Fabien BONNEFOI

Vérification Formelle des Spécifications de Systèmes Complexes
Application aux Systèmes de Transport Intelligents

Soutenance de Thèse sous la direction de

M. Fabrice KORDON

Jury

Mme. Béatrice BÉRARD
Mme. Christine CHOPPY
M. Guy FREMONT
M. Yvon KERMARREC
M. Franck POMMEREAU
M. Jean-Claude ROYER

UPMC SORBONNE UNIVERSITÉS

cnrs

LIP6

COFIROUTE

Systèmes Complexes

Système de Transport Intelligent (STI)

- nombreuses entités / interactions
- phénomènes hybrides (discrets et continus)

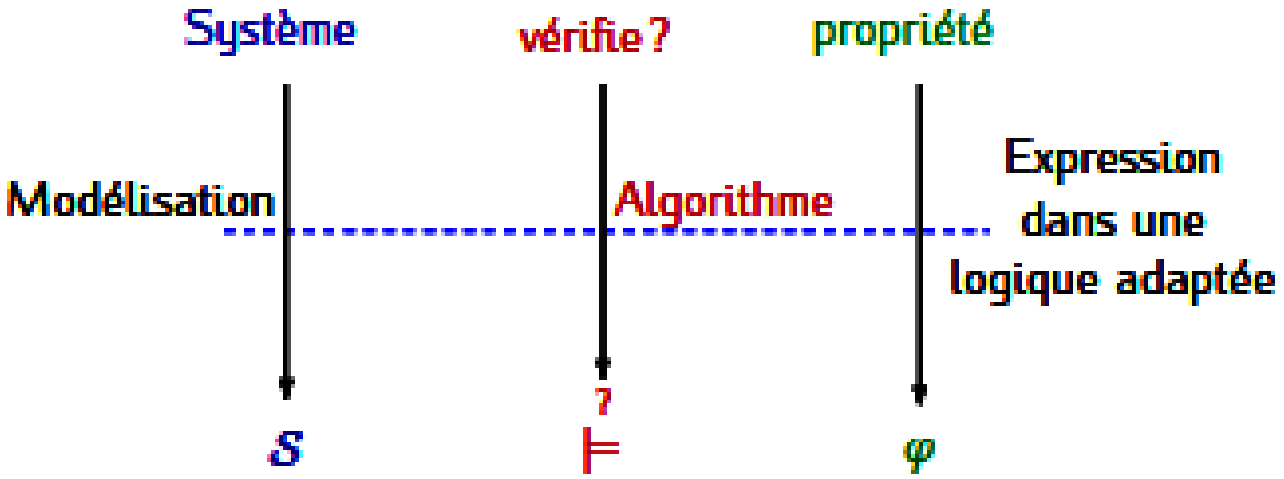
1 - **Simulation**
+ aspects quantitatifs
- exhaustivité des tests

2 - **Preuve par Théorème**
+ preuve formelle
- preuve semi-automatique

3 - **Model checking**
+ preuve automatique
+ contre exemples
- aspects continus / explosion combinatoire

Model Checking

Principes:



Défis de la mise en oeuvre du MODEL CHECKING [Sifakis 2007]

- Liens formels entre le modèle et le système
- Passage à l'échelle et explosion combinatoire:
  - approche par abstractions du système
  - approche par composition du modèle
    (pour une classe de systèmes ou de propriétés)

Projet Européen SAFESPOT

6ème PCRD - 2006-2010 - 51 équipes (12 pays) 8 sous-projets - 38 M€

Objectifs

- améliorer la sécurité
- alerte danger (< 1 min)
(entre véhicules
et depuis l'infrastructure)

Technologies

- Coopération : réseau P2P
- Communications : V2V – V2I / 802.11p (>200km/h)
- Base de donnée locale et temps réelle

Conception

- Architecture : symétrie entre Infrastructure / Véhicules
- Applications ≠ entre Infrastructure et véhicules
- Panneaux à messages variables et IHM embarqué

Complexité du système

- 60 cas d'utilisations / 13 applications
- 7 composants (composites) - 7 interfaces – 13 connections
- DB: > 100 tables : Messages > 200
- Phénomènes **hybrides** (discrets et continus)

Responsable du sous-projet:

Cooperative Safety System Infrastructure Based

Responsable de l'application:

Hazard & Incident Warning (H&IW)

Plan

- Problématiques / Objectifs / Méthode
- Contribution 1 : Architecture formelle pour STI
- Contribution 2 : Vérification des comportements
- Contribution 3 : Vérification de propriétés hybrides
- Résultats
- Conclusion & Perspectives

Problématique & Objectifs

Spécifications pour systèmes complexes

Pb: nombre de partenaires (≠ pratiques & spécialités)

→ consensus industriel → UML

Production de modèles formels pour systèmes complexes

Transformation de modèles UML → ajout d'une sémantique formelle à UML

Pb: passage à l'échelle (nb. d'éléments à modéliser)

Pb: modélisation et vérification au cours des étapes de spécification

Pb: aspects continus et discrets (hybrides)

Objectifs:

Aide à la production de modèles formels pour Systèmes complexes

- Différents niveaux d'abstractions / différentes étapes de spécification
- Vérification des composants et du modèle complet
- Vérification des contraintes hybrides

Méthode

Choix d'une notation formelle
- Réseaux de Petri Symétriques (RdPS) (réduction de l'explosion combinatoire)

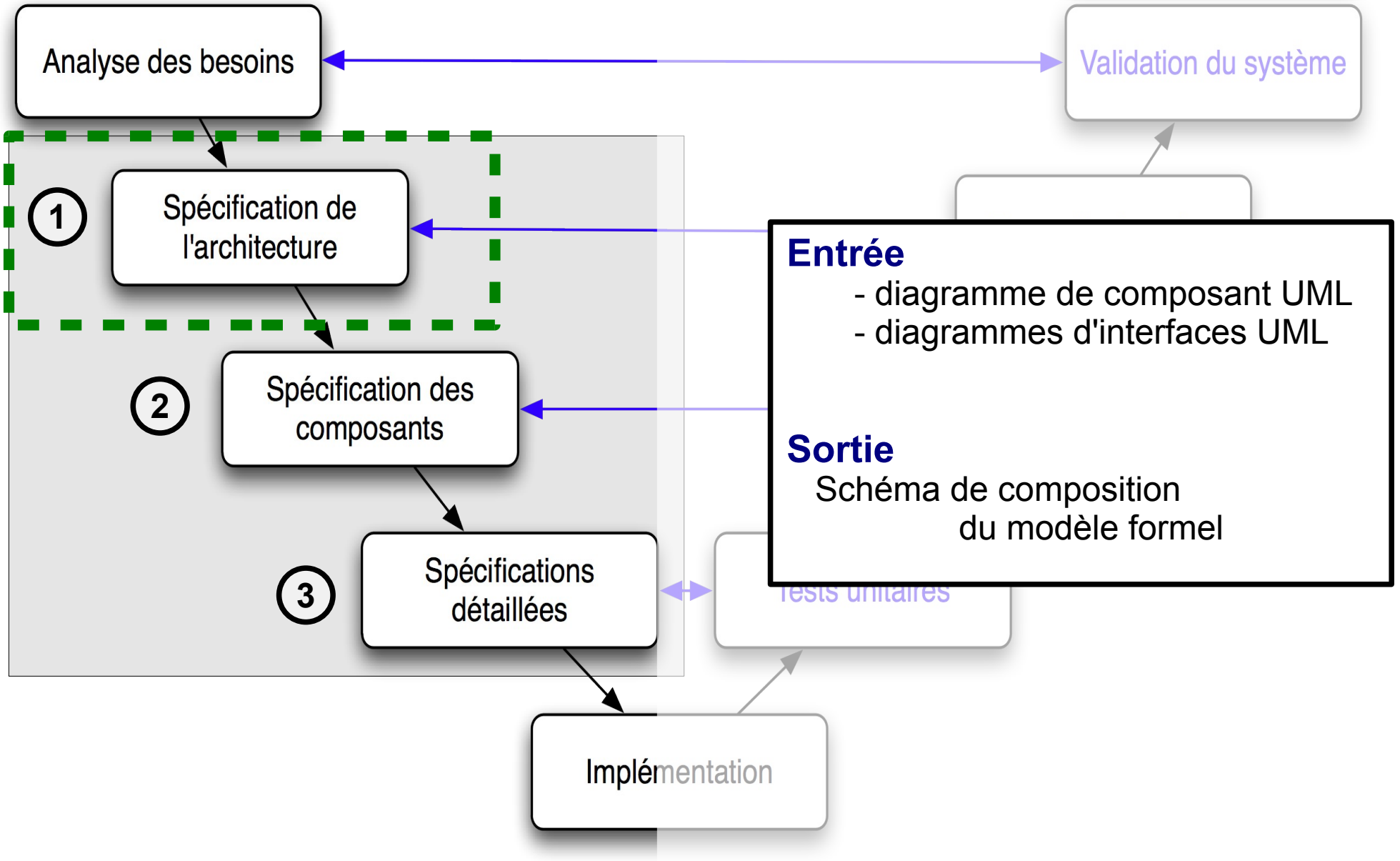
Transformation de spécifications UML en modèles formels
- définition de règles de transformations (ajout sémantique formelle)
- raffinement

Définition d'une architecture pour une classe de systèmes (STI)
- approche modulaire / réutilisation des modèles

Vérification progressive du système
- composant isolé puis modèle assemblé

Technique de discrétisation
- modélisation et vérification des aspects continus

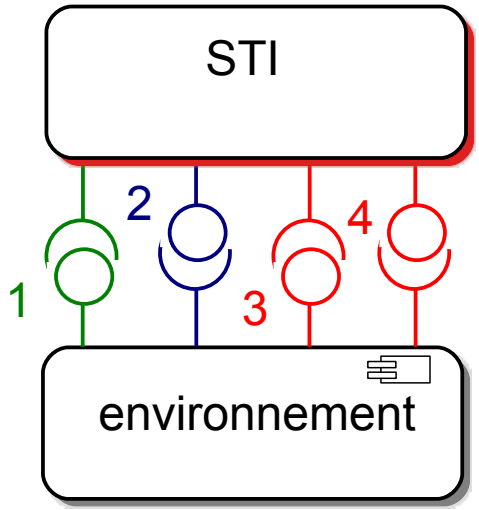
Contribution 1

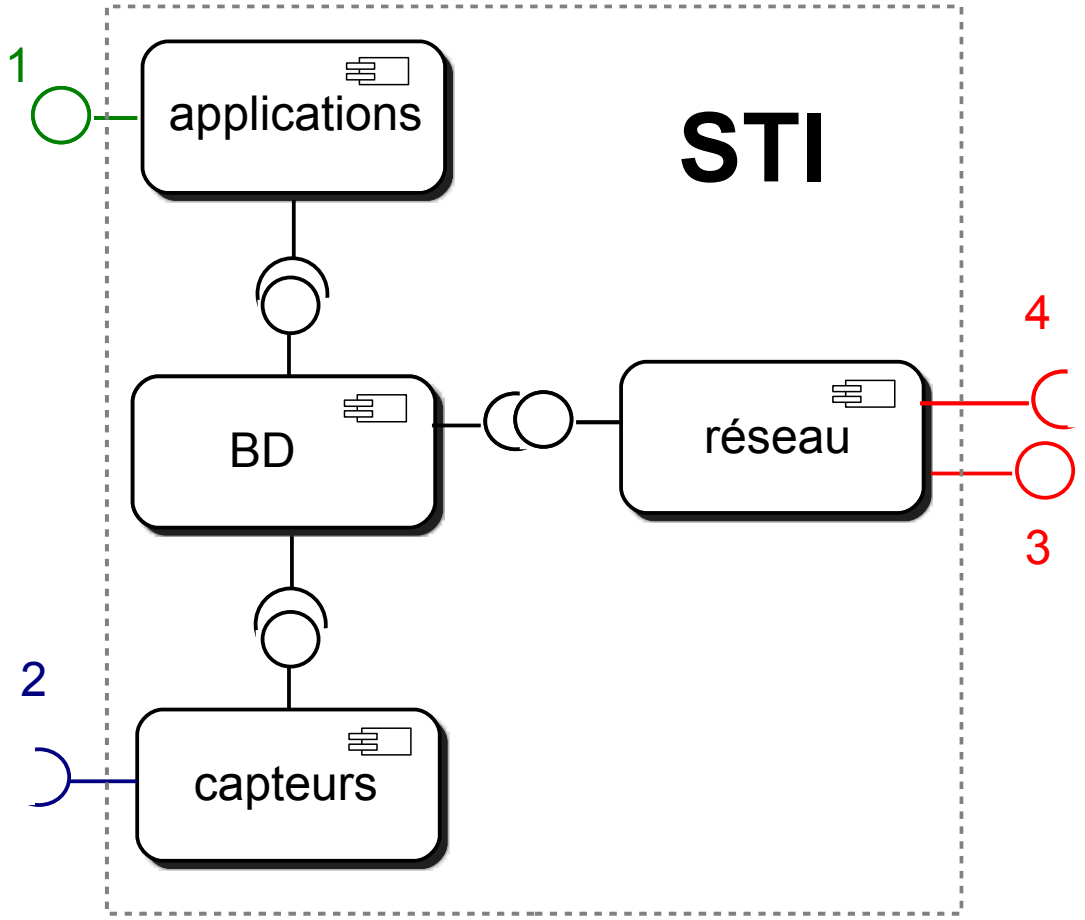


Architecture pour STI

Architecture pour STI

ex: FRAME → non formelle

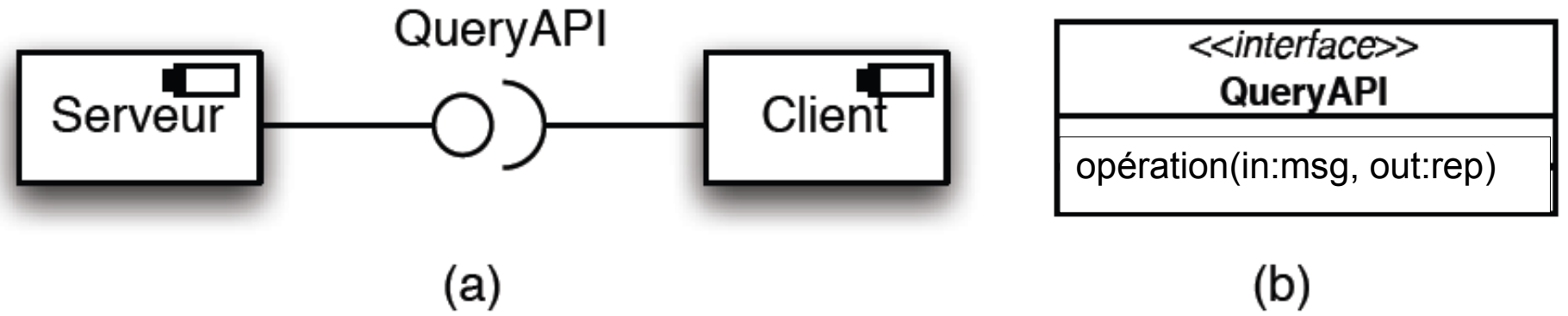




Environnement:

- Ordonnanceur: abstraction du temps
- Canal de communication
- Conducteurs: passif / actif / aléatoire

Transformation des Interfaces 1/2



Sémantique RdPs

Identifiants

Serveur → Class Serveur is 1..M
Client → Class Client is 1..N

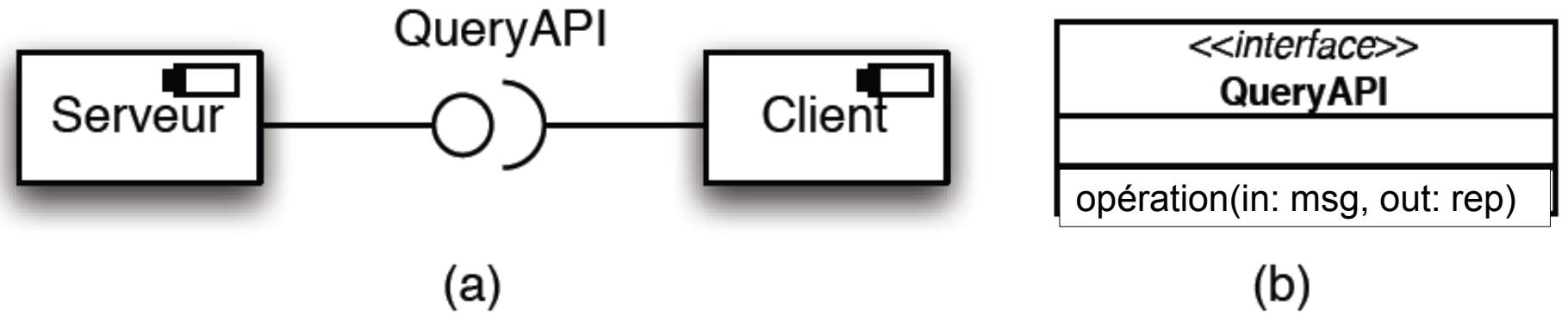
Données

Msg → Class Msg is 1..O
Rep → Class Rep is 1..P
I_Query_in is < Serveur, Client, Msg >
I_Query_out is < Serveur, Cleint, Rep >

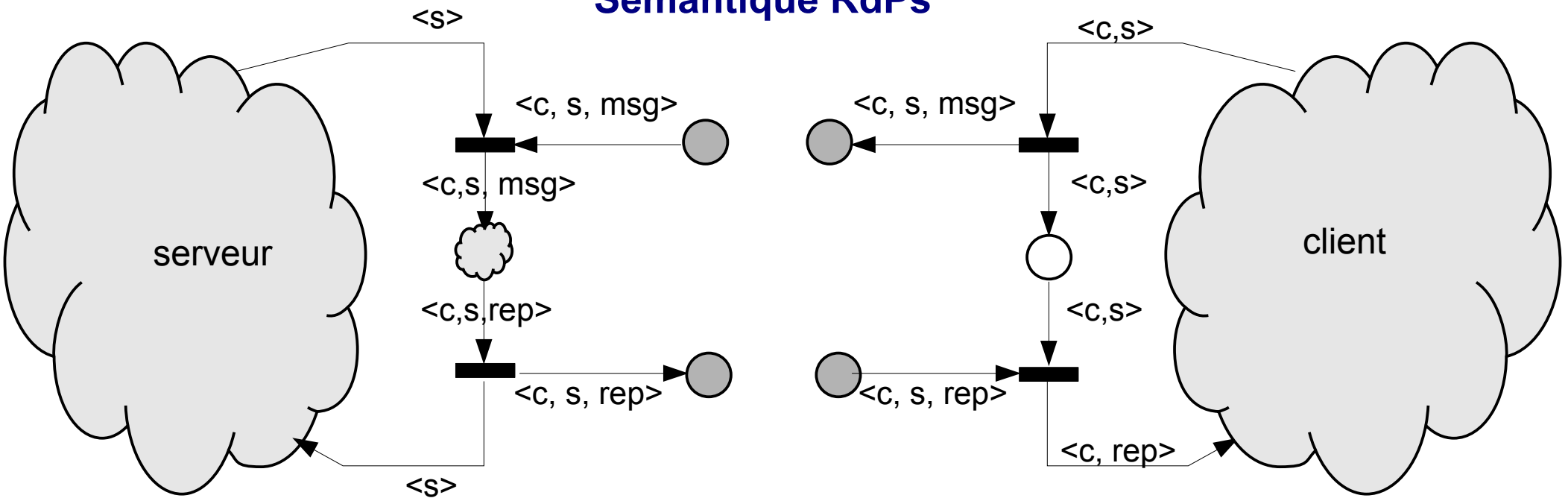
Variables

s in Serveur
c in Client
msg in Msg
rep in Rep

Transformation des Interfaces 2/2

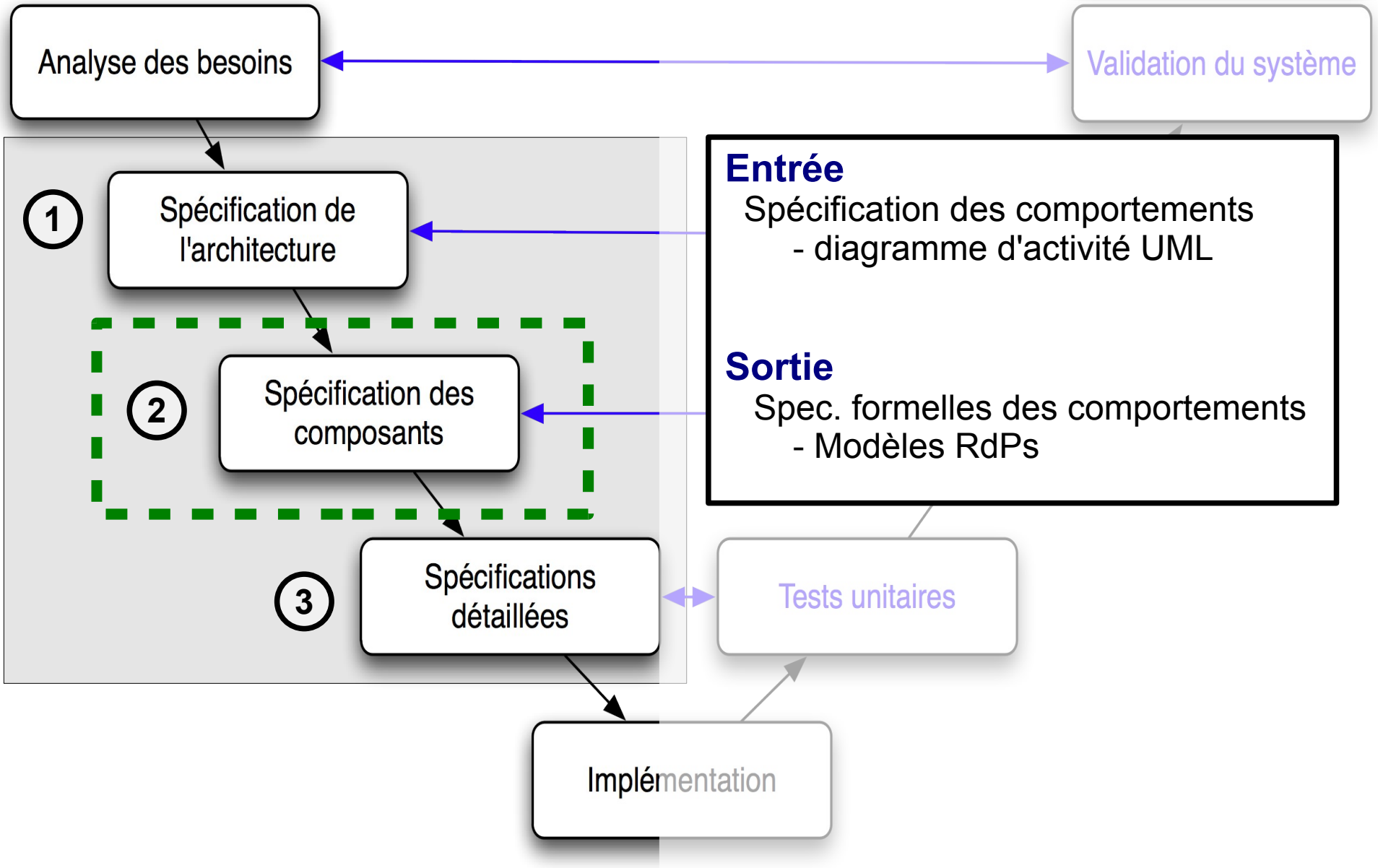


Sémantique RdPs



8 règles de transformations → 6 types d'interfaces différentes

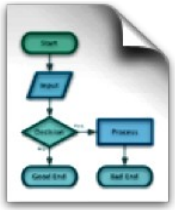
Contribution 2



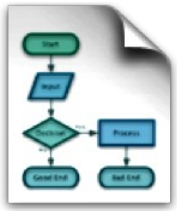
Approche Modulaire

Spécifications
(semi-formelles)

Modèle formel

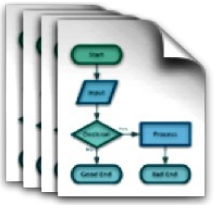


Architecture
(composants UML)

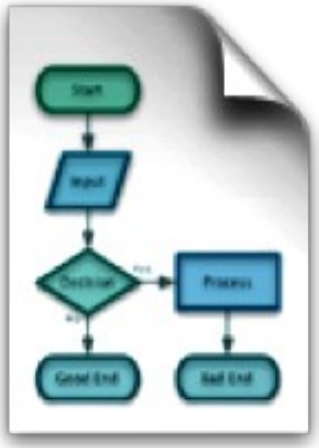


Comportement
(activités UML)

Règles
d'assemblage
(Ps)



Bibliothèque de modèles
(RdPS)



Modèle complet
(RdPS)

Définition d'un environnement minimal

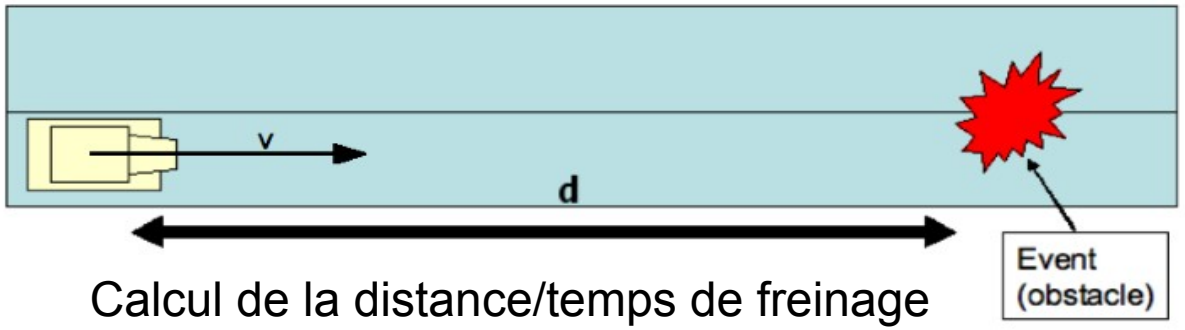
vérification

Hazard & Incident Warning (H&IW)

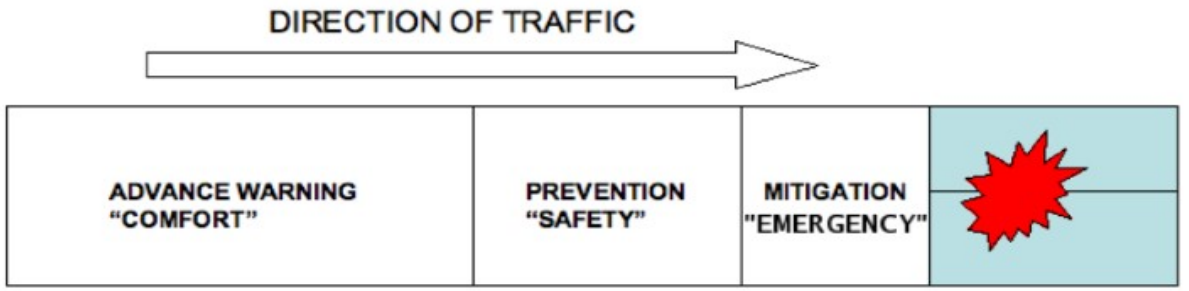
Calcul des alertes

- vitesse (v)
- capacité de freinage (b)
- distance à l'obstacle (d)
- type de route
- condition de circulation

→ aspects continus



Calcul de la distance/temps de freinage



Fonctionnement

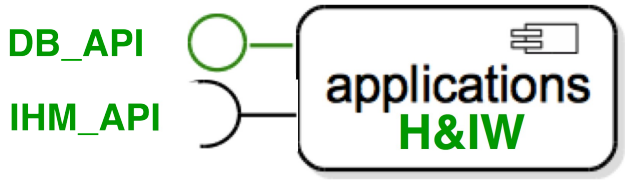
- déployée sur l'infrastructure
- dirigée par l'occurrence d'événements
- différent obstacles (véhicules accidentés / arrêtés, chantiers, piétons ...)
- 1 processus surveille une zone donnée
- déclenche un processus pour le calcule et l'envoi des alertes

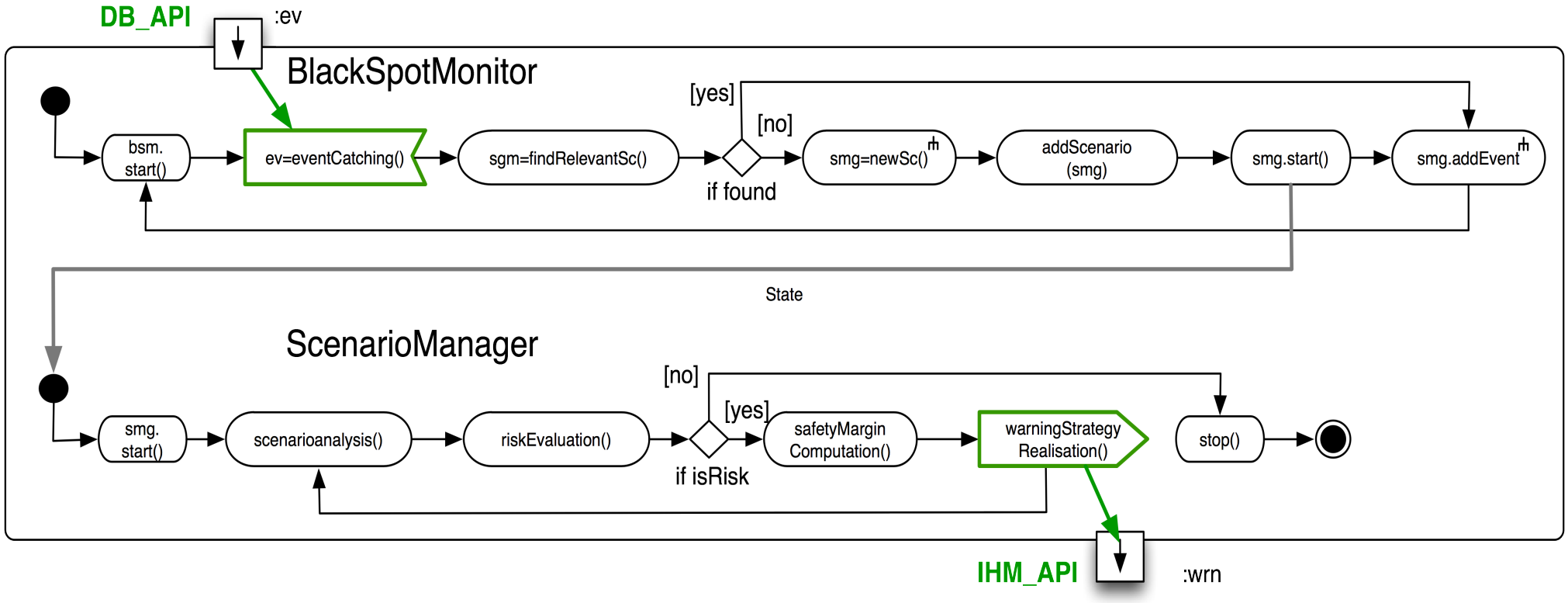
Spécification: activité de H&IW

Hazard & Incident Warning: Obstacle Warning

Le BlackSpotMonitor : contrôle une zone géographique

Le ScenarioManager : déclenche les alertes



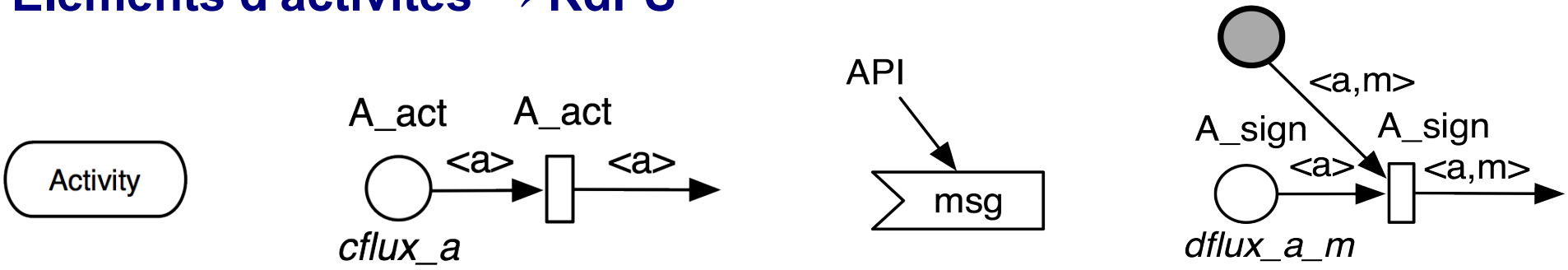


Exemples de Transformations

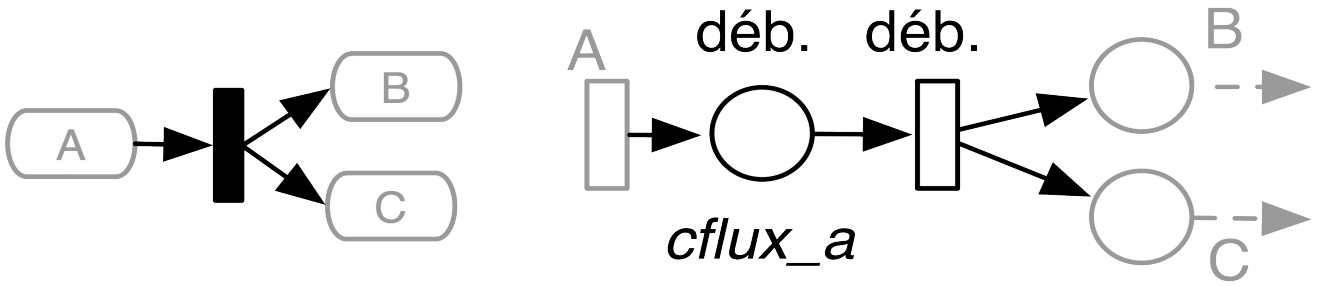
[Eshuis 2003]

- pas exploitation des symétries
- pas de composition

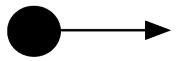
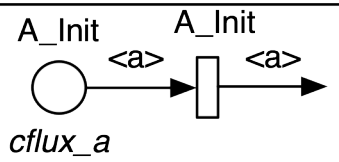
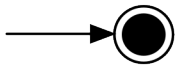
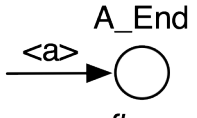
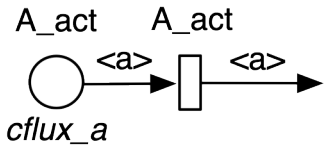
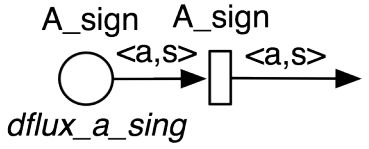
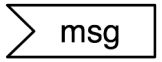
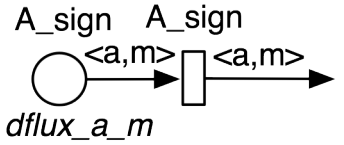
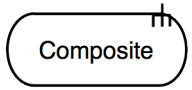
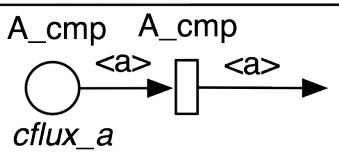
Éléments d'activités → RdPS

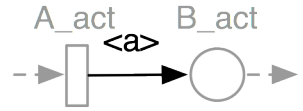
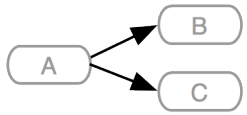
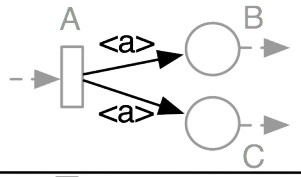
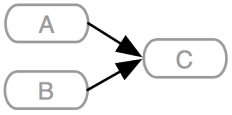
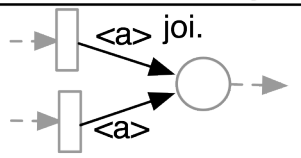
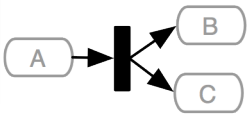
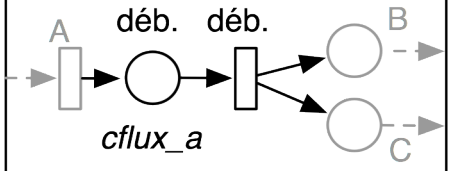
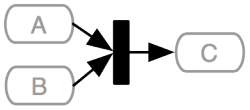
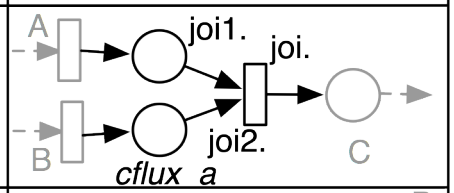
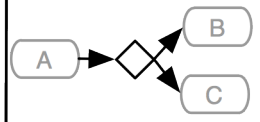
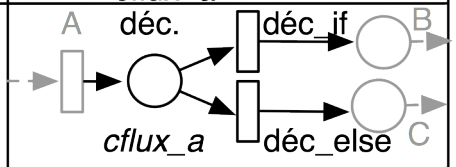
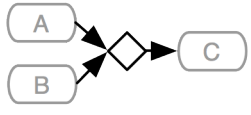
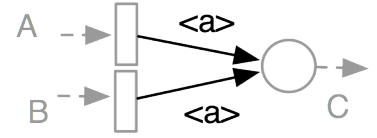


Motifs d'activités → RdPS

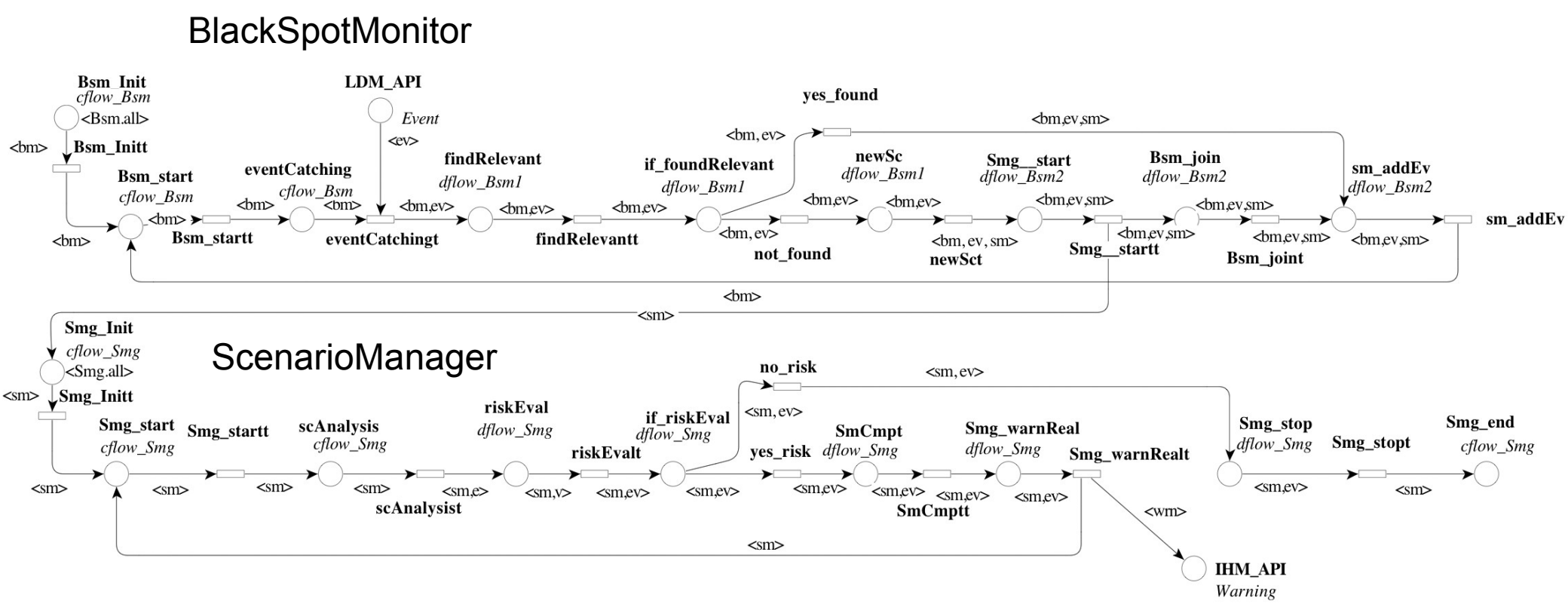


Règles de transformation

Élément d'activité	Représentation UML	Représentation Réseau Symétrique
noeud initial		
noeud final		
acivité		
envoi de signal		
reception de message		
acivité composite		

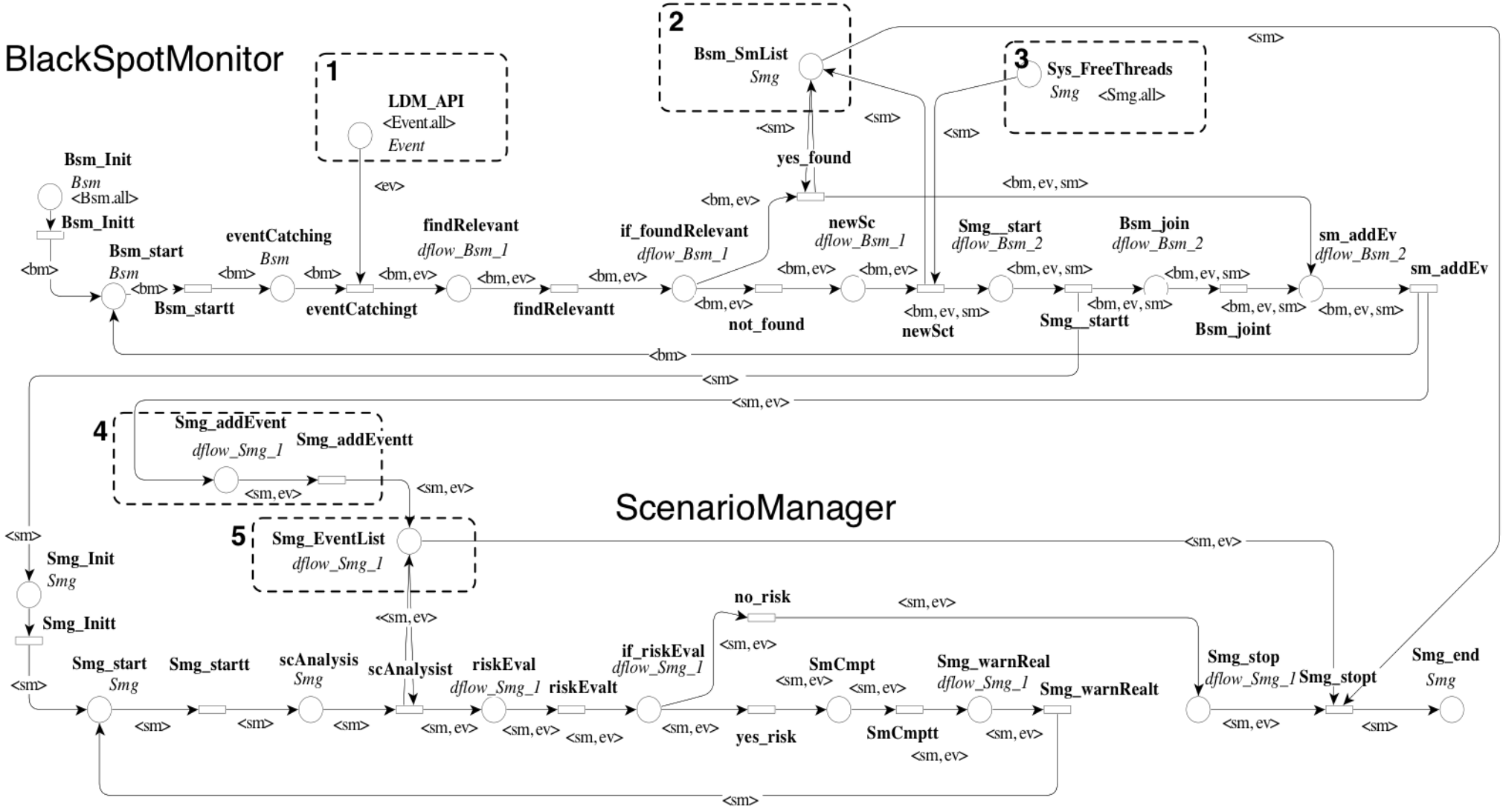
Motif d'activité	Représentation UML	Représentation Réseau Symétrique
flot de contrôle		
flot de contrôle (2 flots sortants)		
flot de contrôle (2 flots entrants)		
débranchement		
jointure		
décision		
fusion		

Modèle formel



Taille : 20 places - 18 transitions - 41 arcs

Environnement de test



Analyse et Résultats

Hazard & Incident Warning

- vérification de la politique de gestion des `threads` (formules CTL)
  → détection d'un cas ou un évènement est ignoré par le ScenarioManager
  → Ajout d'un verrou et d'un test supplémentaire

Architecture ITS

- vérification de la présence d'inter – blocage / modèle borné
- différents scénarios tests (différents conducteurs)
- problème d'explosion combinatoire

Règles de transformation

→ Production rapide des modèles

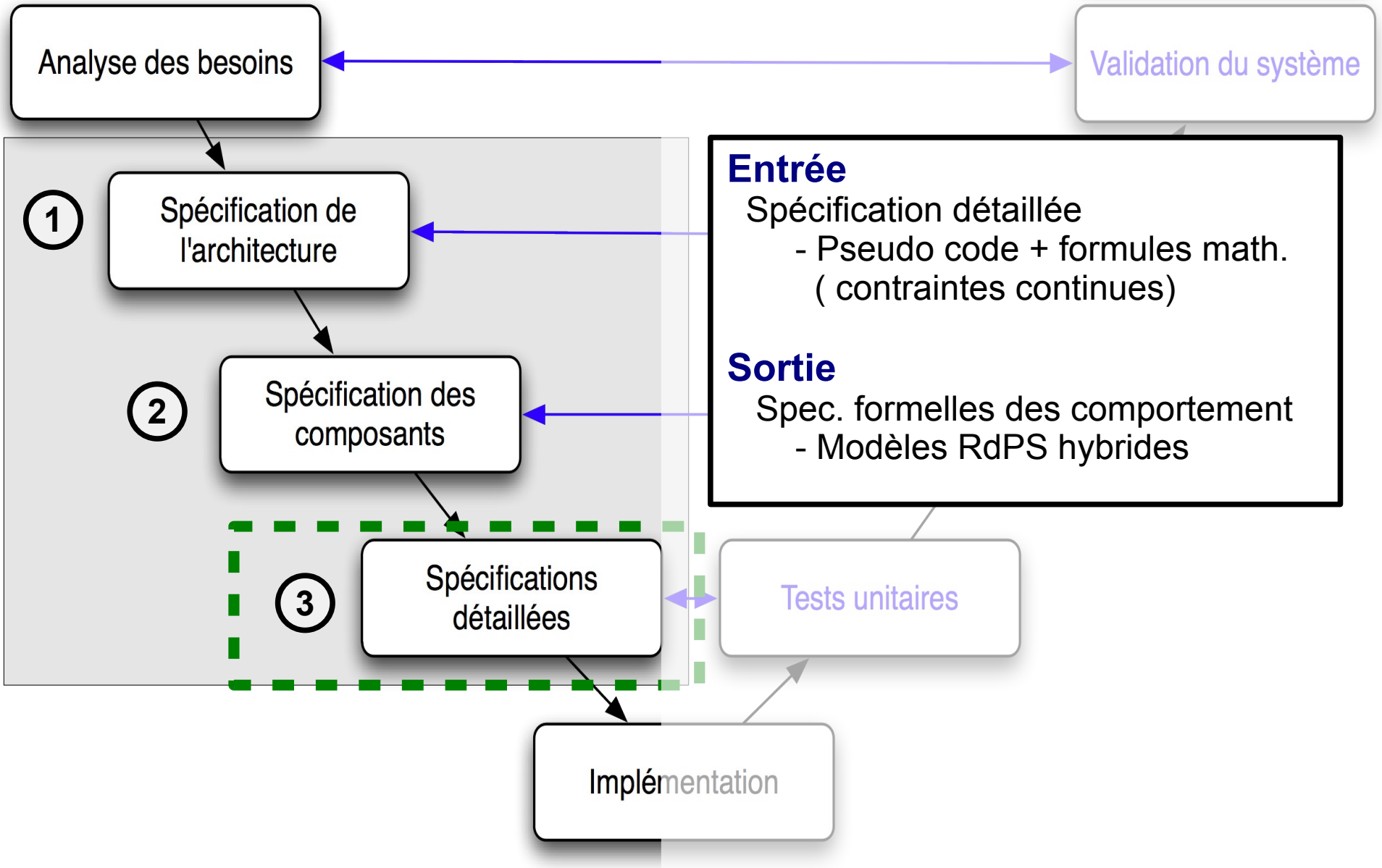
Architecture + bibliothèque

→ réutilisation des modèles
→ nouveaux scénarios d'analyse à moindre coût

Vérifications

→ modèle assemblé / composant isolé avec env. minimal
→ générales : invariants, inter-blocages
→ vérification du comportement comportement (logique temporelle)

Contribution 3



Vérification des aspects Continus

Problématique

- Des contraintes de fonctionnement impliquent des ***aspects continus***
  ex. H&IW: 18 sur 47 (38%)
- Les modèles formels continus sont difficilement décidables

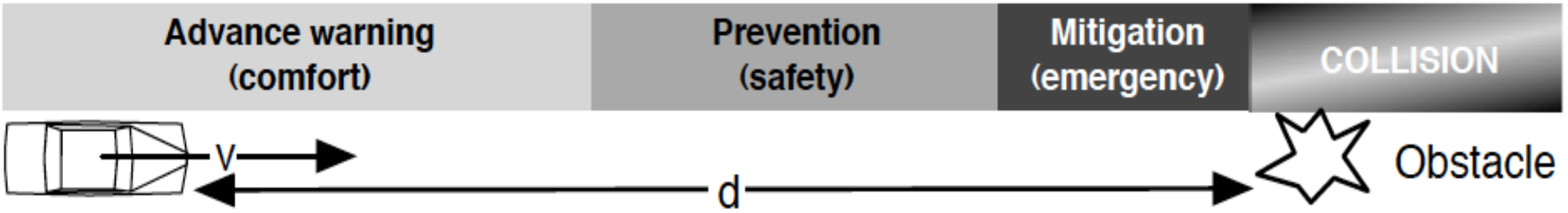
Objectifs

- Modéliser les aspects continus
- Permettre la vérification de propriétés en logique temporelle

Méthode

- Modélisation formelle des fonctions et variables continues en RdPC
- Discrétisation des aspects continus
- Modélisation formelle en RdPS

Spécification



Distance d'arrêt: $f(v,b) = \frac{v^2}{2b}$

Calcul des seuils:

$$EB\_Safety = \frac{v^2}{2b} + v * 3 - d$$

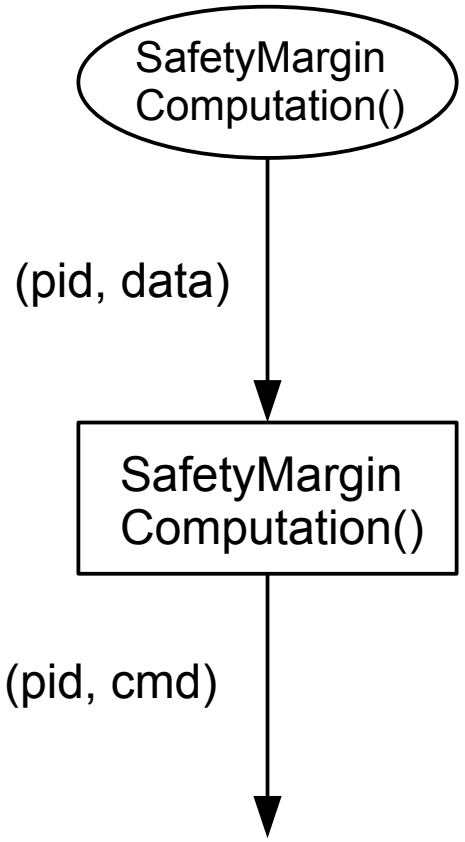
$$EB\_Emergency = \frac{v^2}{2b} + v * 1 - d$$

Pseudo-code:

```
Fonction EB_Stratégie ( v, b, d ) {
    var Eb_Emerg = V² / (2 x b) + (v x 1) – d ;
    var Eb_Safety = Eb_Emerg + (v x 2) ;
    si ( Eb_Safety < 0 ) alors          Commande = « Confort »;
    sinon si ( Eb_Safety < 0 ) alors Commande = « Sécurité »;
    sinon Commande = « Urgence »;
    retourner Commande;
}
```

Modélisation formelle en RdPC

data is (d * v * b)

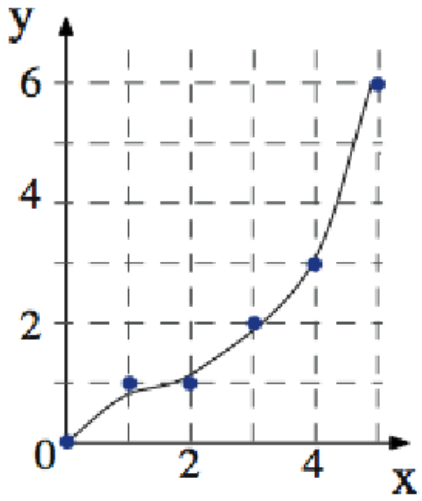



```
function Eb_Strategy(d,v,b){
  Eb_Safety = (v^2)/(2b) + v * 3 - d;
  Eb_Emergency = (v^2)/(2b) + v * 1 - d;
  if (Eb_Safety < 0) then  Command = 'Comfort';
  else if (Eb_Emergency < 0) then Command = 'Safety';
     else  Command = 'Emergency'  endif
  return Command;}
```

```
input(data)
output(cmd)
action(
let
  val c = Eb_Strategy data
in
  (c)
end);
```

Discrétisation

$$f(x) = y$$



Discrétisation à intervalles de même taille

Paramètre : k (nombre d'éléments discret pour chaque variable)

Nombre d'éléments discrets d'une fonction f(x_1.., x_n)) : $\prod_{i=1}^{n} k_i$

Propagation des incertitudes: (bornes maximales)

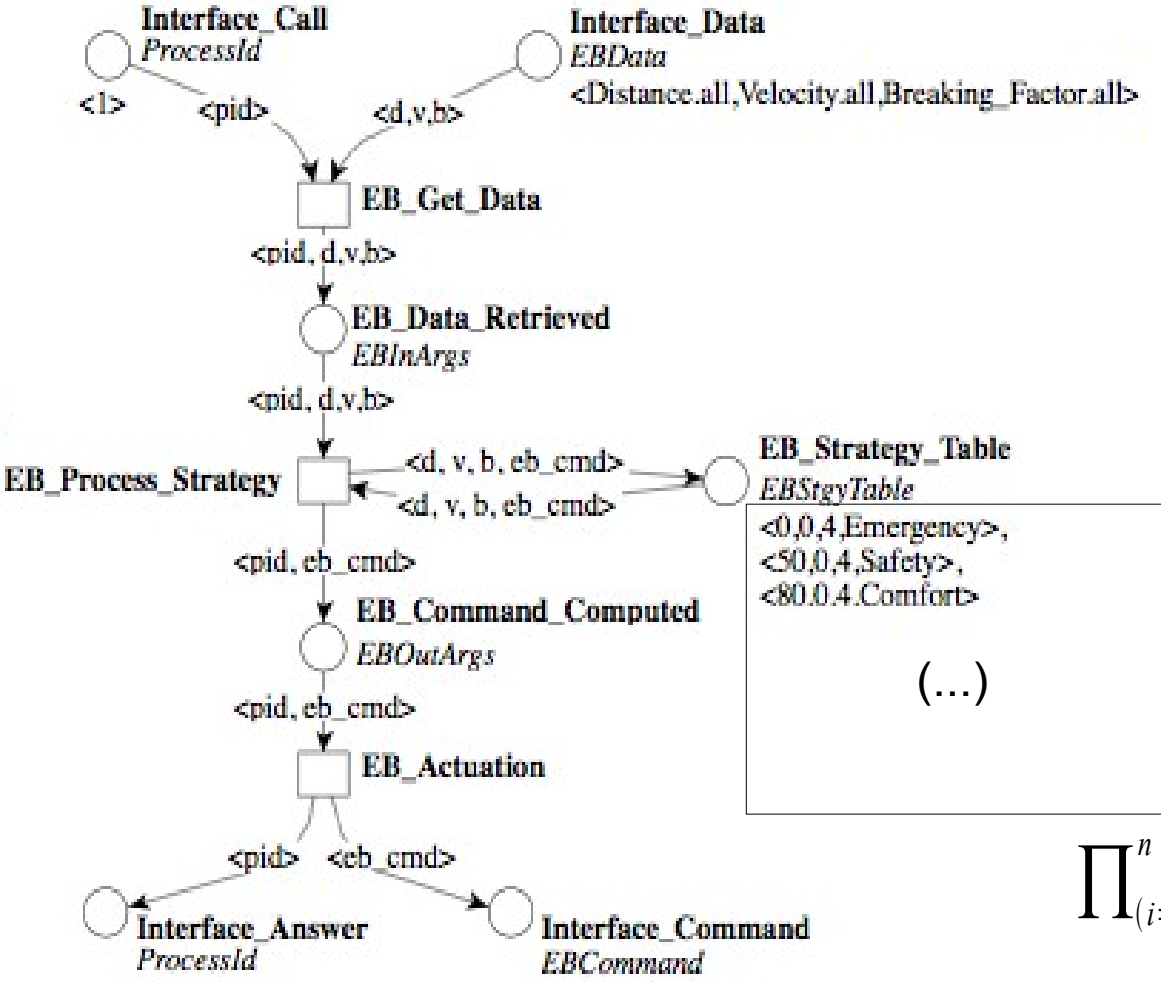
$$\Delta = \frac{x_{max} - x_{min}}{k}$$

$$\Delta_{f(x)} \in [Min(f(x \pm \Delta_x) - f(x)), Max(f(x \pm \Delta_x) - f(x))]$$

$$\Delta_f \in [Min(f(x \pm \Delta_x, y \pm \Delta_y, ..) - f(x, y, ..)), Max(f(x \pm \Delta_x, y \pm \Delta_y, ..) - f(x, y, ..))]$$

Modèle discret en RdPS

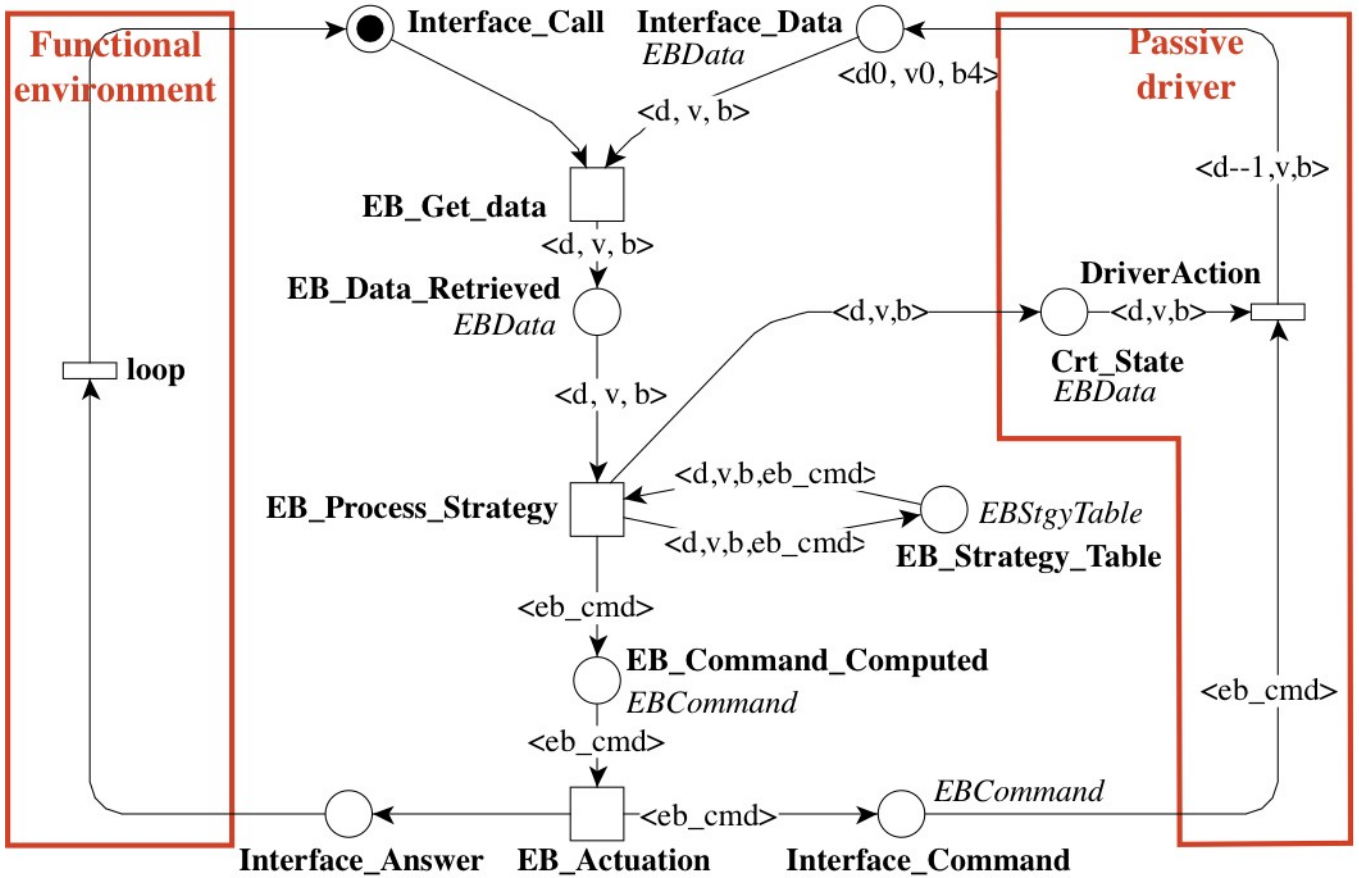
```
Class
  ProcessId is 1..1 ;
  Distance is [ _0, _50, _100, ... , _500 ];
  Velocity is [ _0, _4.6, _0.2, .., _46 ];
  Braking_Factor is [ _3, _3.6, _4.2, .., _9 ];
  EBCommand is [Comfort, Safety, Emergency] ;
Domain
  EBData is <Distance, Velocity, Braking_Factor> ;
  EBInArgs is <ProcessId, Distance, Velocity,
                Braking_Factor > ;
  EBStgyTable is <Distance, Velocity, Braking_Factor,
                  EBCommand> ;
  EBOutArgs is <ProcessId, EBCommand> ;
Var
  pid in ProcessId;
  eb_cmd in EBCommand;
  d in Distance;
  v in Velocity;
  b in Braking_Factor;
```



$$\prod_{(i=1)}^{n} k_i$$

Modèle discret en RdPS

SafetyMargin Computation



Prise en compte des incertitudes

Exemple de propagation d'incertitudes

Valeurs de v, b, d k / $card(EBData)$	$v = 13m/s,\ b = 8m/s^{-2},$ $d = 500m$	$v = 36m/s,\ b = 4m/s^{-2},$ $d = 100m$
$10 / 10^3$	$\Delta_{Eb\_Saf} \in [-70.83m, 74.84m]$ $\Delta_{Eb\_Emerg} \in [-61.64m, 65.64m]$	$\Delta_{Eb\_Saf} \in [-118.9m, 144.5m]$ $\Delta_{Eb\_Emerg} \in [-109.7m, 135.3m]$
$20 / 8 * 10^3$	$\Delta_{Eb\_Saf} \in [-35.87m, 36.81m]$ $\Delta_{Eb\_Emerg} \in [-31.27m, 32.26m]$	$\Delta_{Eb\_Saf} \in [-61.97m, 68.28m]$ $\Delta_{Eb\_Emerg} \in [-57.37m, 63.68m]$

Prise en compte des incertitudes

- Modifier les contraintes de fonctionnement
- Modifier le modèle
- Modifier les formules de logique temporelle

Exemple

AG((EB_Data_Retrieved == <13, 8, 500 >) => AX(EB_Cmd_Cpt == <Safety >))

devient

AG((EB_Data_Retrieved == <13, 8, (500 - Incert.) >) => AX(EB_Cmd_Cpt == <Safety >))

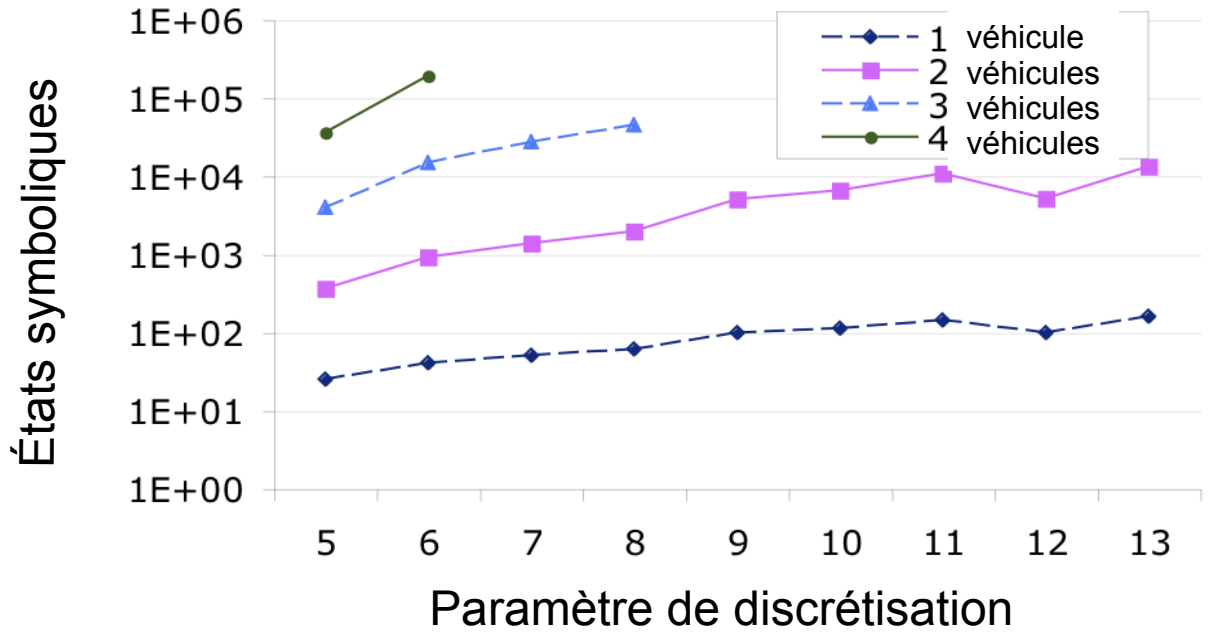
Résultats

Analyse de l'activité de calcul des alertes

- Vérification de la séquence d'envoi des alertes
  - contrainte → logique temporelle
  - prise en compte des incertitudes

Approche par discrétisation

- Vérification des aspects hybrides
- Logique temporelle / propriétés structurelles
- Problème d'explosion combinatoire



Conclusion

Vérification des spécifications de systèmes complexes par model checking

Cas d'étude

→ Système de Transport Intelligent complexe

→ Contexte industriel

Règles de transformation

→ production rapide de modèles formels

Architecture + bibliothèque

→ réutilisation des modèles

→ nouveaux scénarios d'analyse à moindre coût

Discrétisation

→ modélisation des aspects continus

→ prise en compte des incertitudes

Amélioration de la fiabilité des spécifications

Démonstration de H&IW en Juin 2010

Perspectives

Génie Logiciel

→ implémentation des règles de transformation dans un outil

→ transformation des contraintes en logique temporelle
[Lettrari & Klose: Rhapsody 2001] [Kugler: DSV 2004]

→ sémantique UML : données et fonctions

Explosion combinatoire

→ composition du modèle: Instanciable Petri Nets (ITS)

→ composition des propriétés : BIP

→ ModelCheker pour super calculateur : GsPN [Hamez 2007]