Pfd <= 2,99 / n avec une probabilité de 0,95; par exemple pour obtenir une **pfd de 10–4 avec une probabilité de 95 %**, il convient qu'un ensemble de **29 900 tests** soit réalisé **sans échec**
  - Pour α = 0,99, nous obtenons approximativement:
  - Pfd <= 4,6 / n avec une probabilité de 0,99; par exemple pour obtenir une **pfd de10–4** avec une **probabilité de 99 %**, il convient qu'un ensemble de **46 000 tests** soit réalisé sans échec

Contexte des travaux - généralités

- Les caractéristiques du test statistique…
  - les tests sont des échantillons indépendants et suivant une distribution de **probabilité représentative** du **fonctionnement opérationnel** du système
  - la séquence et le nombre de tests réalisés n'influencent pas le résultat obtenu lors de la réalisation isolée d'un test
  - chaque défaillance survenant est détectée
  - le **nombre de tests** réalisés est **important**
  - les défaillances sont rares

…le rendent très coûteux !

- Projet ANR VACSIM : vise à montrer les bénéfices du couplage entre techniques de simulation et méthodes d'analyse formelles pour la validation de la commande des systèmes critiques

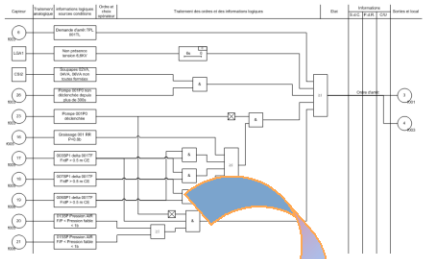
Tâche 1 : Validation par test progressif par parties de systèmes logiques

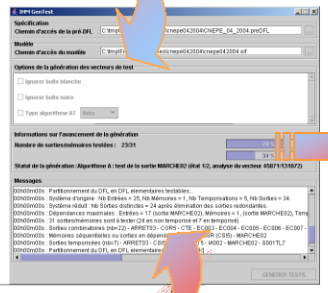
- Le projet ANR TESTEC a montré l'intérêt de l'utilisation des techniques de vérification pour réduire la taille des tests (TEst des Systèmes Temps réel Embarqués Critiques – 07 TLOG 022)
  - Application aux logiques de sécurité industrielles ou de sécurité de machines
  - Livrable L5.1 « Algorithmes de génération et d'exécution du test des systèmes logiques, séquentiels et temporisés non bouclés »
  - Livrable L6.2 « Etudes de cas industriels : mise en oeuvre du démonstrateur sur des applications industrielles, synthèse des études, préconisations de mise en œuvre et d'extensions »
  - Transformation du prototype TESTMINATOR d'EDF R&D en un logiciel de la suite ControlBuild de Geensoft / Dassault Systèmes (bénéficiant de plus d'une possibilité d'animation des spécifications logicielles par les cas de test générés)

- Techniques utilisables pour des systèmes logiques critiques qui peuvent réaliser plusieurs milliers de fonctions simples utilisant typiquement de quelques entrées à une quinzaine d'entrées

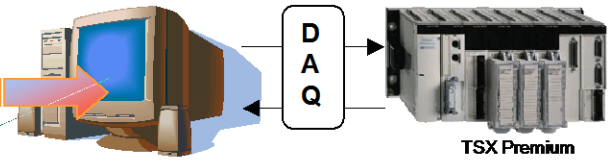
Tâche 1 : Validation par test progressif par parties de systèmes logiques - Testminator

La spécification est traduite dans un formalisme non ambiguë.

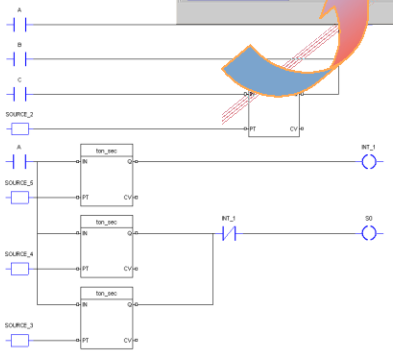




Fichiers de tests
- non temporisés
- temporisés



La réalisation est analysée pour vérifier un ensemble minimal d'hypothèses



L'outil génère un oracle de test en utilisant un ensemble de techniques de réduction permettant de maîtriser l'explosion combinatoire

Le jeu de test est exécuté :

- En simulation
- Ou directement implémenté sur la cible

Tâche 1 : Validation par test progressif par parties de systèmes logiques

- Problème de taille des tests pour plus d'une quinzaine d'entrées
  - Exemples de fonctions logiques complexes qui utilisent plus de quarante ou cinquante entrées: Synthèse d'alarmes de surveillance générale, Arrêt automatique général à une unité de production…
- Le projet TESTEC a été l'occasion de formaliser un algorithme de génération de tests et de vérifications minimales pour ces cas
  - Utilise en plus l'existence de sorties intermédiaires entre les sorties à tester et leurs entrées
  - Validation par un test progressif par parties (TPPP)
  - Le TPPP s'inspire d'une bonne pratique manuelle du test des systèmes hyper-critiques
- Le temps imparti au projet TESTEC n'a cependant pas permis de prototyper ces nouveaux algorithmes et de les mettre en œuvre sur des cas industriels
- Objectif de la tâche 1 : développer un prototype, intégré dans l'environnement ControlBuild, réalisant le test progressif par parties de systèmes logiques non-bouclés et le mettre en œuvre sur des cas industriels

Tâche 1 : Validation par test progressif par parties de systèmes logiques

- Durée : 36 mois (de T0 à T0+36)
- Responsable : EDF R&D
- Partenaires impliqués : Dassault Systèmes, I3S, LURPA

Tâche 1 : Validation par test progressif par parties de systèmes logiques

- Les livrables :

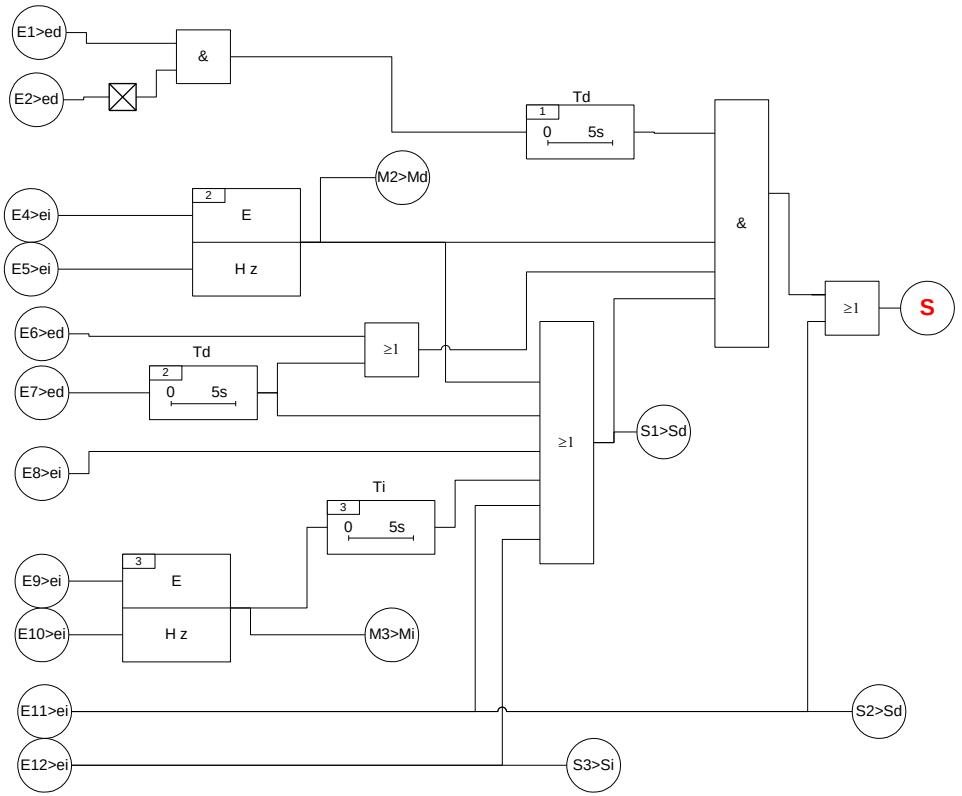
- L1.1 : Spécification et validation sur études de cas d'un algorithme de test progressif par parties de systèmes logiques non-bouclés (T0+12) – Document

- L1.2 : Développement d'un algorithme de test progressif par parties de systèmes logiques non-bouclés dans l'environnement ControlBuild (T0+24) – Logiciel

- L1.3 : Evaluation sur études de cas industriels d'un algorithme de test progressif par parties de systèmes logiques non-bouclés dans l'environnement ControlBuild (T0+36) - Document

Tâche 1 : Validation par test progressif par parties de systèmes logiques

- Exemple:



- Vérifications :
  - Pour le test et la vérification de S : S = f(T1(E1, E2), M2, E6, T2(E7), S1, S2)
  - Pour le test et la vérification de M2 : M2 = f(M(E4, E5))
  - Pour le test et la vérification de S1 : S1 = f(M2, T2(E7), E8, T3(M3), S2, S3)
  - Pour le test et la vérification de M3 : M3 = f(M(E9, E10))
  - Pour le test et la vérification de S2 : S2 = f(E11)
  - Pour le test et la vérification de S3 : S3 = f(E12)

Tâche 1 : Validation par test progressif par parties de systèmes logiques

- Tests :
  - 1ère étape : Algorithme BTP
    - Détermination de l'ensemble des états atteignables du système (temporisations, mémoires, sorties intermédiaires en amont de la sortie S à tester), des séquences de vecteurs d'entrée, et des temps d'atteinte associés
  - 2ème étape : Positionnement du système dans un état E
  - 3ème étape : Jeu d'un vecteur candidat $V_{candidat}$ (e_i, e_d)
    - L'état E évolue suivant (E, E^1, …, E^n) avant stabilisation
    - Le sous-état E_p des sorties, mémoires et temporisations directes évolue suivant (E_p, E_p^1, …, E_p^n)
  - 4ème étape : Critère de sélection des vecteurs de test
    - Si (E_p, e_d ,E_p^1, …, E_p^n) déjà connu et que la séquence (E_p, E_p^1, …, E_p^n) est plus longue que celle déjà sélectionnée, on ne retient pas cette séquence
    - Si (E_p, e_d ,E_p^1, …, E_p^n) n'est pas déjà connu ou que la séquence (E_p, E_p^1, …, E_p^n) est déjà connue et est plus courte que celle déjà sélectionnée, on la retient en écrasant celle déjà sélectionnée si elle existe
  - 5ème étape : Optimisation de l'enchainement des vecteurs de test
    - Enchainement des vecteurs selon une séquence de type RII…IISII…IIRSRSRS…
    - Utilisation de l'état atteint lors du test comme état initial pour le test suivant

Tâche 2 : Validation par simulation de partie opérative

- Bonne pratique industrielle pour la validation des systèmes de commande complexes et/ou critiques : simulateur de la partie opérative connecté aux spécifications, puis à la réalisation du système de commande (tests HIL - Hardware-In-the-Loop)

- Utilisations manuelles du simulateur dans la plupart des cas

- Lacunes dans les méthodes pour son utilisation
  - Pas de démarche d'introduction d'états et de défauts sur le modèle de partie simulée pour la sollicitation de la partie contrôle
    - Sur la partie installation
    - Sur la partie instruments
    - Sur la partie contrôle-commande
  - Objectifs peu clairs de taux de couverture des tests de l'ensemble
    - Représentation des sollicitations de la partie commande
    - Taux de couverture des tests sur les structures, les fonctions, les défauts
    - Distribution de probabilité représentative du fonctionnement opérationnel

Tâche 2 : Validation par simulation de partie opérative

- Absence de formalisation d'une démarche d'utilisation -> impossibilité d'automatiser
  - coût prohibitif
  - efforts démesurés pour une exigence d'un grand nombre de cas de tests (test statistique d'un système critique)

- Objectif de la tâche 2 : Proposer une méthode d'utilisation automatisée d'un simulateur de partie opérative, afin d'augmenter l'efficacité de ce type de technique de validation

- Développer un environnement de simulation de systèmes critiques…
- …intégrant l'analyse de propriétés de sûreté de fonctionnement…
- …qui puisse automatiser une utilisation méthodique du simulateur (définie dans le projet)…
- …sur la base d'objectifs et de taux de couverture de test rationnellement définis !

Tâche 2 : Validation par simulation de partie opérative

- Durée : 36 mois (de T0 à T0+36)
- Responsable : EDF R&D
- Partenaires impliqués : Dassault Systèmes, LaBRI, LURPA

Tâche 2 : Validation par simulation de partie opérative

- Les livrables :

- L2.1 : Spécification et validation, sur études de cas, d'une méthode d'utilisation des simulateurs de parties opératives : objectifs, propriétés de sûreté de fonctionnement, taux de couverture et limitations de la simulation (T0+12) - Document

- L2.2 : Développement d'une utilisation automatisée du simulateur ControlBuild Validation de Dassault Systèmes (T0+24) - Logiciel

- L2.3 : Evaluation sur études de cas industriels d'une utilisation automatisée du simulateur ControlBuild Validation de Dassault Systèmes (T0+36) - Document