ONERA

Analyse d'un système hydraulique avion avec AltaRica

P. Bieber, C. Castel, C. Kehren, C. Seguin ONERA
C. Bougnol, J.P. Heckman AIRBUS

Centre d'Études et de Recherches de Toulouse

Présentation générale du travail

Cadre : projet européen "Enhance Safety Assessment of Complex System"

Alenia, ***Airbus F (+ IXI, IML)*** G UK, Saab, OFFIS, IRST, Prover, ***ONERA (+ LaBri)***

But : améliorer les analyses de sécurité système amont (FHA, PSSA)

expliciter les ***exigences*** de sécurité ***qualitatives***

valider l '***allocation*** des exigences au sein du système

évaluer la tenue des exigences

Approche : expérimentale + formelle

analyse de ***systèmes avion AIRBUS***

recherche de « ***motifs d 'architecture de sécurité*** » récurrents

modélisation ***AltaRica*** des systèmes + expression ***LTL*** des exigences

évaluation à l 'aide des techniques (***model-checking***, ***preuve***, ***génération d 'arbres***…)
et outils associés (***OCAS***, ceux d '***ESACS***, …)

ONERA

Système hydraulique de l ’A320

Caractéristiques du système

FonctionImpact sur le système Rôle ? ***générer*** ***distribuer*** , réguler de

-
-
-

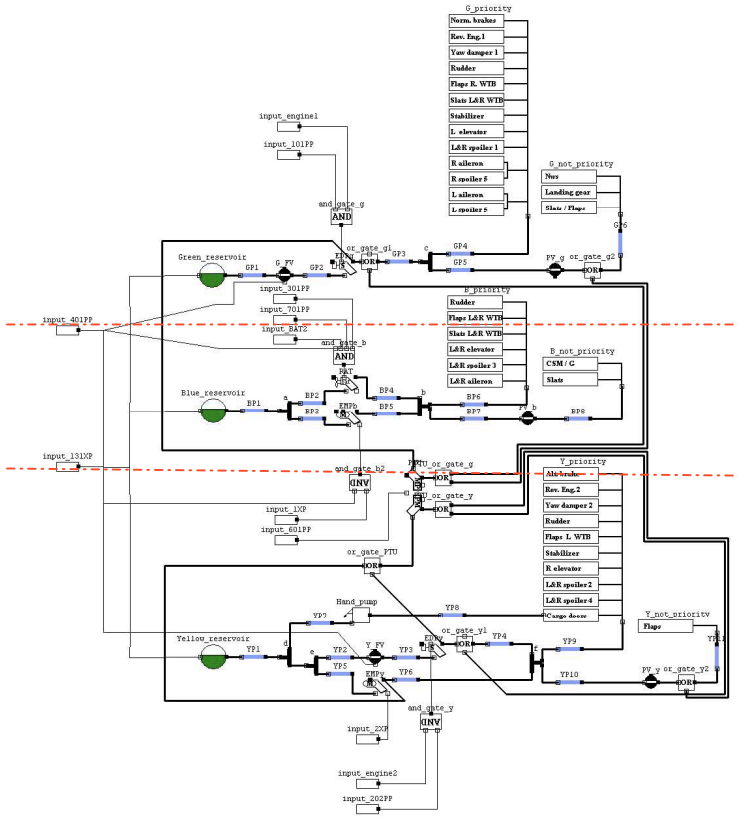
-
-
-

-
-
-

-
-

ONERA

Architecture du système

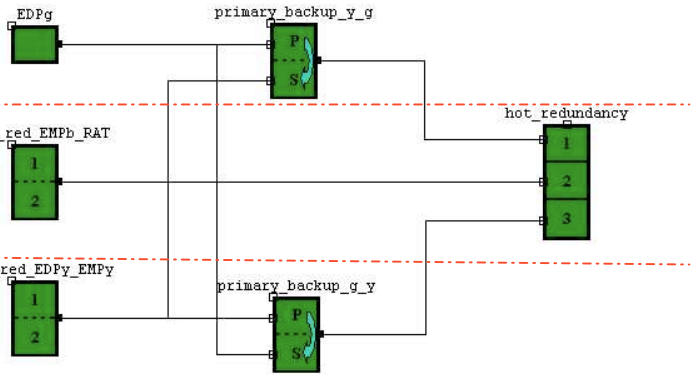


Architecture de sécurité : redondances multiples

vert

bleu

jaune



ONERA

Extraits des analyses de sécurité

Défaillances

à effet immédiat : ***perte définitive*** d 'un composant

- pour des raisons ***internes*** ou
- ***externes*** au composant (exemple : pompe activée sans fluide)

à effet à terme

- ***fuite*** de fluide : vide progressivement les réservoirs
- ***surchauffe*** des pompes

Exemples d 'exigences de sécurité

quantitative : perte totale du système CATASTROPHIQUE : 10^{-9} par heure de vol

contrepartie qualitative

- *robustesse* : il faut au moins trois pannes pour perdre le système
- ***prévention des erreurs*** : le contrôle de l 'activation des pompes est sûr, pas de pompe activée à vide, …

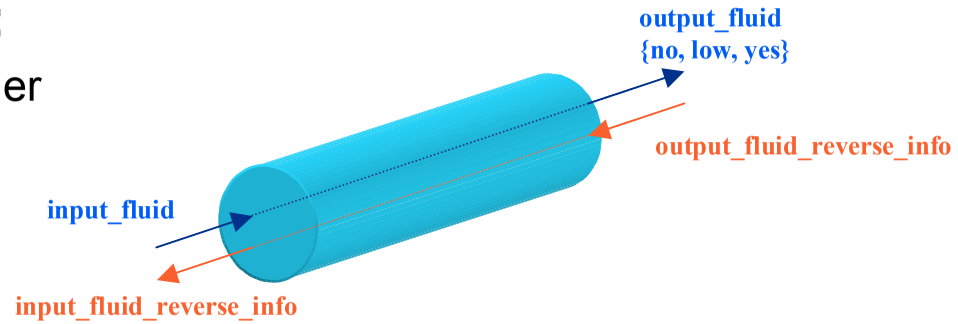
Formalisation

Modélisation AltaRica du tuyau

Information véhiculée par un tuyau :

dépend des modes de défaillance à propager

- fuites se ***propagent dans 2 sens***
- ***niveau*** de pression



Code AltaRica du tuyau

partie automate

```
trans
state_ = ok |- leak -> state_ := leakage;
```

partie assertion

```
assert
(if state_ = ok then (output = input));
(if state_ = leak then
   (if still_fluid(input,output)
     then (input_fluid_reverse_info = low and output_fluid = low)
     else(input_fluid_reverse_info = no and output_fluid = no));
```


ONERA

Le langage de la Logique Temporelle Linéaire

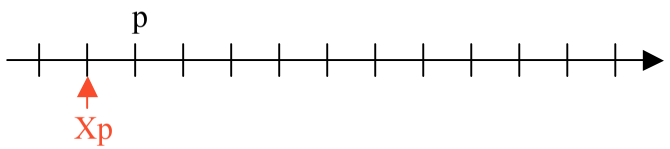
opérateurs booléens classiques & “et”, ~“non”, ...

=> description ***d’états instantanés***

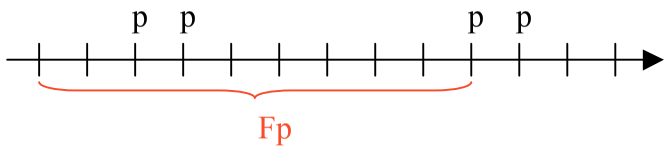
opérateurs temporels :

=> description de ***séquences***

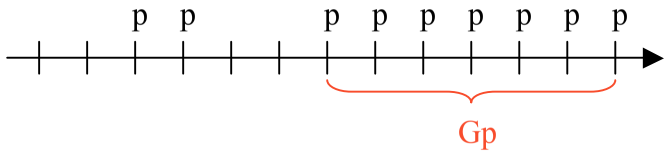
- X p : Next p



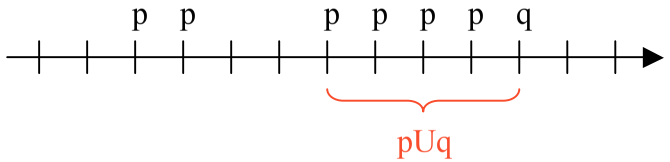
- F p : Finally p



- G p : Globally p



- p U q : p Until q



ONERA

Formalisation des exigences qualitatives en LTL
“La perte totale est causée par au moins 3 fautes”

Interprétation relative au modèle AltaRica

perte ***totale instantanée*** du système = perte des 3 distributions, verte, bleue et jaune

```
system_loss : disty.Output=no & distg.Output=no & distb.Output=no
```

perte ***totale permanente*** du système : `G system_loss`

L’exigence du point de vue du Model-checking

Expression LTL de “au plus 2 fautes” : `G atmost_2_faults`

`atmost_2_faults` : évaluable à l’aide d’un compteur de fautes

Expression LTL de l’exigence : `(G atmost_2_faults) -> ~F(G system_loss)`

L’exigence du point de vue des arbres de défaillances

l’arbre de racine `G system_loss`

a des coupes minimales contenant au moins 3 fautes

ONERA

Exploitation des modèles formels

Les modèles AltaRica réalisés : plusieurs classes pour un même système

Topologie de l ’architecture

vue fonctionnelle : le fluide circule dans un sens unique

- ***système « linéaire »***

vue physique : le fluide circule dans 2 sens

- possibilité de ***système « bouclé »,*** propage correctement les fuites

Dimension temporelle

vue statique : l ’état du système est indépendant de l ’ordre d ’arrivée des événements

- analyse relative à une configuration particulière (***phase de vol, activation des composants figées***)

vue dynamique : ***prise en compte du passé***

- pour valider le contrôle
- pannes en cascade, effets du temps...

Evaluation à l ’aide de Cécilia OCAS

Simulation

objectif

- ***valider*** le modèle
- ***injecter*** des fautes (AMDEC)

aisée pour tous les types de modèles

- ***importance du graphisme*** pour facilité le dialogue avec les experts du domaine
- ***temps de réponse excellent*** (possibilité de coupler le système hydraulique avec un système électrique de complexité comparable)

Génération d ’arbre

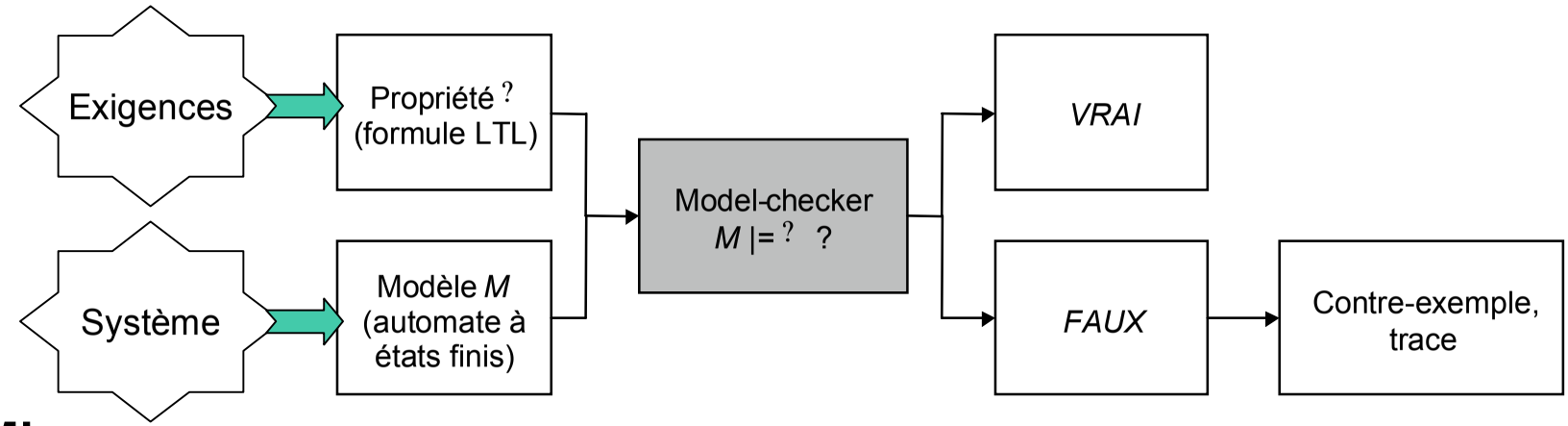
pas applicable à la classe « ***dynamique*** »

des ***problèmes*** avec les « ***boucles*** » => en cours d ’étude chez IXI

des ***résultats corrects instantanément*** dans les autres cas

Model-checking avec SMV

Principe de l ’évaluation



Mise en œuvre

langage d ’entrée de SMV

- proche d ***’AltaRica dataflow*** pour la partie système
- LTL, CTL pour la partie exigence

traduction manuelle de AltaRica vers SMV

- difficulté : les ***boucles « instantanées »*** sont à éliminer

résultats instantanés pour ***toutes les classes de modèles***

ONERA

Conclusion

Bilan

Le langage AltaRica

+ des concepts ***simples***, assimilables rapidement

+ ***adaptés*** aux modèles de type SdF

? Mémorisation du passé pour décrire des contrôleurs lourdes

+ une ***sémantique formelle*** pour appliquer différentes techniques

Cecilia OCAS : indispensable pour le développement de systèmes réels

éditeur graphique convivial : gestion de bibliothèques, connexion de composants à la souris, …

simulation graphique puissante … avec une ***programmation minimale***

? ***Génération d 'arbre*** pour une classe ***limitée*** de modèles actuellement

Model-checking

+***Technique éprouvée***, applicable avec succès dans les ***phases amont***

? Prise en compte des réels, de ***modèles très fins*** : étudiée dans ESACS

Travaux en cours

chez IXI et à l 'IML

Test de nouveaux ***algorithmes de génération d 'arbre*** sur les variantes du systèmes hydraulique

au LaBri

développement d 'un ***model-checker spécifique***

\+ ***traducteurs*** vers des langages dataflow (Lustre)

- réutilisation de model-checkers existant
- couplage de contrôleur Lustre avec un modèle physique ?

ONERA + AIRBUS

étude de l ***'allocation des exigences***

- Pratiques usuelles de décomposition des exigences : pattern
- mécanismes de composition d 'exigences : preuves en LTL, composition d 'arbres