

---

## Vérification automatique des modèles BPMN par transformation en réseaux de Pétri

---

Encadré par :

P<sup>r</sup> Aarab Zineb

D<sup>r</sup> Lyazidi Achraf



Soutenu le 19 juillet 2025 par :

M<sup>r</sup> Allal Yahia

Devant le jury :

P<sup>r</sup> Mouline Salma

P<sup>r</sup> Cherdal Safae

P<sup>r</sup> Aarab Zineb

D<sup>r</sup> Lyazidi Achraf

Année Universitaire : 2024-2025

# Plan

- 1 Introduction
- 2 État de l'art (MDA, BPMN, Réseaux de Petri)
- 3 Contribution : Approche de vérification automatique des modèles BPMN par transformation en réseaux de Pétri
- 4 Étude de cas
- 5 Conclusion et Perspective

# Introduction

# Contexte général

- Les entreprises doivent vraiment bien organiser et gérer toutes leurs activités.
- Plusieurs langages existent pour modéliser les processus : BPMN, UML, etc.
- Le BPMN est le langage le plus utilisé pour les processus métiers grâce à ses avantages (présentés après).
- Parfois, il arrive que certains modèles BPMN soient ambigus ou incorrects.

# Problématique

- Les entreprises font face à des processus de plus en plus complexes.
- Sans vérification, un modèle BPMN peut contenir des erreurs ou des blocages.
- Une petite erreur peut bloquer ou ralentir tout un processus.
- Vérifier à la main, c'est difficile et prend du temps.



**Comment être sûr qu'un modèle BPMN est correct et sans erreur cachée ?**

# Objectifs du projet

- **Fiabiliser les modèles BPMN** : vérification rigoureuse des processus.
- **Transformation automatique** : conversion BPMN vers réseaux de Pétri.
- **Analyse formelle** : détection de blocages, vivacité, terminaison.
- **Outil web intuitif** : accessible aux utilisateurs non-experts.
- **Validation par une étude de cas** : démonstration concrète de l'approche.

# État de l'art (MDA, BPMN, Réseaux de Petri)

# Ingénierie Dirigée par les Modèles (IDM/MDE)

## Ingénierie Dirigée par les Modèles (IDM/MDE)

*"Approche utilisant des modèles comme artefacts principaux plutôt que le code, où :*

- *Les modèles sont formels et exécutables*
- *Les transformations entre modèles sont automatisées*
- *La traçabilité des exigences est garantie"*

*(Combemale et al., 2016)*

## Application du standard MDA de OMG :

- **PIM** (Platform-Independent Model) : Modèles BPMN
- **PSM** (Platform-Specific Model) : Réseaux de Pétri

### Références :

Combemale, B., et al. (2016). *Engineering Modeling Languages*. CRC Press.

Object Management Group. (2014). *MDA Guide rev. 2.0*

# Qu'est-ce que le BPMN ?

## Définition simple

Le BPMN (Business Process Model and Notation) est un langage graphique standard utilisé pour modéliser et représenter les processus métier. Il permet de décrire visuellement les étapes d'un processus.

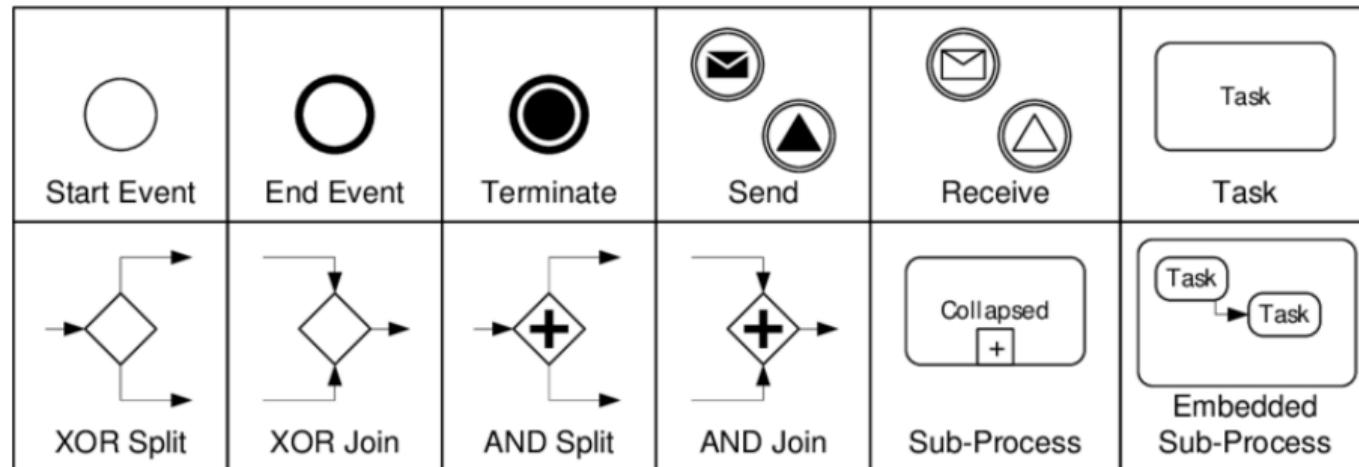


Figure 1 – Principaux symboles de base du BPMN

# Exemple BPMN

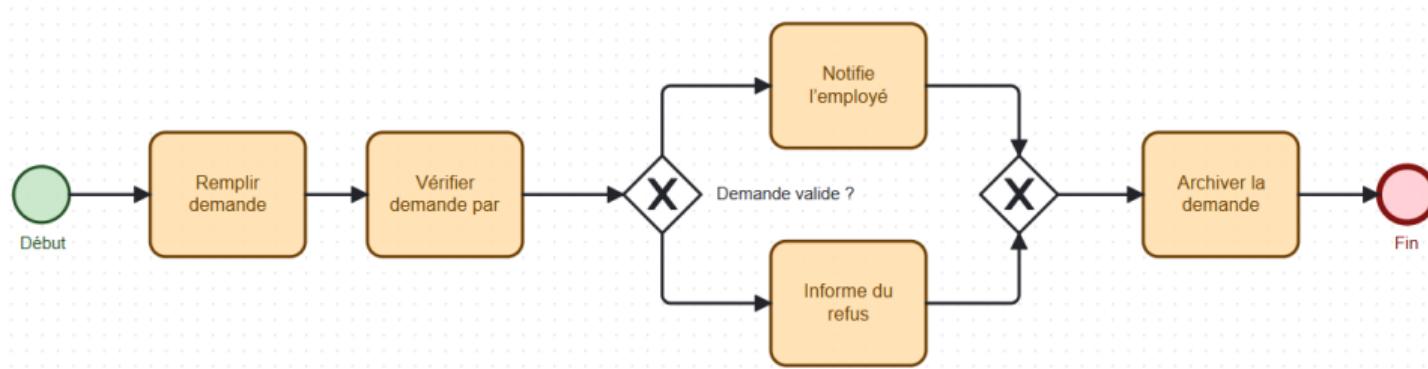


Figure 2 – Exemple BPMN au processus de demande de congé

# Avantages de BPMN

- ✓ **Standard international** : facilite l'échange entre entreprises.
- ✓ **Facile à lire et à comprendre** : utilise des symboles graphiques simples.
- ✓ **Communication claire** entre les équipes (métier/technique).
- ✓ **Simulation et automatisation possibles.**

## À retenir

Donc, c'est un **choix logique** pour modéliser les processus métiers.

# Avantages de BPMN

- ✓ **Standard international** : facilite l'échange entre entreprises.
- ✓ **Facile à lire et à comprendre** : utilise des symboles graphiques simples.
- ✓ **Communication claire** entre les équipes (métier/technique).
- ✓ **Simulation et automatisation possibles.**

## À retenir

Donc, c'est un **choix logique** pour modéliser les processus métiers.

# Limites du BPMN

- ✗ Il manque de précision : parfois, le sens du modèle n'est pas clair.
- ✗ On ne peut pas vérifier automatiquement s'il y a des erreurs ou des blocages.
- ✗ Quand un processus est trop grand, le schéma devient vite difficile à lire.

## À noter

Pour être sûr que tout fonctionne bien, il faut utiliser d'autres outils pour vérifier les modèles.

→ C'est pourquoi on utilise les réseaux de Pétri.

# Limites du BPMN

- ✗ Il manque de précision : parfois, le sens du modèle n'est pas clair.
- ✗ On ne peut pas vérifier automatiquement s'il y a des erreurs ou des blocages.
- ✗ Quand un processus est trop grand, le schéma devient vite difficile à lire.

## À noter

Pour être sûr que tout fonctionne bien, il faut utiliser d'autres outils pour vérifier les modèles.

→ **C'est pourquoi on utilise les réseaux de Pétri.**

# Les réseaux de Pétri

## Définition

Un réseau de Pétri est un dessin qui sert à représenter le déroulement d'un processus, en utilisant trois éléments : les places, les transitions et les arcs.

Symbol	Nom	Description
	Place (cercle)	État ou situation
	Transition (carré)	Action ou événement
	Arc (flèche)	Relie places et transitions

Table 1 – Éléments de base d'un réseau de Pétri

# Exemple de réseau de Pétri

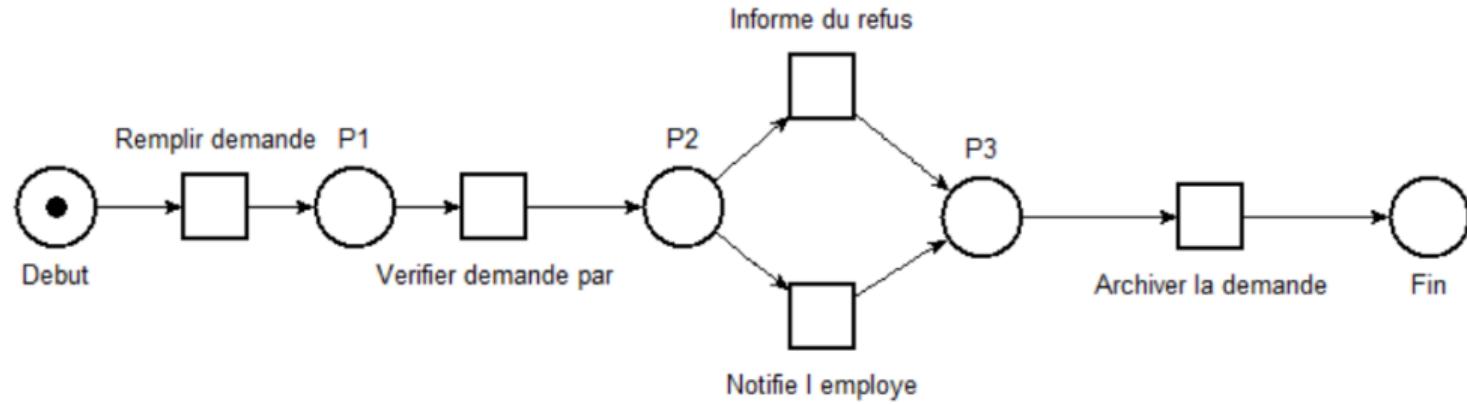


Figure 3 – Exemple : Réseau de Pétri correspondant au processus de demande de congé

# Types de réseaux de Pétri

Type de réseau	Description	Objectif principal
De base	Places, transitions, arcs	Vérification structurelle
Coloré (CPN)	Jetons avec données	Intégration de données métier
Temporel	Transitions avec délais	Analyse temporelle
Stochastique (GSPN)	Transitions probabilistes	Évaluation de performance
Hiérarchique	Sous-réseaux imbriqués	Modularité

Table 2 – Types de réseaux de Pétri

# Choix du Réseau de Pétri utilisé

## Modèle choisi et justification

**Modèle choisi :** Réseaux de Pétri

**Pourquoi ?** Le plus utilisé dans la recherche pour l'analyse formelle. Basé sur des fondements mathématiques solides, il permet une modélisation précise et vérifiable.

## Type utilisé

**Type utilisé :** Réseau de Pétri de **base** (places, transitions, arcs)

**Pourquoi ?** Simple, efficace, idéal pour détecter rapidement les erreurs et blocages sans complexifier l'analyse.

# Choix du Réseau de Pétri utilisé

## Modèle choisi et justification

**Modèle choisi :** Réseaux de Pétri

**Pourquoi ?** Le plus utilisé dans la recherche pour l'analyse formelle. Basé sur des fondements mathématiques solides, il permet une modélisation précise et vérifiable.

## Type utilisé

**Type utilisé :** Réseau de Pétri de **base** (places, transitions, arcs)

**Pourquoi ?** Simple, efficace, idéal pour détecter rapidement les erreurs et blocages sans complexifier l'analyse.

# Synthèse sur les travaux existants

- Plusieurs méthodes ont été proposées pour transformer les modèles BPMN en réseaux de Pétri.

Méthode	Références
Transformation directe	Dijkman et al.
Modèles simplifiés	Mouline, Lyazidi
Réseaux colorés	Ramadan, Meghzili, Dechsupa
Réseaux hiérarchiques	Travaux récents

Table 3 – Méthodes proposées pour la transformation  
BPMN en réseaux de Pétri



# Forces et limites des approches existantes

## Points forts

- **Vérification formelle** des processus
- **Détection des erreurs** et blocages

## Limites

- Prise en charge incomplète de tous les cas BPMN
- Outils souvent complexes à utiliser
- Peu de solutions simples et accessibles à tous

## Conclusion

Aucune méthode ne répond totalement aux besoins, notamment en termes de prise en charge complète des éléments BPMN, d'automatisation simple et de vérification intuitive.

# Forces et limites des approches existantes

## Points forts

- **Vérification formelle** des processus
- **Détection des erreurs** et blocages

## Limites

- Prise en charge incomplète de tous les cas BPMN
- Outils souvent complexes à utiliser
- Peu de solutions simples et accessibles à tous

## Conclusion

Aucune méthode ne répond totalement aux besoins, notamment en termes de prise en charge complète des éléments BPMN, d'automatisation simple et de vérification intuitive.

# Forces et limites des approches existantes

## Points forts

- **Vérification formelle** des processus
- **Détection des erreurs** et blocages

## Limites

- Prise en charge incomplète de tous les cas BPMN
- Outils souvent complexes à utiliser
- Peu de solutions simples et accessibles à tous

## Conclusion

**Aucune méthode ne répond totalement aux besoins, notamment en termes de prise en charge complète des éléments BPMN, d'automatisation simple et de vérification intuitive.**

## **Contribution : Approche de vérification automatique des modèles BPMN par transformation en réseaux de Pétri**

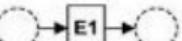
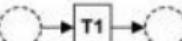
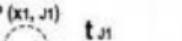
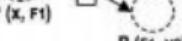
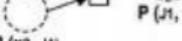
## Solution proposée

- **Transformation simple** : Mapping BPMN vers Petri clair et direct.
- **Transformation complète** : Gestion des éléments avancés (XOR, AND ).
- **Application Web Automatisée** : Transformation fluide et performante.
- **Vérification Formelle** : Validation via TINA pour vérifier la vivacité, bornitude, etc.
- **Simplicité d'utilisation** : Interface web intuitive pour utilisateurs non-experts.

# Mapping entre BPMN et RdP (Inspiré de Dijkman & Fyama)

## Approche inspirante

Méthode simple, claire et efficace pour le mapping BPMN-RdP

BPMN Object	Petri-net Module	BPMN Object	Petri-net Module	BPMN Object
 $y$ Start s	 $P_s \xrightarrow{t_s} P(s, y)$	 $x$ End e	 $P(x, e) \xrightarrow{t_e} P(e, y)$	 $x \rightarrow y$ Message E
 $y$ Message E	 $P(x, E1) \xrightarrow{E1} P(E1, y)$	 $x \rightarrow y$ Task T	 $P(x, T1) \xrightarrow{T1} P(T1, y)$	 $x \rightarrow y_1, y_2$ Fork F1
 $y_1, y_2$ Fork F1	 $P(x, F1) \xrightarrow{t_{F1}} P(F1, y_1), P(F1, y_2)$	 $x_1, x_2 \rightarrow y$ Join J1	 $P(x_1, J1) \xrightarrow{t_{J1}} P(x_2, J1) \xrightarrow{t_{J1}} P(J1, y)$	 $x \rightarrow y$ Merge M1
 $c \rightarrow y_1, -c \rightarrow y_2$ (Data-based) Decision D1	 $P(x, D1) \xrightarrow{t_{D1, y_1}} P(D1, y_1), P(x, D1) \xrightarrow{t_{D1, y_2}} P(D1, y_2)$	 $x_1, x_2 \rightarrow y$ Decision D1	 $P(x_1, M1) \xrightarrow{t_{M1, x_1}} P(x_2, M1) \xrightarrow{t_{M1, x_2}} P(M1, y)$	 $x \rightarrow \text{Message } E1$ (Event-based) Decision V1
				 $\text{Message } E1 \rightarrow T1 \rightarrow y_2$ Receive task T1
				 $P(x, V1) \xrightarrow{T1} P(T1, y_2)$

[ Note ]:

$x, x_1$  or  $x_2$  represents an input object, and  $y, y_1$  or  $y_2$  represents an output object.

Figure 4 – Mapping entre BPMN et Réseaux de Pétri (adapté de Dijkman & Fyama).

## Exemple 1 : Start Event suivi d'une tâche

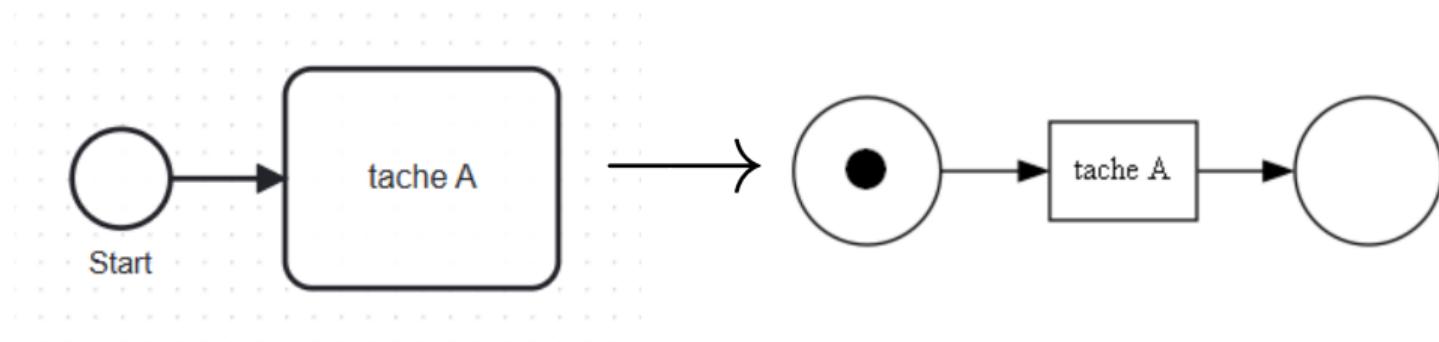


Figure 5 – Mapping du Start Event suivi d'une tâche.

## Exemple 2 : Passerelle XOR suivie de deux tâches

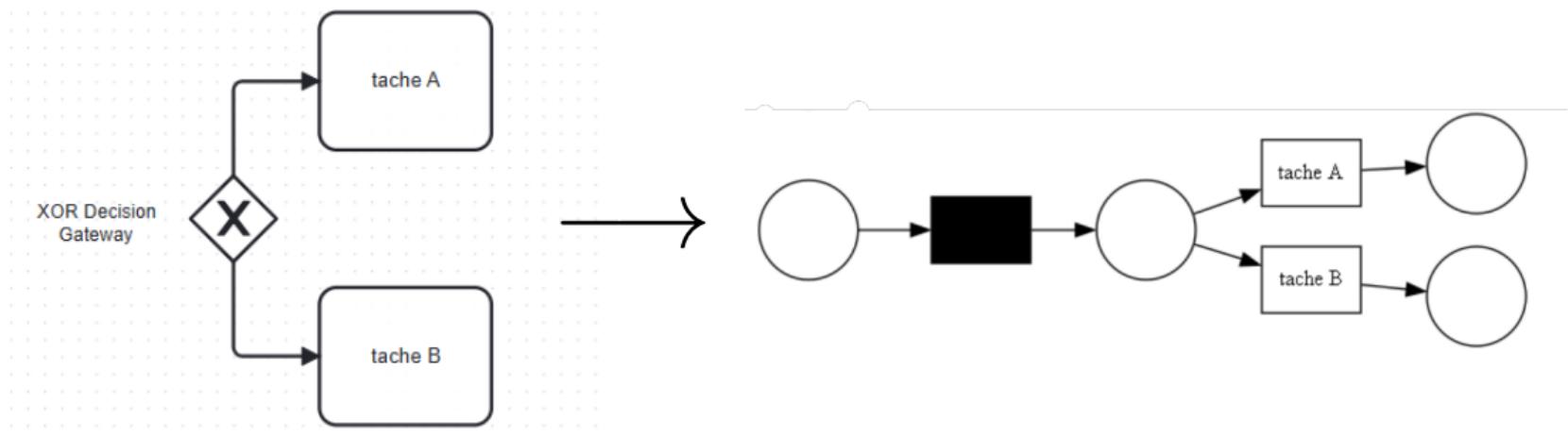


Figure 6 – Mapping de la passerelle XOR suivie de deux tâches.

# Outils et technologies utilisés

## Backend



Figure 7 – Python



Figure 8 – FastAPI



Figure 9 – pm4py

## Frontend

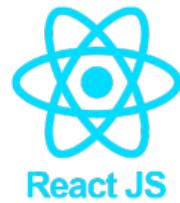


Figure 10 – React.js

# Interface d'accueil

**YAHIA**

Accueil      Transformation      Contactez-nous

## Transformation de Model BPMN

Choisissez un fichier d'extension **BPMN**

**Choisir fichier...**      Transformer

```
graph LR; Start(( )) --> AssignApprover[Assign Approver]; AssignApprover --> ApproveService[Approve Service]; ApproveService --> DecisionApprove{Decision approved?}; DecisionApprove --> RechPlan[Recherche plan]; RechPlan --> DecisionSuccess{Is successful?}; DecisionSuccess --> End1((( )));
    DecisionSuccess --> PBT[Prepare Back Transfer];
    PBT --> ArchiveService[Archive Service];
    ArchiveService --> End2((( )));
    DecisionApprove --> End1;
```

Exemple de BPMN 2.0

Figure 11 – Interface d'accueil de l'application.

# Interface de transformation

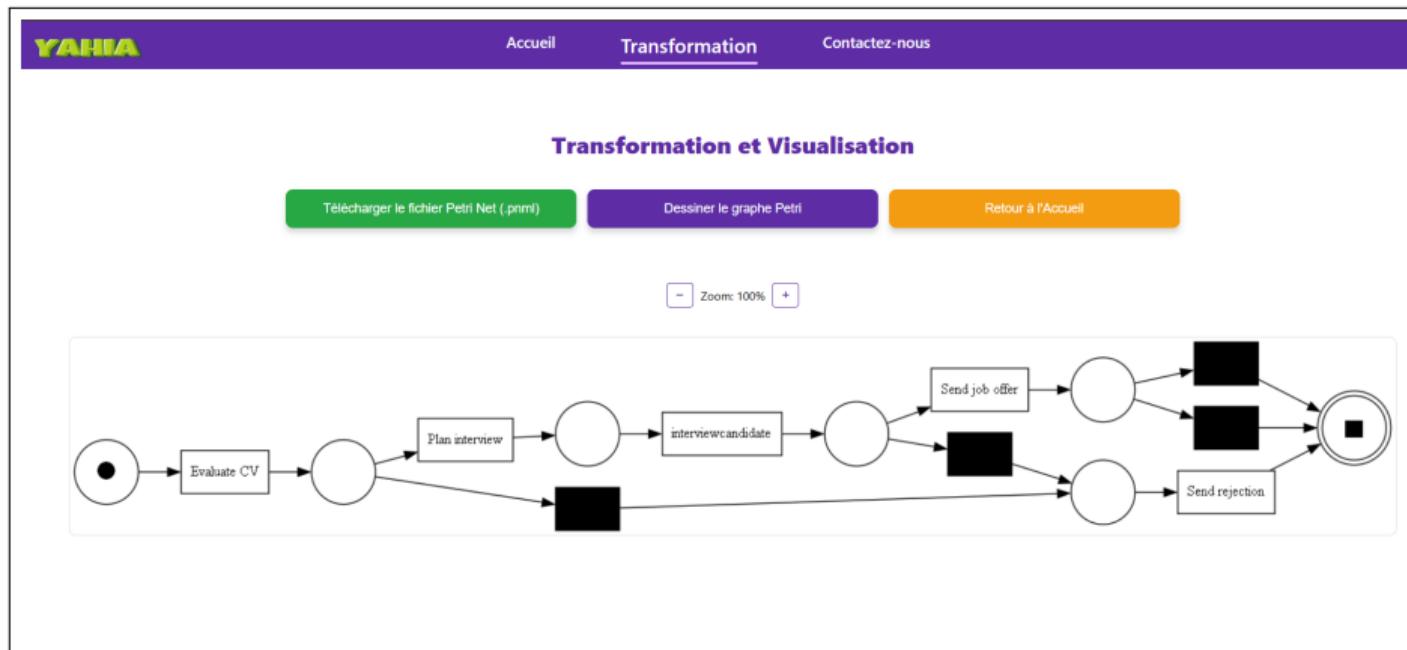


Figure 12 – Interface de transformation.

# Interface de contact

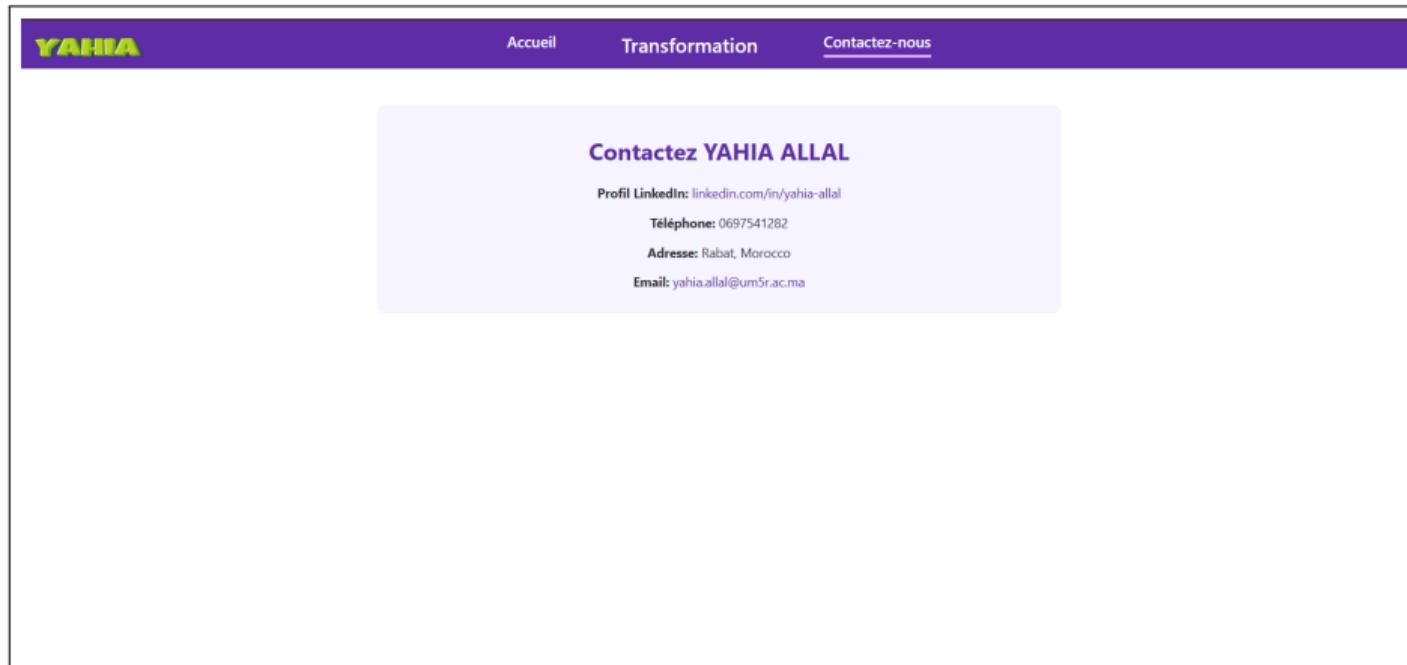


Figure 13 – Interface de contact de l'application.

# Analyse des modèles : TINA (Tlme Petri Net Analyzer)

- **Facilité d'utilisation** : Interface simple.
- **Dessiner graphiquement** : Modification de réseaux PNML.
- **Analyse complète** : Édition, simulation, vérification.
- **Simulation dynamique** : Suivi des transitions.
- **Vérification formelle** : Vivacité, bornitude, blocages.
- **Analyse rapide** : Outils d'énumération et structurelle.

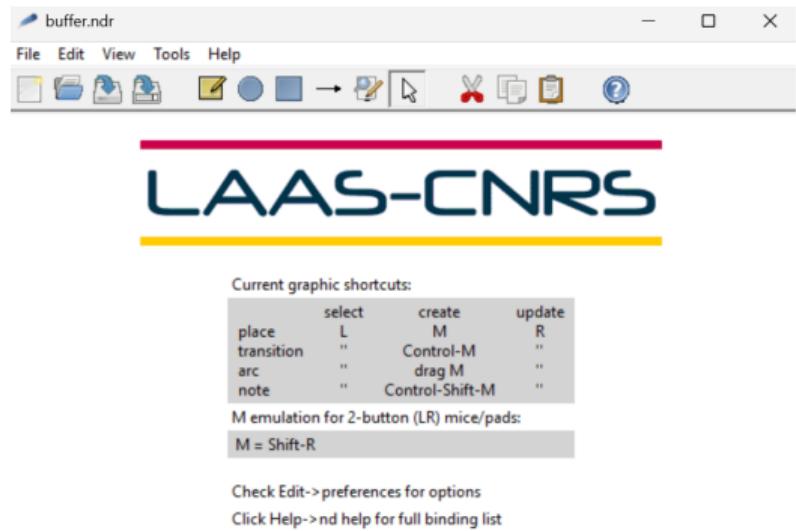


Figure 14 – Interface de l'application TINA

# Vérification des propriétés du modèle

Propriété	Description
Absence de blocages	Pas d'arrêt définitif
Vivacité	Activation des transitions
Accessibilité	Atteindre tous les états
Bornitude	Limiter les jetons
Validité	Absence d'erreurs

Table 4 – Propriétés critiques vérifiées dans le réseau de Pétri

# Étude de cas

# Exemple de résolution d'une équation du second degré

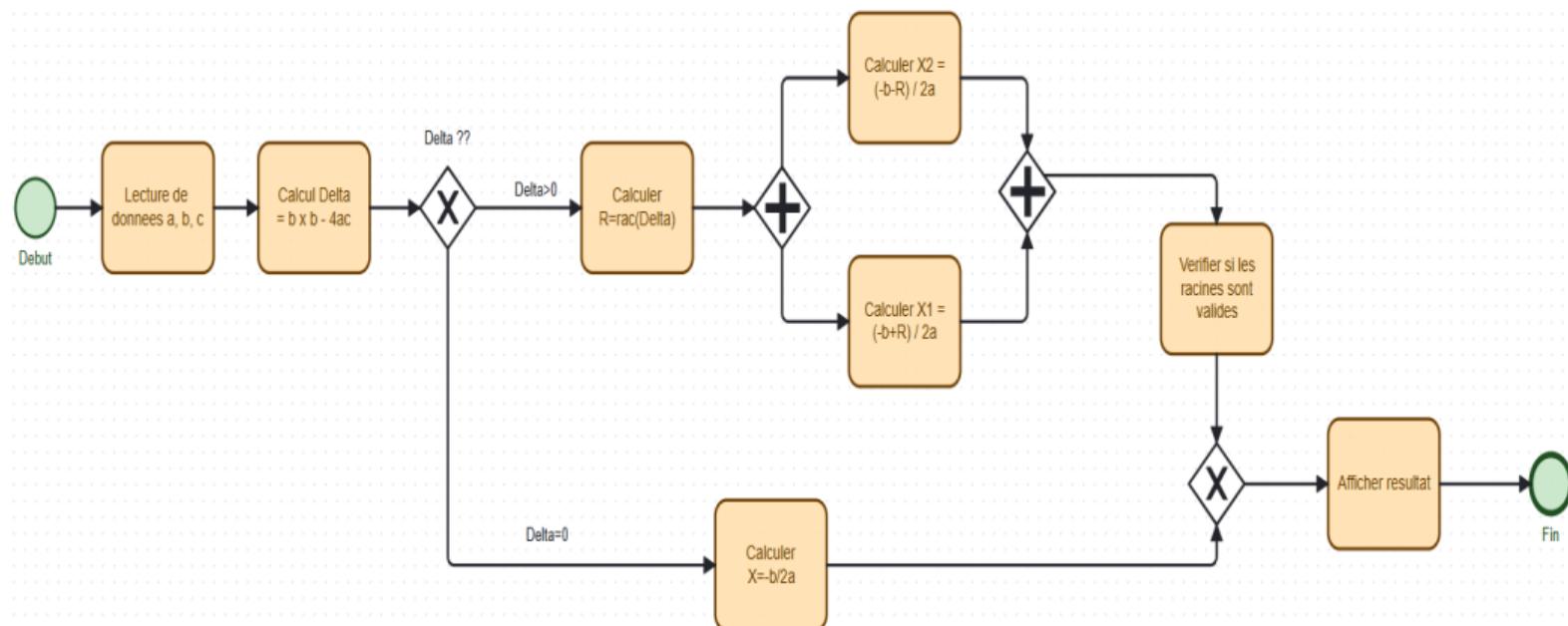


Figure 15 – Modèle BPMN de La résolution d'une équation du second degré.

# Modèle BPMN : Blocage

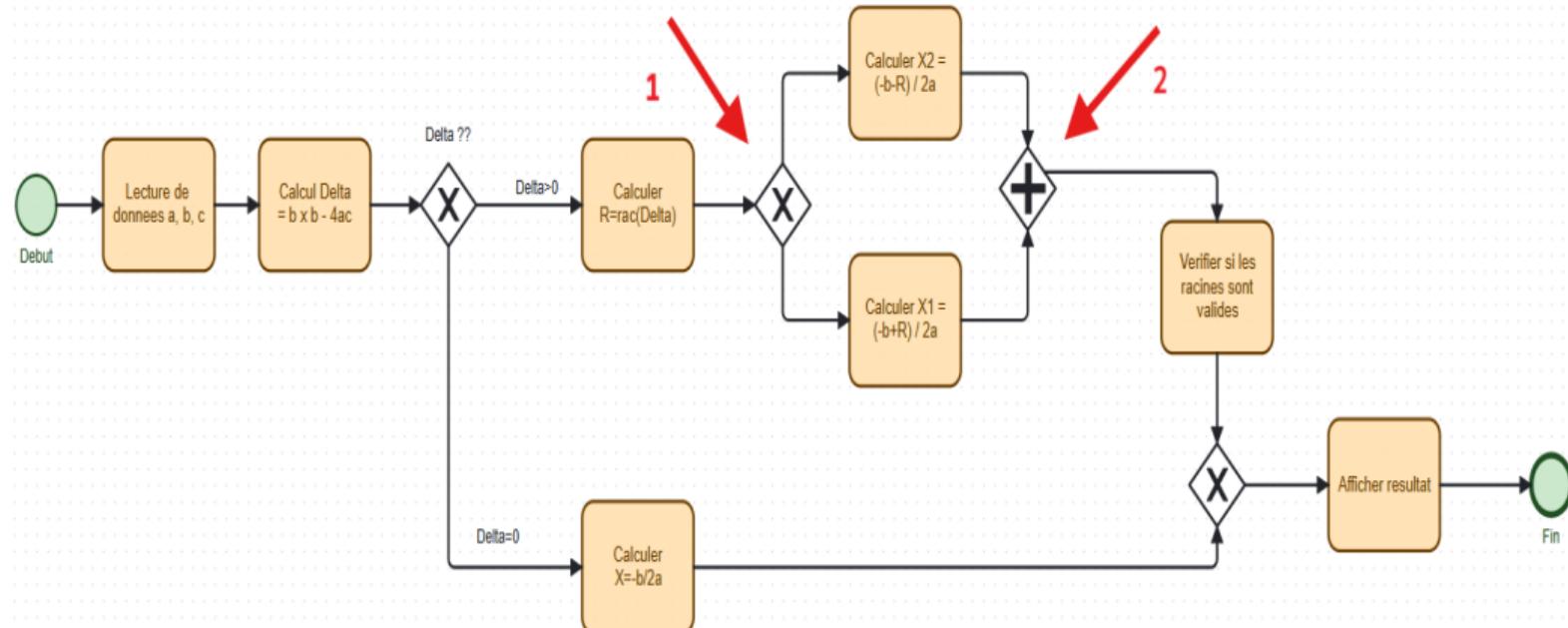


Figure 16 – Modèle BPMN illustrant un blocage dû à une mauvaise gestion des passerelles XOR et AND.

# Réseau de Pétri : Blocage

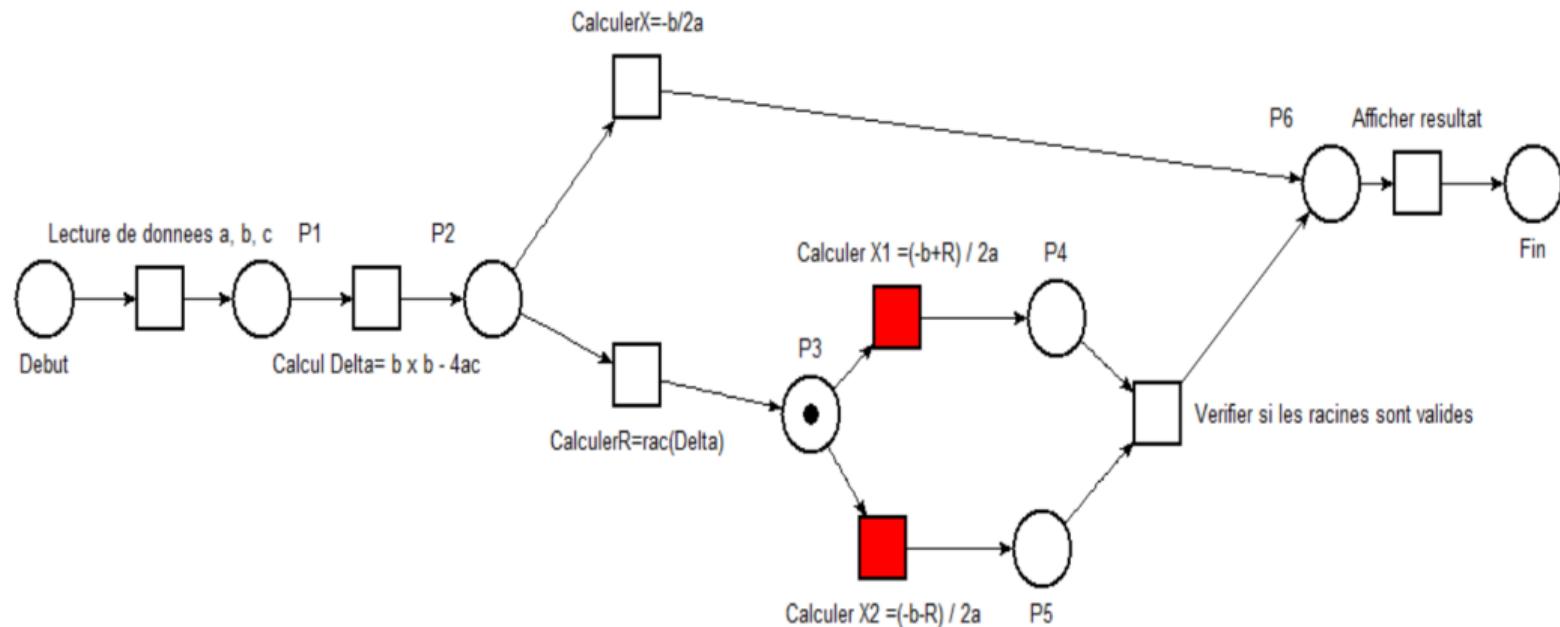


Figure 17 – Réseau de Pétri correspondant au modèle BPMN avec problème blocage

# Réseau de Pétri : Blocage

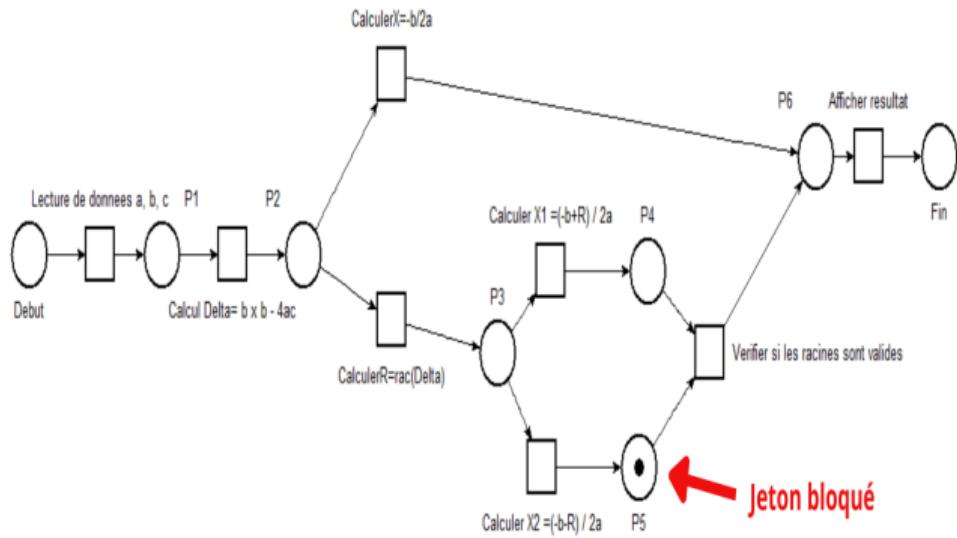


Figure 18 – Réseau de Pétri correspondant au modèle BPMN avec problème blocage.

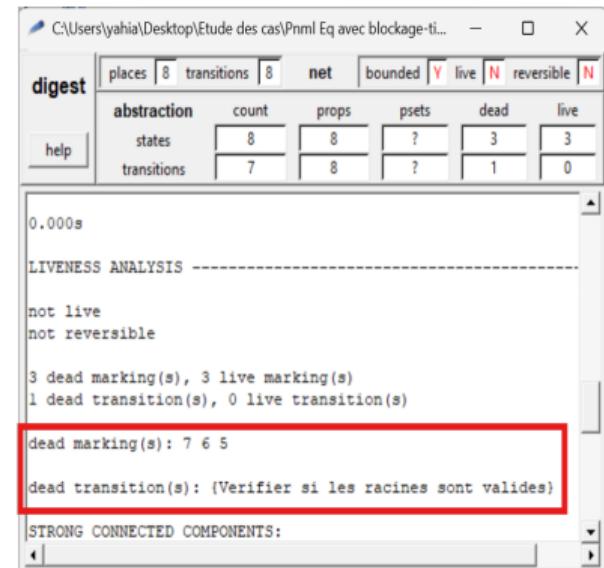


Figure 19 – Tableau d'analyse

# Modèle BPMN : Accessibilité

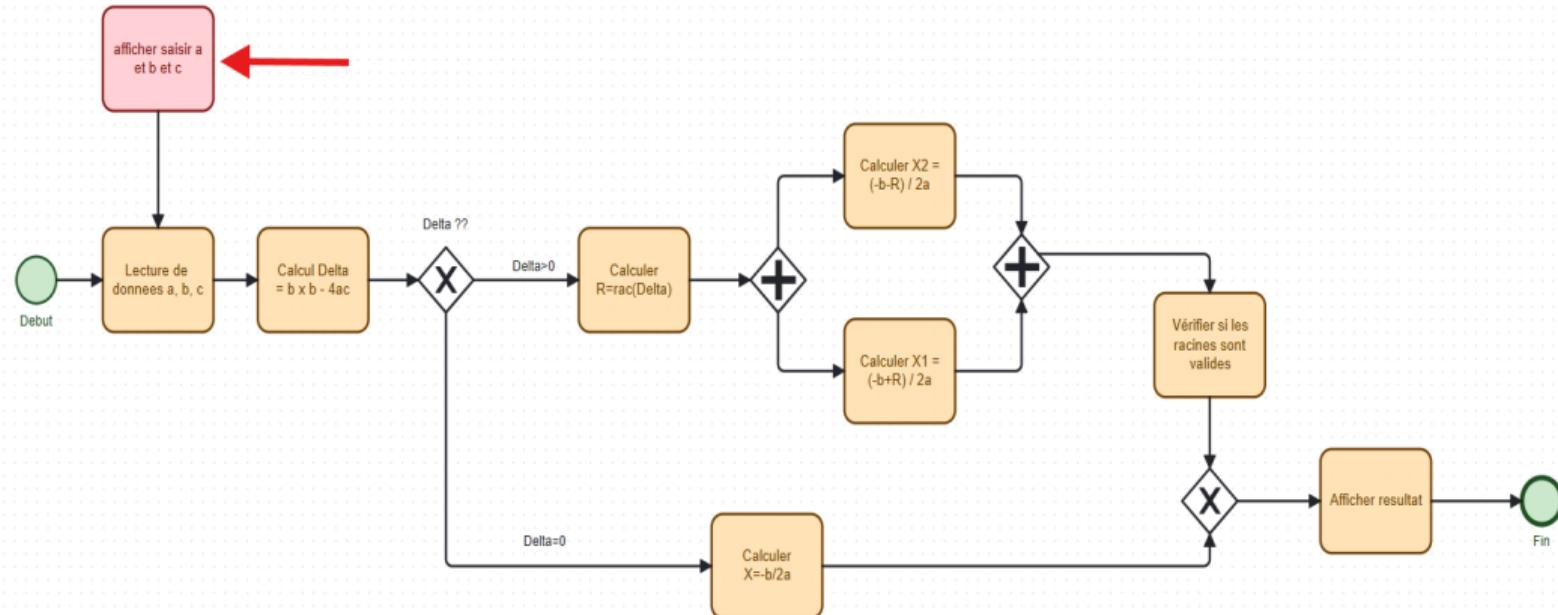


Figure 20 – Modèle BPMN illustrant un problème d'accessibilité.

# Réseau de Pétri : Accessibilité

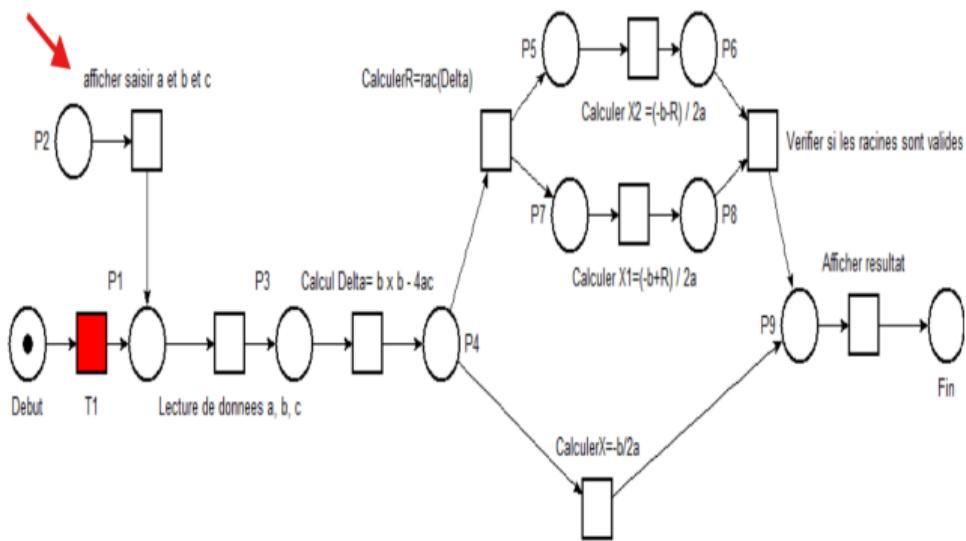


Figure 21 – Réseau de Pétri correspondant au modèle BPMN avec problème d'accessibilité.

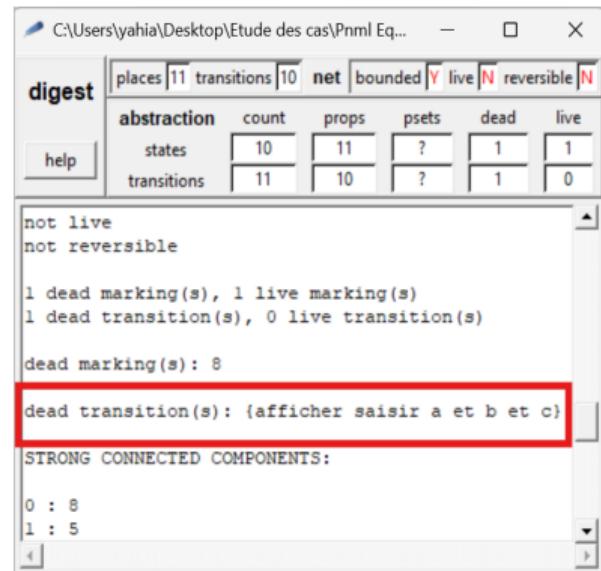


Figure 22 – Tableau d'analyse

# Modèle BPMN : Bornitude

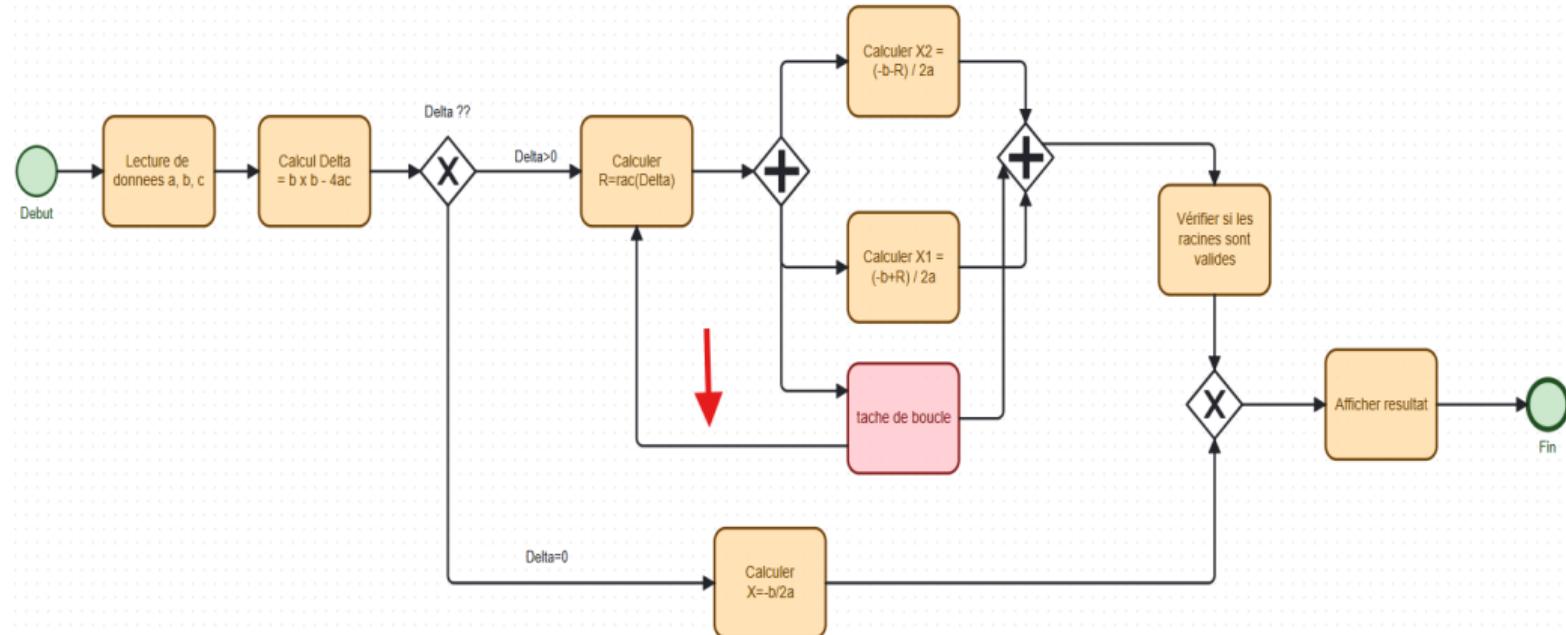


Figure 23 – Modèle BPMN avec une boucle potentiellement non bornée.

# Réseau de Pétri : Bornitude

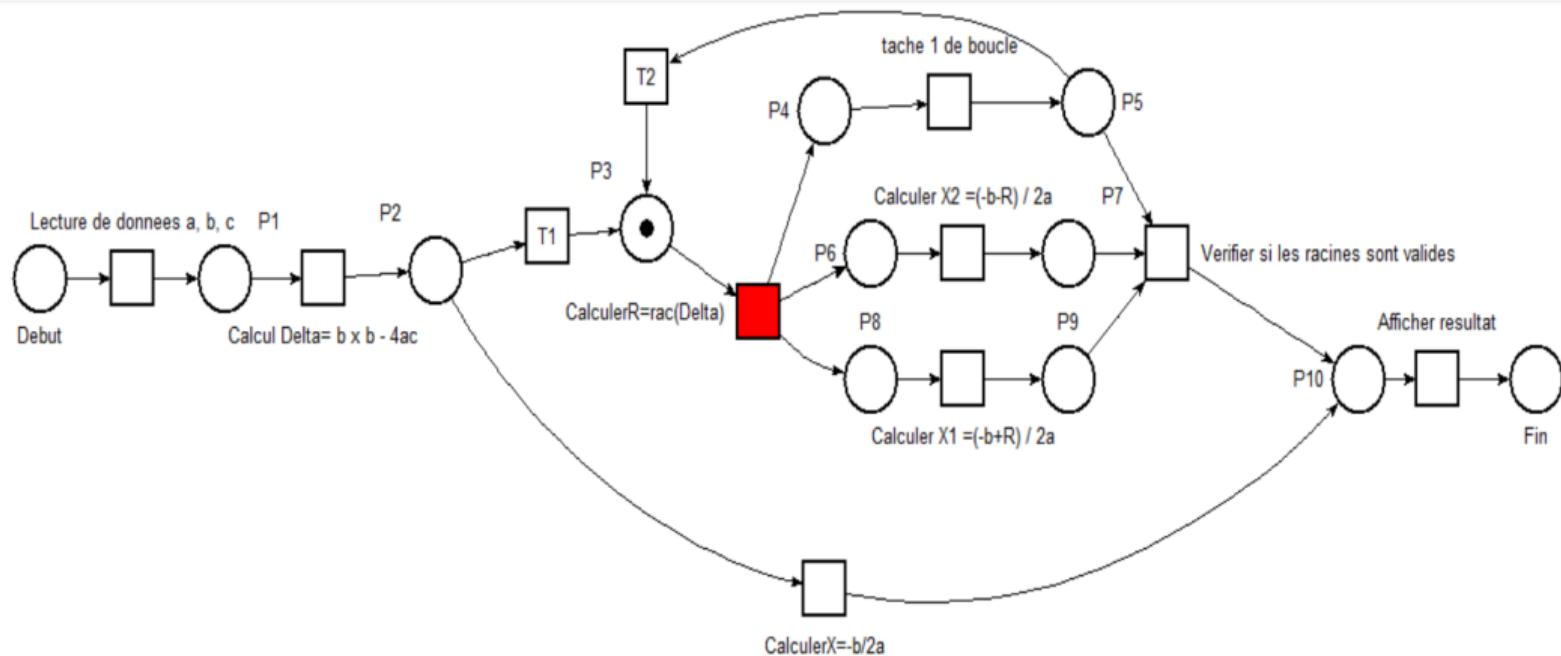


Figure 24 – Réseau de Pétri généré à partir du modèle BPMN non borné.

# Réseau de Pétri : Bornitude

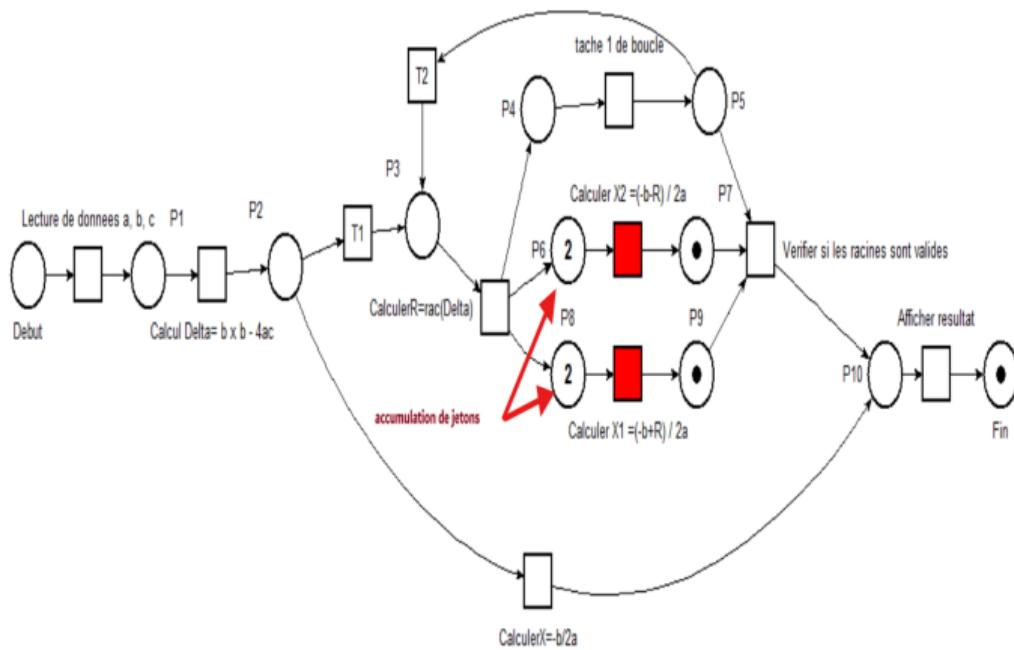


Figure 25 – Réseau de Pétri correspondant au modèle BPMN avec problème de Bornitude.

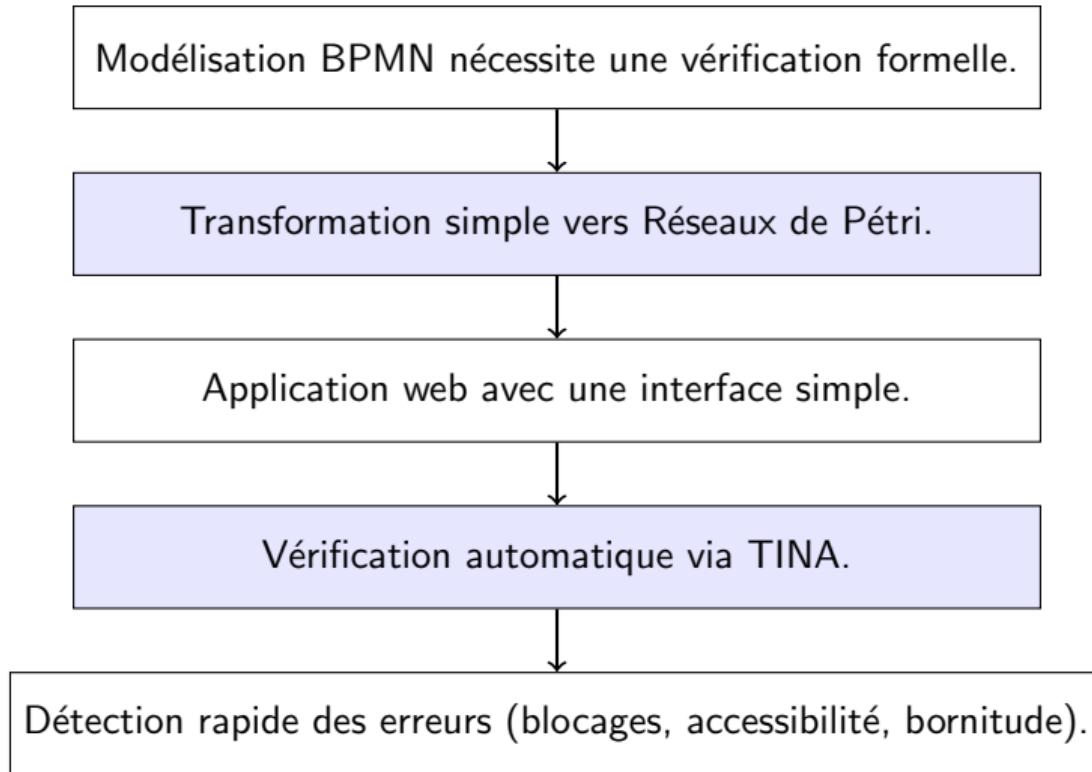
```

C:\Users\yahia\Desktop\Etude des cas\prm1 de l'Equation avec pro de boucle-tina...
Tina version 3.8.5 -- 01/16/25 -- IAAS/CNRS
mode -R
INPUT NET -----
parsed net (pml de 1 Equation avec pro de boucle)
12 places, 11 transitions, 26 arcs
net (pml de 1 Equation avec pro de boucle)
tr (Afficher resultat) P10 -> Fin
tr (Calcul Delta= b * b - 4ac) P1 -> P2
tr (Calculer X1 =(-b+R) / 2a) P8 -> P9
tr (Calculer X2 =(-b-R) / 2a) P6 -> P7
tr (CalculerR=rac(Delta)) P3 -> P4 P6 P8
tr (CalculerX=b/2a) P6 -> P10
tr (Lecture de donnees a, b, c) Debut -> P1
tr T1 P2 -> P3
tr T2 P5 -> P3
tr (Verifier si les racines sont valides) P5 P7 P9 -> P10

```

Figure 26 – Tableau d'analyse

# Conclusion



# Perspective

- Gérer les contraintes de temps et les aspects probabilistes.
- Permettre de traiter plusieurs fichiers BPMN en même temps.
- Prendre en charge tous les éléments de BPMN 2.0.
- Améliorer les rapports avec des diagnostics et des conseils.
- Développer une interface facile à utiliser pour voir et corriger les modèles.



Merci pour votre **attention !**

