

Etude d'un langage pour le développement d'applications interactives critiques

Vers un accompagnement formel de ce développement

Airbus - 25 Juin 2014

Bruno d'Ausbourg
Vincent Lecrubier
ONERA DTIM/LAPS

Mise au point préalable

Soyons clairs...

- Les travaux présentés ne sont pas **aboutis**
 - Il n'y a pas de produit, pas de méthode définitive
 - Il n'y a qu'un projet et des réflexions en cours
 - Une proposition révisable et amendable...
 - ...donnant matière à la rédaction d'une thèse
- Il s'agit bien de simplement **présenter**
 - Une direction suivie jusqu'à présent dans le cadre de travaux strictement exploratoires centrés sur une problématique mal connue
 - Une réflexion tout de même menée en partenariat
 - Une démarche pragmatique fondée sur l'expérience de beaucoup d'échecs en milieu industriel

Mise au point préalable

Soyons clairs...

- Les travaux présentés ne sont pas **aboutis**
 - Il n'y a pas de produit, pas de méthode définitive
 - Il n'y a qu'un projet et des réflexions en cours
 - Une proposition révisable et amendable...
 - ...donnant matière à la rédaction d'une thèse
- Il s'agit bien de simplement **présenter**
 - Une direction suivie jusqu'à présent dans le cadre de travaux strictement exploratoires centrés sur une problématique mal connue
 - Une réflexion tout de même menée en partenariat
 - Une démarche pragmatique fondée sur l'expérience de beaucoup d'échecs en milieu industriel

Mise au point préalable

Soyons clairs...

- Les travaux présentés ne sont pas **aboutis**
 - Il n'y a pas de produit, pas de méthode définitive
 - Il n'y a qu'un projet et des réflexions en cours
 - Une proposition révisable et amendable...
 - ...donnant matière à la rédaction d'une thèse
- Il s'agit bien de simplement **présenter**
 - Une direction suivie jusqu'à présent dans le cadre de travaux strictement exploratoires centrés sur une problématique mal connue
 - Une réflexion tout de même menée en partenariat
 - Une démarche pragmatique fondée sur l'expérience de beaucoup d'échecs en milieu industriel

Plan

1 Problématique

- Quelques éléments à prendre en compte
- Alors tout s'éclaire...

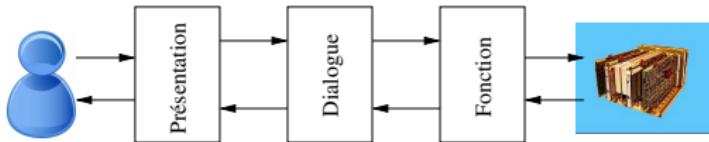
2 Apprendre du passé

- L'expérience de nombreux échecs

3 Faire du neuf

- Vers un langage de définition des IHMs embarquées critiques
- Un prototype de langage

Choses connues à propos de l'interaction



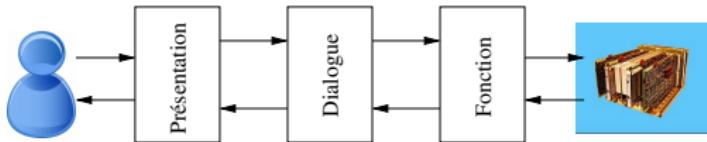
Un constat

- La part du logiciel d'interaction devient volumineuse au sein des applicatifs embarqués ou de contrôle
- La complexité de ces logiciels ne cesse de croître
- Complexité + volume ⇒ erreurs, fautes et pannes
- Pourtant le développement de ces logiciels reste peu encadré

Quelques causes identifiées

- Le domaine de l'interaction est pluri-disciplinaire
- L'interaction met en oeuvre :
 - L'utilisateur
 - Les dispositifs de dialogue et d'interaction
 - Le système
- Logiciels "plats de spaghetti"
- L'interaction : une discipline marginale
- L'interaction c'est pas intéressant

Choses connues à propos de l'interaction



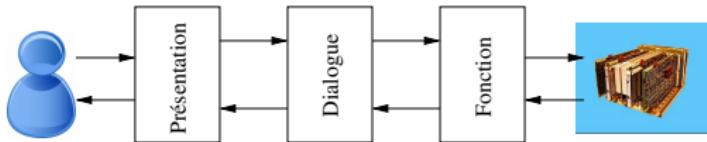
Un constat

- La part du logiciel d'interaction devient volumineuse au sein des applicatifs embarqués ou de contrôle
- La complexité de ces logiciels ne cesse de croître
- Complexité + volume ⇒ erreurs, fautes et pannes
- **Pourtant le développement de ces logiciels reste peu encadré**

Quelques causes identifiées

- Le domaine de l'interaction est pluri-disciplinaire
- L'interaction met en oeuvre :
 - L'utilisateur
 - Les dispositifs de dialogue et d'interaction
 - Le système
- Logiciels "plats de spaghetti"
- L'interaction : une discipline marginale
- L'interaction c'est pas intéressant

Choses connues à propos de l'interaction



Un constat

- La part du logiciel d'interaction devient volumineuse au sein des applicatifs embarqués ou de contrôle
- La complexité de ces logiciels ne cesse de croître
- Complexité + volume ⇒ erreurs, fautes et pannes
- **Pourtant le développement de ces logiciels reste peu encadré**

Quelques causes identifiées

- Le domaine de l'interaction est pluri-disciplinaire
- L'interaction met en oeuvre :
 - L'utilisateur
 - Les dispositifs de dialogue et d'interaction
 - Le système
- Logiciels "plats de spaghetti"
- L'interaction : une discipline marginale
- L'interaction c'est pas intéressant

Le problème avec les logiciels d'interaction embarqués c'est que...

- Le secteur aéronautique, dans ces 30 dernières années, a conçu des méthodes et des outils **rigoureux** pour le développement de ses logiciels embarqués **critiques**
 - Le domaine de l'interaction n'était pas concerné
- De nouveaux dispositifs d'interaction apparaissent dans les cockpits
 - Dotés d'un forte sophistication
 - Pilotés par des logiciels de plus en plus **complexes**
- Or les applications interactives embarquées doivent se comporter comme **prévu**
 - Le cas contraire peut générer des situations catastrophiques

Conviction 1

Il y a donc un réel enjeu à maîtriser la conception et le développement des logiciels d'interaction

Le problème avec les logiciels d'interaction embarqués c'est que...

- Le secteur aéronautique, dans ces 30 dernières années, a conçu des méthodes et des outils **rigoureux** pour le développement de ses logiciels embarqués **critiques**
 - Le domaine de l'interaction n'était pas concerné
- De nouveaux dispositifs d'interaction apparaissent dans les cockpits
 - Dotés d'un forte sophistication
 - Pilotés par des logiciels de plus en plus **complexes**
- Or les applications interactives embarquées doivent se comporter comme **prévu**
 - Le cas contraire peut générer des situations catastrophiques

Conviction 1

Il y a donc un réel enjeu à maîtriser la conception et le développement des logiciels d'interaction

Le problème avec les logiciels d'interaction embarqués c'est que...

- Le secteur aéronautique, dans ces 30 dernières années, a conçu des méthodes et des outils **rigoureux** pour le développement de ses logiciels embarqués **critiques**
 - Le domaine de l'interaction n'était pas concerné
- De nouveaux dispositifs d'interaction apparaissent dans les cockpits
 - Dotés d'un forte sophistication
 - Pilotés par des logiciels de plus en plus **complexes**
- Or les applications interactives embarquées doivent se comporter comme **prévu**
 - Le cas contraire peut générer des situations catastrophiques

Conviction 1

Il y a donc un réel enjeu à maîtriser la conception et le développement des logiciels d'interaction

Le problème avec les logiciels d'interaction embarqués c'est que...

- Le secteur aéronautique, dans ces 30 dernières années, a conçu des méthodes et des outils **rigoureux** pour le développement de ses logiciels embarqués **critiques**
 - Le domaine de l'interaction n'était pas concerné
- De nouveaux dispositifs d'interaction apparaissent dans les cockpits
 - Dotés d'un forte sophistication
 - Pilotés par des logiciels de plus en plus **complexes**
- Or les applications interactives embarquées doivent se comporter comme **prévu**
 - Le cas contraire peut générer des situations catastrophiques

Conviction 1

Il y a donc un réel enjeu à maîtriser la conception et le développement des logiciels d'interaction

Plus fondamentalement...

- Les approches pour le développement des systèmes embarqués critiques sont, de près ou de loin, fondées sur un cycle en V
- Ce modèle n'est **pas adapté** au développement des logiciels d'Interaction
 - Itérations multiples entre concepteurs et usagers nécessaires
 - Conception et tests fortement mêlés
 - Les divers et multiples participants au développement n'ont pas de moyen partagé d'exprimer rigoureusement et de façon appropriée le comportement attendu des interfaces

Conviction 2

Il manque un langage commun, bien défini, pour représenter le logiciel applicatif conçu. La disponibilité d'un tel langage assure que :

- Les *concepteurs* peuvent itérer sur leur conception avant de l'injecter dans les processus de développement
- Les *développeurs* peuvent tester la conformité de leur développement à la définition formulée pour l'application logicielle

Plus fondamentalement...

- Les approches pour le développement des systèmes embarqués critiques sont, de près ou de loin, fondées sur un cycle en V
- Ce modèle n'est **pas adapté** au développement des logiciels d'Interaction
 - Itérations multiples entre concepteurs et usagers nécessaires
 - Conception et tests fortement mêlés
 - Les divers et multiples participants au développement n'ont pas de moyen partagé d'exprimer rigoureusement et de façon appropriée le comportement attendu des interfaces

Conviction 2

Il manque un langage commun, bien défini, pour représenter le logiciel applicatif conçu. La disponibilité d'un tel langage assure que :

- Les *concepteurs* peuvent itérer sur leur conception avant de l'injecter dans les processus de développement
- Les *développeurs* peuvent tester la conformité de leur développement à la définition formulée pour l'application logicielle

D'où l'idée de ...

- Proposer un **langage** de définition **pivot**
 - Suffisamment abstrait pour être dégagé des problèmes d'implantation
 - Sémantique formalisée
 - Notations aisées pour la définition d'applications interactives
 - Appréhendable par tous les participants à la définition des applications
- Proposer un **cadre** permettant :
 - Le prototypage des applications
 - La vérification et la preuve de propriétés sur ces applications
 - Une génération de code apportant un haut degré d'assurance
 - Une optimisation des tests de l'application développée

Plan

1 Problématique

- Quelques éléments à prendre en compte
- Alors tout s'éclaire...

2 Apprendre du passé

- L'expérience de nombreux échecs

3 Faire du neuf

- Vers un langage de définition des IHMs embarquées critiques
- Un prototype de langage

Téléphonie à écran

1996-2000

- Initialement une demande du CNET
 - Maîtriser le comportement des téléphones à écran
 - Utilisation de Lustre pour décrire l'interaction
 - Construction de modèles Lustre à partir des codes UIL issus de générateurs d'interfaces
- Lustre : un bon support pour la modélisation des comportements interactifs d'un téléphone
- Réutilisation par ALCATEL : utilisation de Lustre pour représenter le cœur du dialogue interactif (téléphonie mobile)
 - Le dialogue rapidement décrit en Lustre
 - Vérifications de propriétés
 - Génération automatique du code du dialogue
 - Couplage avec les composants fonctionnels ou de présentation
 - *Time to market*

Un premier raté

Le CNET a disparu, ses équipes de recherche ont été dissoutes... ces travaux aussi

Téléphonie à écran

1996-2000

- Initialement une demande du CNET
 - Maîtriser le comportement des téléphones à écran
 - Utilisation de Lustre pour décrire l'interaction
 - Construction de modèles Lustre à partir des codes UIL issus de générateurs d'interfaces
- Lustre : un bon support pour la modélisation des comportements interactifs d'un téléphone
- Réutilisation par ALCATEL : utilisation de Lustre pour représenter le cœur du dialogue interactif (téléphonie mobile)
 - Le dialogue rapidement décrit en Lustre
 - Vérifications de propriétés
 - Génération automatique du code du dialogue
 - Couplage avec les composants fonctionnels ou de présentation
 - *Time to market*

Un premier raté

Le CNET a disparu, ses équipes de recherche ont été dissoutes... ces travaux aussi

- Systèmes de contrôle au sol
 - Besoin issu d'un incident grave survenu sur un système de contrôle
 - Perte d'un satellite
- CNES et CS
- Problématique
 - Propriétés de sûreté à vérifier exhaustivement
 - Guider l'effort de test
- Approche utilisée
 - Utilisation de Lustre pour la formalisation du dialogue

Une impasse

La modélisation en Lustre permet de raisonner et calculer sur les modèles. Mais sa production suppose que l'on connaît a priori les codes et que l'on sait où, quoi et comment chercher...

- Systèmes de contrôle au sol
 - Besoin issu d'un incident grave survenu sur un système de contrôle
 - Perte d'un satellite
- CNES et CS
- Problématique
 - Propriétés de sûreté à vérifier exhaustivement
 - Guider l'effort de test
- Approche utilisée
 - Utilisation de Lustre pour la formalisation du dialogue

Une impasse

La modélisation en Lustre permet de raisonner et calculer sur les modèles.
Mais sa production suppose que l'on connaît a priori les codes et que l'on sait où, quoi et comment chercher...

CNET, le retour !...

Verbatim (FT RD, LIG, LISI/ENSMA, IRISA, Clearsy)

- Problématique : communication multi modale
 - Propriétés ergonomiques et de sûreté à vérifier exhaustivement
 - Réduire les coûts de développement et de test
 - *Time to Market*
 - Traitement de la multi modalité
- Approches utilisées
 - Analyse statique de codes C et Java
 - Abstractions de constructions reconnues dans le code
 - Formalisation en Promela
 - Vérification par model checking

Un léger mieux

L'utilisation intensive d'abstractions et de techniques d'analyse statiques sur des codes C ou Java permet la construction de modèles *réduits* de l'application.

CNET, le retour !...

Verbatim (FT RD, LIG, LISI/ENSMA, IRISA, Clearsy)

- Problématique : communication multi modale
 - Propriétés ergonomiques et de sûreté à vérifier exhaustivement
 - Réduire les coûts de développement et de test
 - *Time to Market*
 - Traitement de la multi modalité
- Approches utilisées
 - Analyse statique de codes C et Java
 - Abstractions de constructions reconnues dans le code
 - Formalisation en Promela
 - Vérification par model checking

Un léger mieux

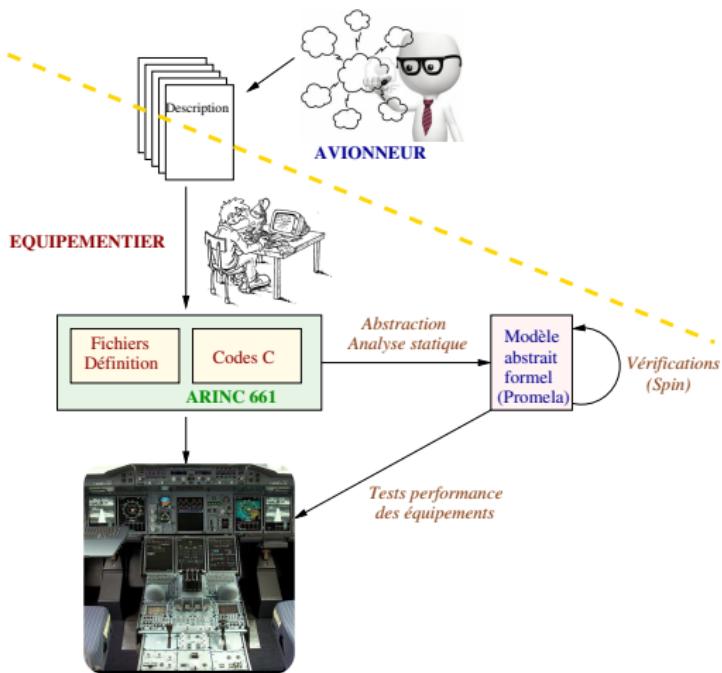
L'utilisation intensive d'abstractions et de techniques d'analyse statiques sur des codes C ou Java permet la construction de modèles *réduits* de l'application.

Quelques refs sur le sujet...

- Bruno D'AUSBOURG, Guy DURRIEU et Pierre ROCHE. "Using flows for describing and verifying the behaviour of interactive systems". French. In : *Annales Des Télécommunications* 51.9-10 (1996), p. 474–482
- Bruno D'AUSBOURG, Guy DURRIEU et Pierre ROCHE. "Deriving a formal model of an interactive system from its UIL description in order to verify and to test its behaviour". English. In : *Design, Specification and Verification of Interactive Systems '96*. Sous la dir. de Francois BODART et Jean VANDERDONCKT. Eurographics. Springer Vienna, 1996, p. 105–122. ISBN : 978-3-211-82900-4
- Bruno D'AUSBOURG. "Using Model Checking for the Automatic Validation of User Interface Systems." In : *DSV-IS*. Sous la dir. de Panos MARKOPOULOS et Peter JOHNSON. T. 1. Springer, 30 mar. 2006, p. 242–260
- Bruno D'AUSBOURG et al. "Helping the Automated Validation Process of User Interfaces Systems". In : *Proceedings of the 20th International Conference on Software Engineering. ICSE '98*. Kyoto, Japan : IEEE Computer Society, 1998, p. 219–228. ISBN : 0-8186-8368-6
- Bruno D'AUSBOURG et Jacques CAZIN. "Using TRIO Specifications to Generate Test Cases for an Interactive System". English. In : *Design, Specification and Verification of Interactive Systems '99*. Sous la dir. de David DUKE et Angel PUERTA. Eurographics. Springer Vienna, 1999, p. 148–166. ISBN : 978-3-211-83405-3
- Alexandre CORTIER, Bruno D'AUSBOURG et Yamine AÏT-AMEUR. "Formal Validation of Java/Swing User Interfaces with the Event B Method". English. In : *Human-Computer Interaction. Interaction Design and Usability*. Sous la dir. de JulieA. JACKO. T. 4550. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, p. 1062–1071. ISBN : 978-3-540-73104-7

Application à des logiciels d'interaction cockpit (THAV)

Renforcer le processus de validation à l'aide de techniques formelles ...



Besoin exprimé

- Evaluer l'apport de techniques de vérification formelle
- Sans toucher au processus de développement
- Réduire l'effort de test
- Exhiber des tests de performance

Approche expérimentée

- Analyse statique et abstraction des codes (définition et C)
- Formalisation en Promela

Utilisation de techniques de vérification sur modèle

- Vérification de propriétés comportementales
- Génération de tests de performances sur matériel
- Génération de scénarios de simulation

Ça ne marche pas bien...

Encore raté !

- L'effort de formalisation s'exerce sur le produit du développement
- Pas de réelle vérification de conformité du code à une définition
- Pas de validation convaincante
- Formalisation délicate, *ad hoc*, non générique, fondée sur une structure connue et repérable du code
- Manque d'un texte de référence

Ça ne marche pas bien...

Encore raté !

- L'effort de formalisation s'exerce sur le produit du développement
- Pas de réelle vérification de conformité du code à une définition
- Pas de validation convaincante
- Formalisation délicate, *ad hoc*, non générique, fondée sur une structure connue et repérable du code
- Manque d'un texte de référence

Plan

1 Problématique

- Quelques éléments à prendre en compte
- Alors tout s'éclaire...

2 Apprendre du passé

- L'expérience de nombreux échecs

3 Faire du neuf

- Vers un langage de définition des IHMs embarquées critiques
- Un prototype de langage

Pour un langage *L/L* de description d'applications interactives

EQUIPEMENTIER



CONCEPTEUR



● Langage de description *L/L*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

● Prototypage rapide

● Projections vers d'autres formalismes *L'*

- Vérification par model checking ou preuve

● Utilisation de techniques de réécriture

● Concrétisation et génération

- Choix de toolkits
- Codes embarqués critiques WIMP
- Codes post WIMP

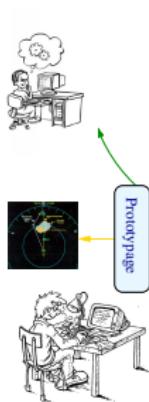
● Génération de tests

● Objectifs

- Vérification et validation amont
- Génération de code sûr

Pour un langage *L/L* de description d'applications interactives

EQUIPEMENTIER



CONCEPTEUR



● Langage de description *L/L*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

● Prototypage rapide

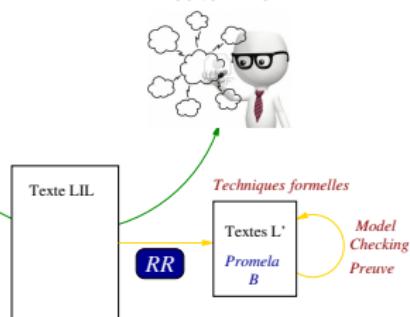
- Projections vers d'autres formalismes *L'*
 - Vérification par model checking ou preuve
- Utilisation de techniques de réécriture
- Concrétisation et génération
 - Choix de toolkits
 - Codes embarqués critiques WIMP
 - Codes post WIMP
- Génération de tests
- Objectifs
 - Vérification et validation amont
 - Génération de code sûr

Pour un langage *L/L* de description d'applications interactives

EQUIPEMENTIER



CONCEPTEUR



● Langage de description *L/L*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

● Prototypage rapide

● Projections vers d'autres formalismes *L'*

- Vérification par model checking ou preuve

● Utilisation de techniques de réécriture

● Concrétisation et génération

- Choix de toolkits
- Codes embarqués critiques WIMP
- Codes post WIMP

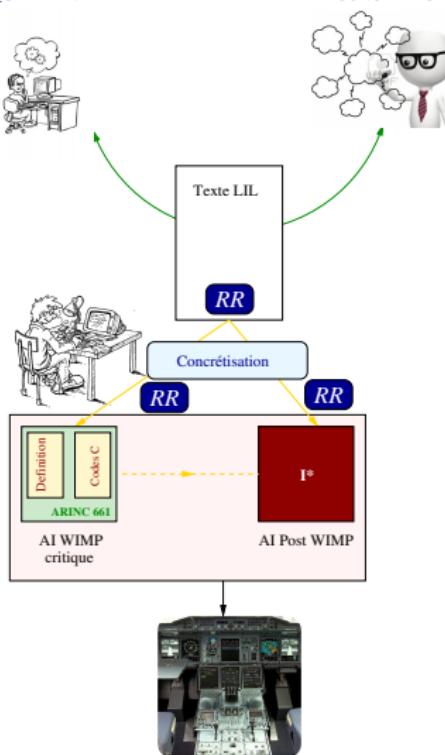
● Génération de tests

● Objectifs

- Vérification et validation amont
- Génération de code sûr

Pour un langage *LIL* de description d'applications interactives

EQUIPEMENTIER



• Langage de description *LIL*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

• Prototypage rapide

• Projections vers d'autres formalismes *L'*

- Vérification par model checking ou preuve

• Utilisation de techniques de réécriture

• Concrétisation et génération

- Choix de toolkits
- Codes embarqués critiques WIMP
- Codes post WIMP

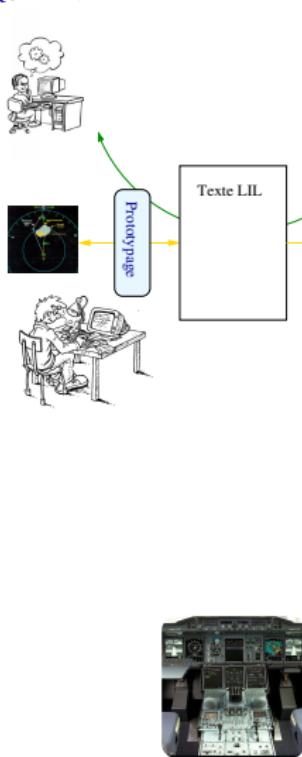
• Génération de tests

• Objectifs

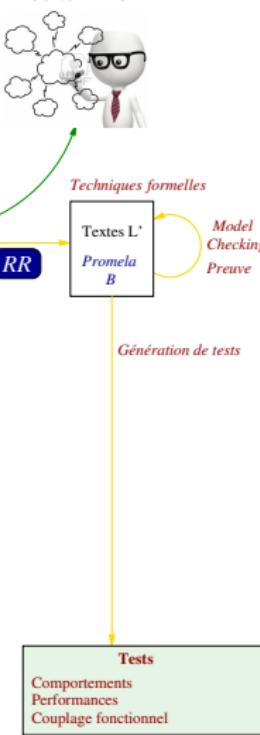
- Vérification et validation amont
- Génération de code sûr

Pour un langage *LIL* de description d'applications interactives

EQUIPEMENTIER



CONCEPTEUR



Langage de description *LIL*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

Prototypage rapide

Projections vers d'autres formalismes *L'*

- Vérification par model checking ou preuve

Utilisation de techniques de réécriture

Concrétisation et génération

- Choix de toolkits
- Codes embarqués critiques WIMP
- Codes post WIMP

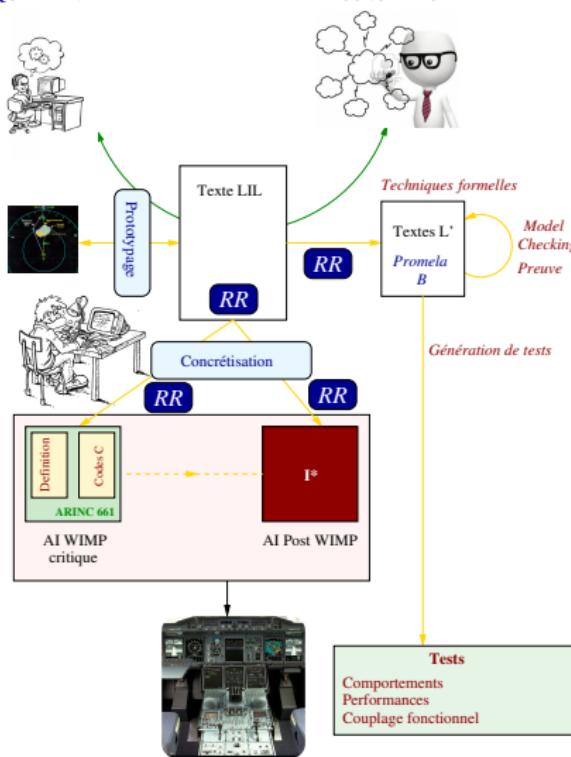
Génération de tests

Objectifs

- Vérification et validation amont
- Génération de code sûr

Pour un langage *LIL* de description d'applications interactives

EQUIPEMENTIER



Langage de description *LIL*

- Formel, déclaratif, abstrait, prototypable
- Description des interacteurs, du dialogue et du lien avec la part fonctionnelle
- Expression d'exigences ou de propriétés

Prototypage rapide

Projections vers d'autres formalismes *L'*

- Vérification par model checking ou preuve

Utilisation de techniques de réécriture

Concrétisation et génération

- Choix de toolkits
- Codes embarqués critiques WIMP
- Codes post WIMP

Génération de tests

Objectifs

- Vérification et validation amont
- Génération de code sûr

Projections vers d'autres formalismes L'

- L/L est un langage abstrait
 - Spécifique au domaine de la conception de logiciels d'interaction
 - Notations aisées pour le concepteur
 - Déclarations conformes à son mode de pensée
 - Formalisé et doté d'une sémantique précise
- Projections de L vers d'autres formalismes L'
 - Pour la vérification (Promela, LTL) ou la preuve (B événementiel)
 - Pour la génération de tests du système
 - Pour la génération de code
- Ces projections reposent sur des **transformations**
 - Règles de réécriture
 - Formellement construites et validées

Génération de logiciels interactifs

- Nécessite une étape d'enrichissement et de *concrétisation* de textes *L*
 - Choix d'interacteurs dans des toolkits
 - Définition concrète
- Génération de code pour différentes plateformes
 - ARINC 661 pour des applications embarquées critiques WIMP
 - I* pour des applications (critiques) post WIMP
- Ces opérations reposent sur des **transformations**
 - Règles de réécriture
 - Formellement construites et validées

Voir Vincent...