LAAS-CNRS

Analyse d'un modèle AADL à l'aide de Pola

Pierre-Emmanuel Hladik
Florent Peres
Xiaomu Shi

AFADL 2010
9-11 juin 2010 Poitiers

Contexte du travail

Vérifier les propriétés d'un système temps réel à partir de son modèle

Quel modèle ?

- Profil UML MARTE
- **AADL**
- …

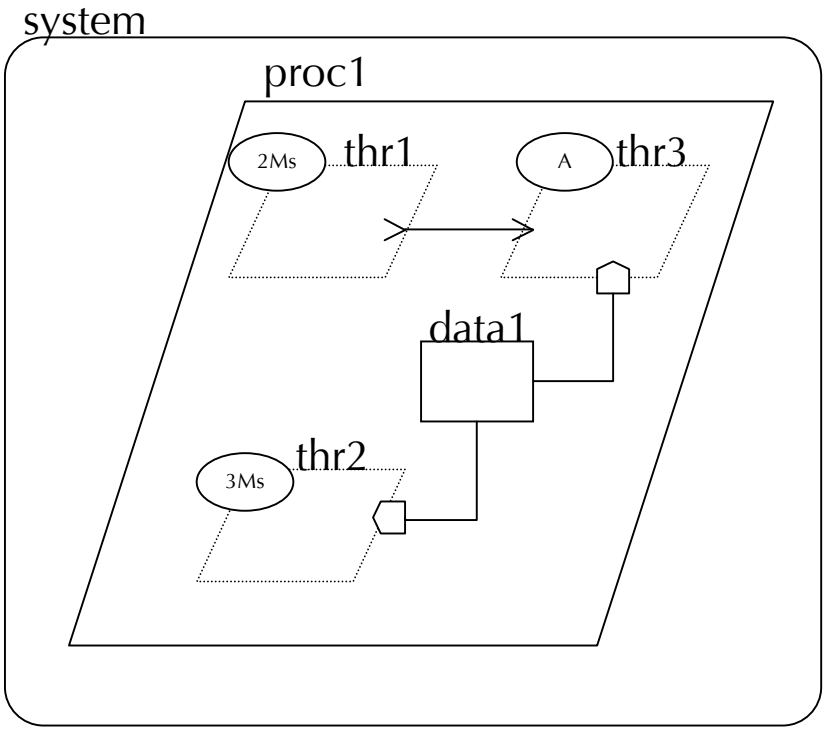
Que veut-on vérifier ?

- Vérification des exigences fonctionnelles
- **Vérification des exigences non-fonctionnelles (ex. temporelles)**
- …

AADL

- Langage dédié à la description d'architectures matérielles et logicielles de systèmes embarqués
- Standardisé par le SAE version 2.0 janvier 2009
- Langage textuel et graphique
- Langage hiérarchique à base de composants
- Un composant à une interface et des **propriétés**
- Le langage peut être facilement étendu à l'aide **d'annexes**

Exemple AADL




```
…
process proc1
end proc1;

process implementation proc1.impl;
  subcomponents
    T1: thread thr1.impl
    T2: thread thr2.impl
    T3: thread thr3.impl
  connections
    event port T1.Complete->T3.Dispatch
end proc1.impl;

…
thread thr3
  features
    requireData1: requires data access data1.impl;
  properties
    Dispatch_Protocol => Aperiodic;
    Compute_Execution_time => 1Ms..1Ms;
    Compute_Deadline=>1MS;
end thr3
```

Outils/méthodes de vérification pour AADL

Projet ITEA SPICES recense plus d'une douzaine d'outils/méthodes

Vérification d'exigences non-fonctionnelles

Simulation	Méthodes analytiques	Réécriture pour le model checking
Cheddar ADes	Cheddar MAST	BIP FIACRE IF

Utilisation d'annexes : certains standardisées (annexe comportementale), d'autres ad-hoc (propriétés Cheddar)

En règle générale les annexes ont été introduites pour ajouter de l'expressivité au modèle.

Notre approche (1/2)

Règle 1
Ne pas enrichir AADL

Que peut-on faire de AADL sans les annexes ?

Premières difficultés :

- Sémantique d'AADL n'est pas formelle
- Faire de la vérification à partir d'un langage « généraliste » est difficile

Notre approche (2/2)

Règle 2
Faire une vérification « globale »
du comportement temporel

Ne pas faire d'hypothèses a priori sur le comportement à vérifier

Première difficultés :

- Avoir les outils pour faire la vérification d'un modèle « générique »

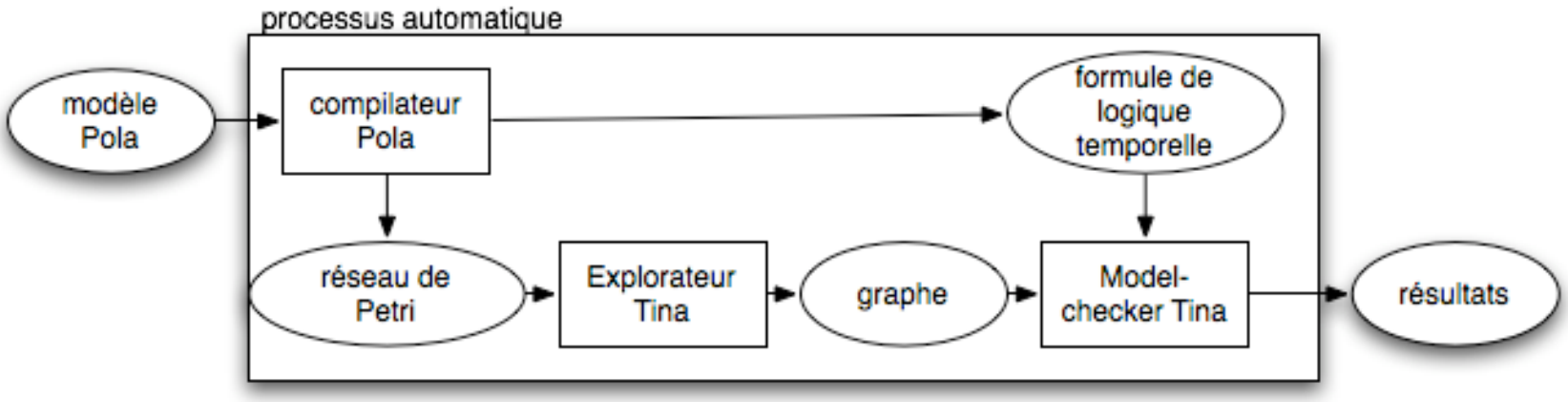
(La vraie) Règle 1
Utiliser Pola

Peut-on utiliser Pola conjointement avec un modèle haut niveau ?

Pola

Thèse F. Peres, Réseaux de Petri à inhibition/permission - Application à la modélisation et vérification de système de tâches temps réel, 2010

- Langage descriptif dédié à la spécification des systèmes temps réel
- Langage textuel (partie déclarative et partie comportementale)
- Sémantique formelle (sous forme de réseaux de Petri)
- « Model checkable » (WYSIWID : what you specify is what is done)
- Chaîne d'outils de vérification existante



Exemple Pola

```
system sys is
…
task T3 is
  action act1 in [1,1] with alloc2
  deadline 1
  policy DM
end
res proc is preemptable
res data is not preemptable…
…
policy DM is min D
allocation alloc1 is
  resources proc, data
  tasks T1, T2
allocation alloc2 is
  resources proc
  tasks T3
…
behavior is
  tr t1end -> t3begin
  lb sys.T1.act1 t1end
  lb sys.T3.released t3 begin
end
```

Déclaration des éléments

Description de l'exécution/ordonnancement

Ajout d'un comportement ad-hoc

Travail à réaliser

Transformer un modèle AADL vers Pola

…mais, AADL n'est pas formelle :

- Impossible de prouver la transformation
- Théoriquement autant de transformations possibles que d'interprétation du standard

Pourquoi passer par un langage intermédiaire et pas directement en [mettre ici le langage bas-niveau préféré] ?

- Construire sur le travail déjà réalisé autour de Pola
- Les concepts AADL et Pola sont très proches => moins d'interprétations hasardeuses

Correspondances entre les langages

Première étape
Comparaison entre les comportement des éléments des langages

System	
AADL	Pola
system *name* ... **end** *name ;*	**System** *name* **is**... **end**
actual_processor_binding => ... *applies to* ...	spéciale
Processor	
AADL	Pola
processor implementation *name* ... **end** *name ;*	**res** *name* **is preemptable**
Preemptive_Scheduler => *false*	**res** ... **is not preemptable**
Scheduling_Protocol => *DMS*	**policy** *DM* **is** *min D* + **task policy** *DM*
Scheduling_Protocol => *EDF*	**policy** *EDF* **is** *min d* + **task policy** *EDF*
Thread	
AADL	Pola
thread implementation `name` ... **end** `name ;`	**task** *name* **is**... **end**
`Dispatch_Protocol => periodic + Period => X`	**period** *[X,X]*
Dispatch_Protocol => *sporadic* + *Period* => *X*	**period** *[X,w]*
Dispatch_Protocol => *aperiodic*	**behavior** avec réseau de Petri spécifique
Compute_Execution_Time => *X..Y*	**action** ... **in** *[X,Y]*
Compute_Deadline => *X*	**deadline** *X*
Priority => *X*	**level** *X*
Data	
AADL	Pola
data *name* ... **end** *name ;*	**res** *name* **is not preemptable**
`Concurrency_Control_Protocol`	**behavior** avec réseau de Petri spécifique

Problèmes rencontrés (1/2)

La sémantique d'AADL est très faible sur certains points, en particulier les propriétés :

- Politiques d'ordonnancement
- Réveil des threads
- thread swap execution time
- …

```
Supported_Scheduling_Protocols: type enumeration (<project-specified>);

-- The following are examples of scheduling protocols: -- (FixedTimeline,
Cooperative, RMS, EDF, SporadicServer, SlackServer, ARINC653)
```

The `Supported_Scheduling_Protocols` property enumeration type specifies the set of scheduling protocols that are supported. Scheduling protocols that can be provided by implementers include:

- …
- Fixed priority server based on Rate Monotonic Scheduling (RMS) (polling server, deferrable server, sporadic server, slack stealer).

… In the case of RMS, the periodic thread policy is priority assignment according to **decreasing rate**, **aperiodic** and sporadic threads according to their minimum inter-arrival time, and background as FIFO…

De plus, sans l'annexe comportementale, il n'est pas possible de faire grand chose...

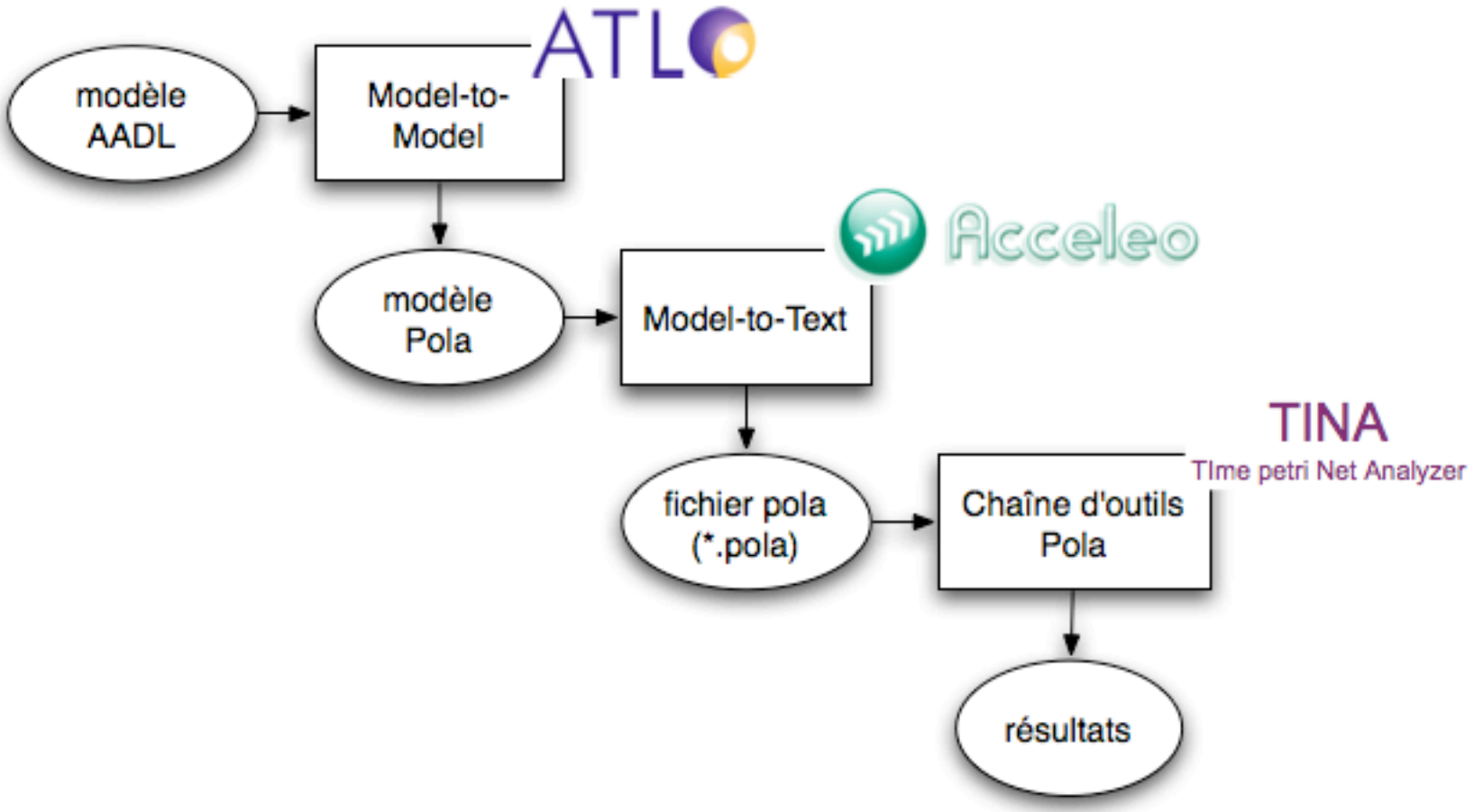
Problèmes rencontrés (2/2)

Pola n'est pas assez expressif :

- Politique d'accès aux ressources
- Politique d'ordonnancement dynamique

Définir la transformation revient à définir une sémantique formelle pour AADL

Mise en œuvre



Conclusion

- Il y a des manques dans AADL pour pouvoir faire la vérification d'un modèle ordonnancé
- Les annexes « officielles » ne suffisent pas pour combler ces manques
- Pola est très proche d'un langage métier

Qu'en conclure ?

- irréaliste : Etendre Pola pour en faire un nouvel AADL
- traditionnel : Ajouter à AADL les éléments nécessaire pour la vérification
- pragmatique : Accepter de ne pas tout spécifier correctement en AADL