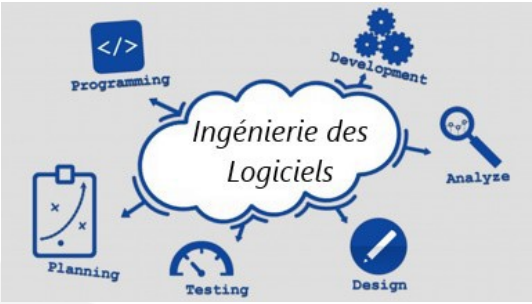
Ingénierie des Logiciels



Dr. Said MEGHZILI

Centre Universitaire de Mila

Département de Mathématique et Informatique

Email: s.meghzili@univ-mila.dz

1.0

Février 2022

Table des matières

I Chapitre 03 : Transformation de Graphes

1. Introduction

Dans ce chapitre, nous présentons les principes théoriques et pratiques de la transformation de graphes. Dans un premier temps, nous abordons les concepts de base suivants : la notion de graphe, le système de transformation de graphes, la grammaire de graphes et les outils de la transformation de graphes. Ensuite, nous présentons une transformation des modèles BPMN vers les réseaux de Petri en utilisant l'outil AToMPM. Cette transformation est basée sur la méta-modélisation et la transformation de graphes. Enfin, nous appliquons la transformation proposée sur un modèle BPMN.

2. Concepts de Transformations de Graphes

2.1. Concept de Graphe

Définition : Un graphe

Un graphe est constitué de sommets ou nœuds qui sont reliés par des arêtes. Plus formellement, on appelle graphe G = {S, A} tel que:

- S : ensemble fini non vide d'éléments appelés sommets ou nœuds.

- A : ensemble fini non vide de paires de sommets appelés arcs.

- A ⊆ X × X = {(s, t)|s, t ∈ S}, chaque arc de A relie deux sommets de S.

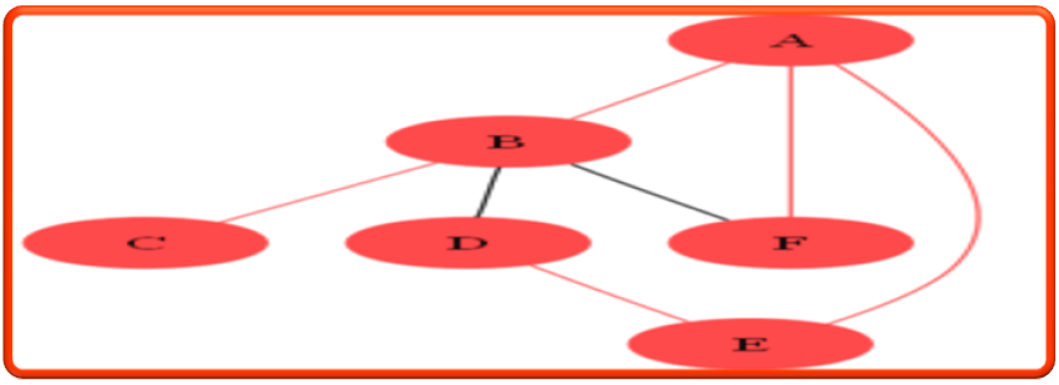


Figure 3.1 Graphe non orienté

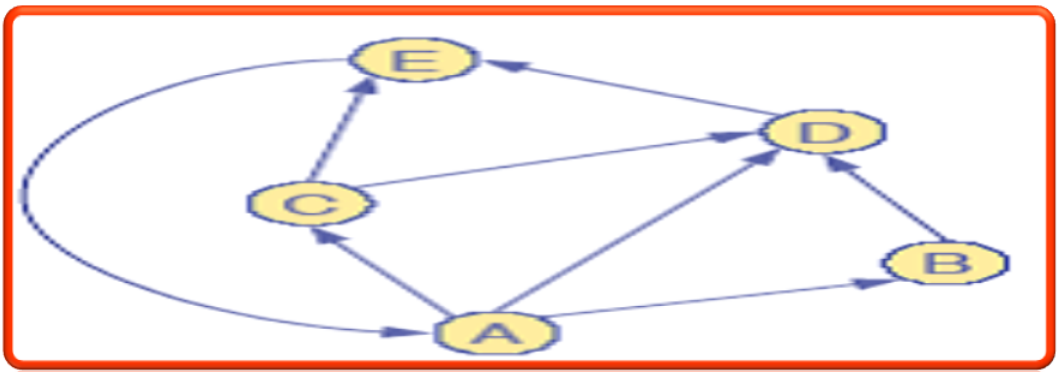


Figure 3.2 Graphe orienté

Définition : Graphe étiqueté

Un graphe étiqueté est un graphe orienté dans lequel les arcs possèdent un ensemble non vide d'étiquettes.

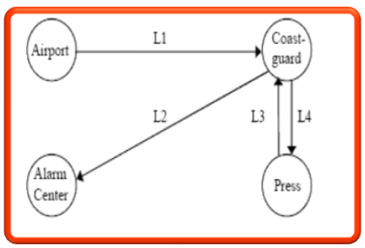


Figure 3.3: Graphe orienté étiqueté

Définition : Sous-graphe

sous graphe G'(S',A') d'un graphe G(S,A) est un graphe composé d'un sous ensemble de sommet S'∈ S et d'un sous ensemble d'arêtes A'∈A tel que A' représente les arêtes reliant les sommets S' dans le graphe G'.

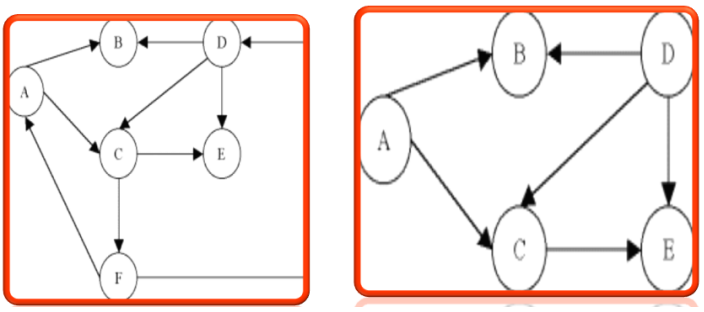


Figure 3.4: Graphe et sous-graphe

2.2. Grammaires de Graphes

Définition

Une grammaire de Graphe[9]* est généralement définie par un triplet: GG = (P, S, T)

Où : P est un ensemble de règles, S est un un graphe initial, T est un ensemble de symboles.

Nous utilisons des grammaires de graphes pour :

- Exprimer la sémantique opérationnelle (spécification du simulateur)
- Transformer des modèles en modèles comportementaux équivalents exprimés dans un autre formalisme.
- Optimiser les modèles.
- Générer du code pour un outil particulier (sémantique dénotationnelle).

Complément : Principe d'une règle

Une règle de transformation de graphe[9]* est définie par : r = (L, R, K, glue, emb, cond). Elle consiste en:

- Deux graphes ***L*** graphe de côté gauche et ***R*** graphe de côté droit.
- Un sous graphe ***K*** de ***L***.
- Une occurrence ***glue*** de ***K*** dans ***R*** qui relie le sous graphe avec le graphe de côté droit.
- Une relation d'enfoncement ***emb*** qui relie les sommets du graphe de côté gauche et ceux du graphe du côté droit.
- Un ensemble ***cond*** qui spécifie les conditions d'application de la règle.

Méthode : Application des règles

L'application d'une règle r= (L, R, K, glue, emb, cond) à un graphe G produit un graphe résultant H. Le graphe H fourni, peut être obtenu depuis le graphe d'origine G en passant par les cinq étapes suivantes :

1. Choisir une occurrence du graphe de côté gauche L dans G.
2. Vérifier les conditions d'application d'après cond.
3. Retirer l'occurrence de L (jusqu'à K) de G ainsi que les arcs pendillé, c.-à-d. tous les arcs qui ont perdu leurs sources et/ou leurs destinations. Ce qui fournit le graphe de contexte D de L qui a laissé une occurrence de K.

4. Coller le graphe de contexte D et le graphe de côté droit R suivant l'occurrence de K dans D et dans R. c'est la construction de l'union de disjonction de D et R et, pour chaque point dans K, identifier le point correspondant dans D avec le point correspondant dans R.
5. E. Enfoncer le graphe du côté droit dans le graphe de contexte de L suivant la relation d'enfoncement emb . L'application de r sur un graphe G pour fournir un graphe H est appelée une dérivation directe depuis G vers H à travers r, elle est dénotée par G ⇒ H ou simplement par G ⇒ H.

2.3. Système de transformation de graphes

La transformation de graphe est spécifiée sous forme d'un modèle de grammaires de graphes. Ces dernières sont une généralisation des grammaires de Chomsky pour les graphes. Elles sont composées de règles. Une règle est constituée de deux parties, le Left Hand Side (LHS) et le Right Hand Side (RHS). Le LHS est la partie gauche de la règle, destinée à être mise en concordance avec les parties du graphe (appelé host graph) où on veut appliquer la règle. La partie droite de la règle, Le RHS, décrit la modification qui sera effectuée sur le host graph, elle substitue dans le host graph la partie identifiée par la partie gauche de la règle [18]*.

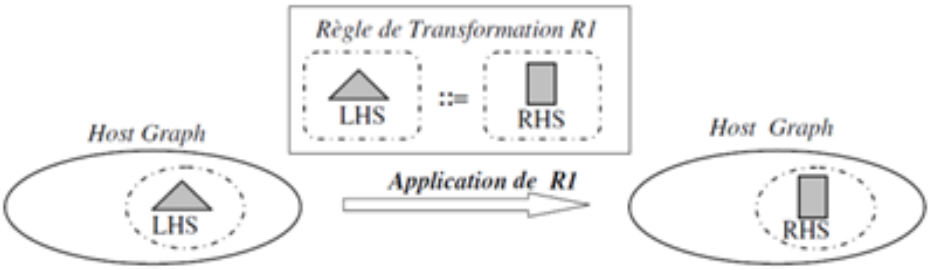


Figure 3.5 Principe de l'application d'une règle de transformation

Un système de transformation de graphe est défini comme un système de réécriture de graphes qui applique les règles de la Grammaire de Graphes sur son graphe initial jusqu'à ce que plus aucune règle ne soit applicable [18]*.

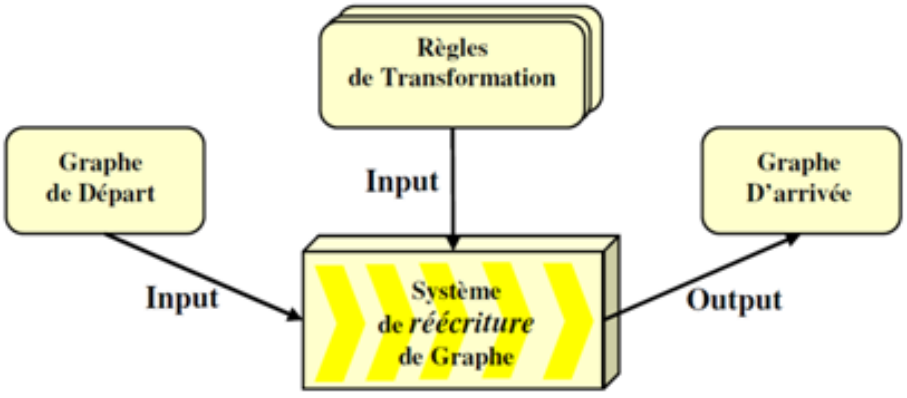


Figure 3.6: Système de réécriture de graphes

Fondamental : Avantage

Approche formelle fondée sur des ***bases mathématiques***:

- La théorie des graphes
- Les grammaires formelles
- La réécriture de termes

2.4. Outils de transformation de graphes

Plusieurs outils de transformation de graphes existent actuellement, parmi lesquels :

- AGG [AGG]:The Attributed Graph Grammar System.
- AToM3: A Tool for Multi-formalism and Meta-Modelling.
- AToMPM : A Tool for Multi-Paradigm Modeling.
- VIATRA :VIsual Automated model TRAnsformations.
- FUJABA: From UML to Java and back again.
- GreAT : The Graph Rewrite And Transformation tool suite.

L'outil AToMPM

- AToMPM[8]* signifie « A Tool for Multi-Paradigm Modeling ».
- Outil pour la modélisation multi-paradigme.
- Successeur de AToM3.
- Il basé complètement sur le Web.
- Il fonctionne sur le nuage.
- Effectuer des transformations de modèles, et manipuler et gérer des modèles.
- AToMPM possède :

    1. Une couche de Méta-modélisation qui permet La modélisation graphique d'un formalisme ainsi que la génération automatique d'un outil pour manipuler les différents modèles décrits dans le formalisme spécifié.
    2. Un système de réécriture de graphes qui permet Les manipulations de modèles par application itérative des règles d'une Grammaire de Graphes.

3. Transformation des modèles BPMN vers les Réseaux de Petri

3.1. Les formalismes source BPMN et cible Rdp

Définition : Notation BPMN

BPMN "Business Process Model and Notation" est un langage pour la modélisation des processus métier.

L'objectif principal du langage BPMN est de fournir une notation standard qui soit simple, facile et compréhensible par tous les utilisateurs métiers.

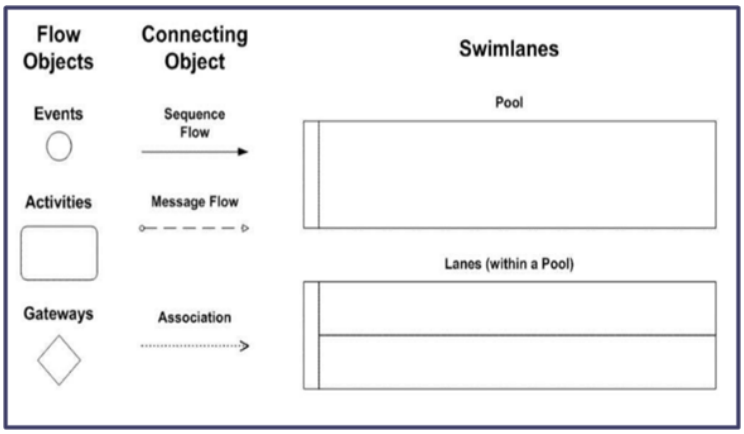


Figure 3.7: Les principaux éléments de la notation BPMN

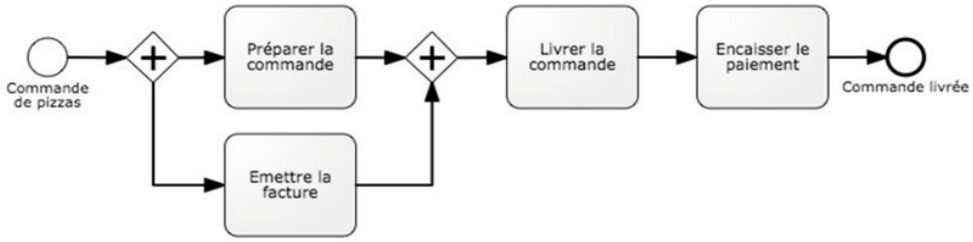


Figure 3.8 : Exemple d'un modèle de la pizzeria avec BPMN

Définition : Réseaux de Petri

Un réseau de Pétri (aussi connu comme un réseau de Place/Transition) est un modèle graphique et mathématique permettant de modéliser et de vérifier le comportement dynamique des systèmes [13]*.

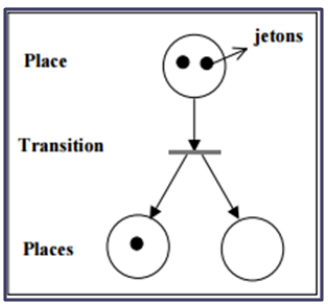


Figure 3.9 : Exemple d'un modèle RdP

Motivations de la transformation des modèles BPMN vers RdP

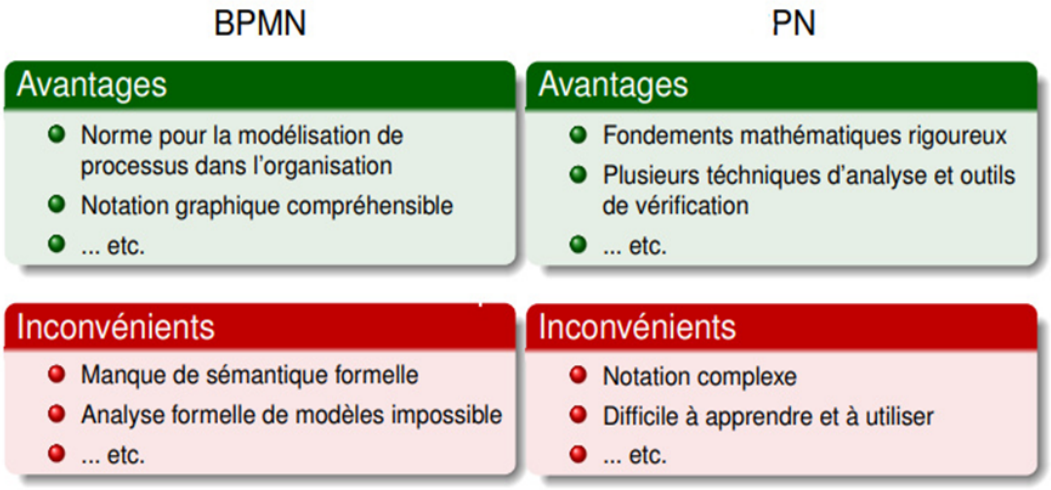


Figure 3.10 : Avantages et inconvénients des modèles source et cible

- Chacune des deux modèles (BPMN et Rdp) a des points forts et des points faibles.
- Les deux approches peuvent être combinées dans le sens où les inconvénients de l'un peuvent être surmontés grâce aux apports de l'autre.

Méthode

Afin de transformer les modèles BPMN vers les réseaux de Pétri, on suit les trois étapes suivantes:

1. Construction d'un méta-modèle pour le langage BPMN.
2. Construction d'un méta-modèle pour les réseaux de Pétri.
3. Définition des règles de la transformation (Grammaire de Graphes).

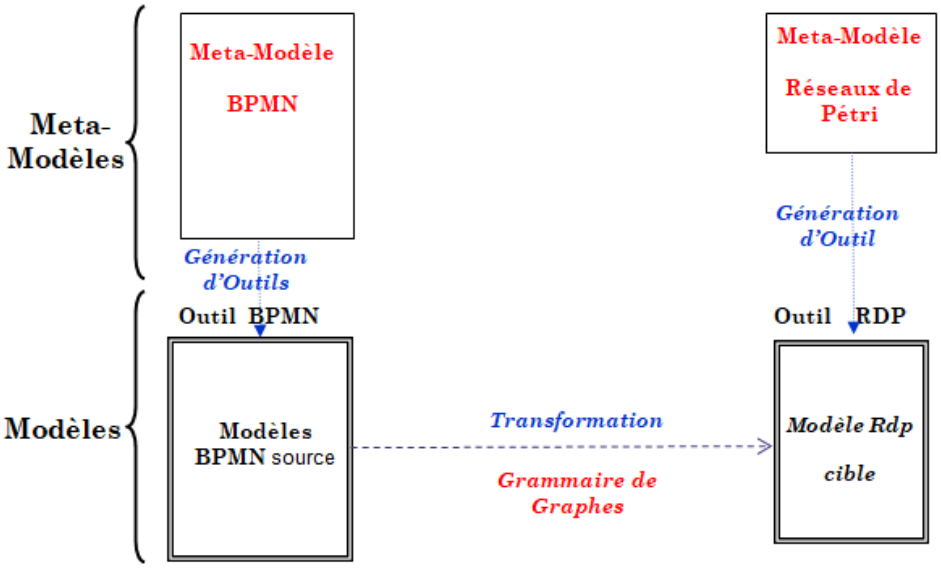


Figure 3.11 : Architecture de la transformation

3.2. Méta-Modélisation des BPMN et les RdP

Méta-Modélisation des BPMN

La figure 3.12 montre le méta-modèle (la syntaxe abstraite) des modèles BPMN. Ce méta-modèle regroupe les éléments de bases qui comprennent une activité (Activity), deux type de branchement (XOR et AND), et les flux de séquence (AND2Act, Act2XOR, XOR2Act, sequencearc).

La figure 3.13 montre la syntaxe concrète de BPMN.

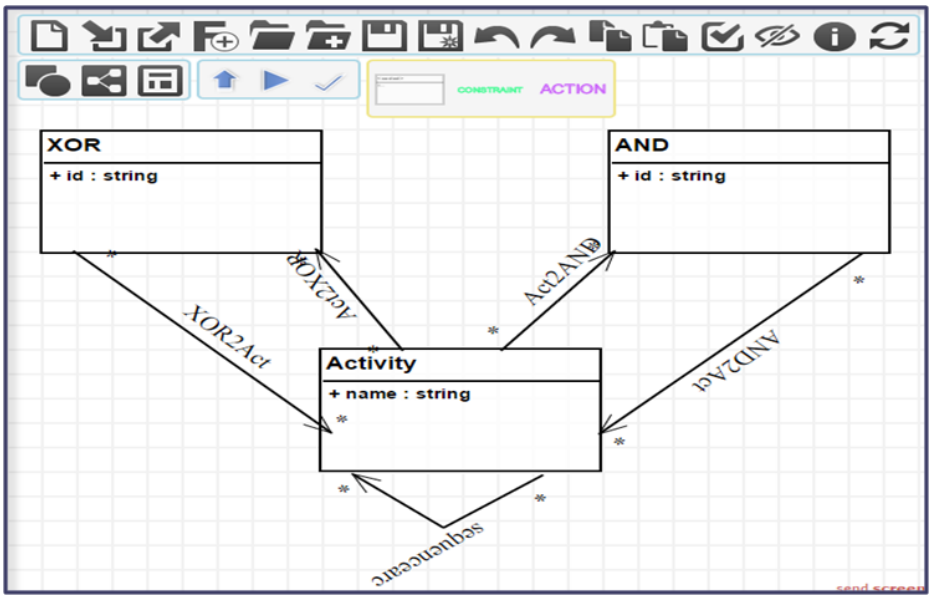


Figure 3.12 : Méta-Modèle de la notation BPMN

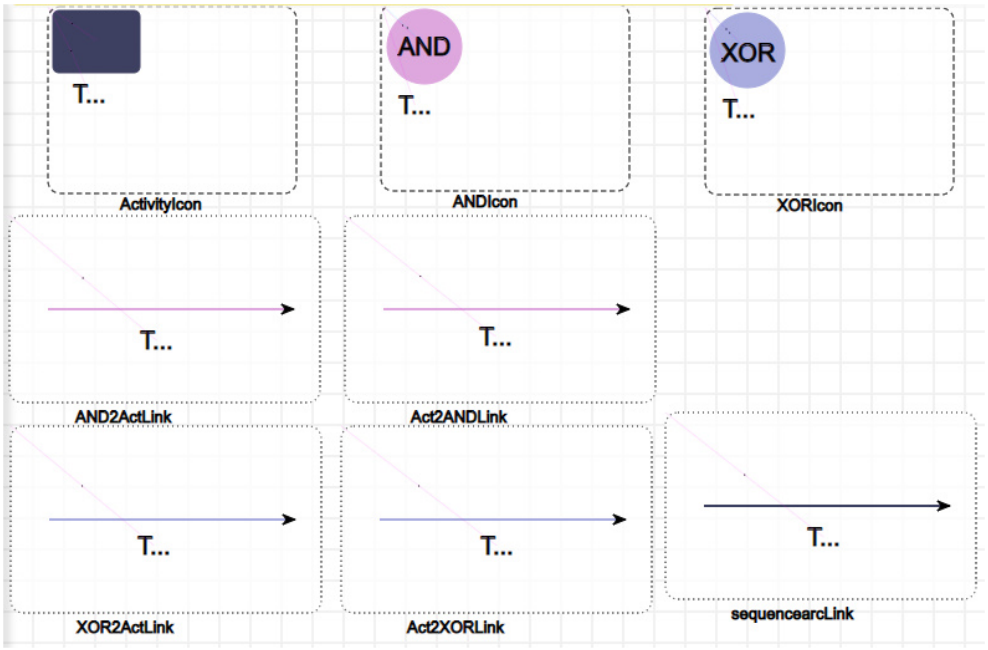


Figure 3.13 : Syntaxe concrète de la notation BPMN

Méta-Modélisation des Réseaux de Petri

La figure 3.14 montre le méta-modèle (la syntaxe abstraite) des Réseaux de Petri. Ce méta-modèle regroupe les éléments de bases qui comprennent une transition (Transition), et une place (Place), et deux types des arcs (T2P: outarc et P2T: inarc).

La figure 3.15 montre la syntaxe concrète des Réseaux de Petri.

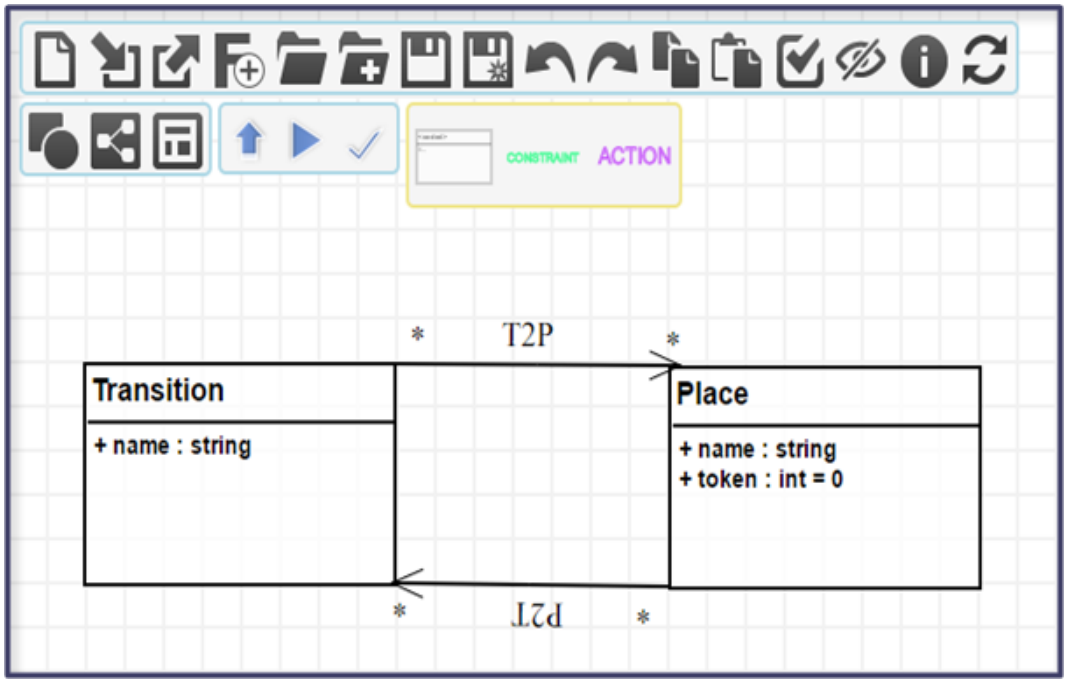


Figure 3.14 : Méta-Modèle des Réseaux de Petri

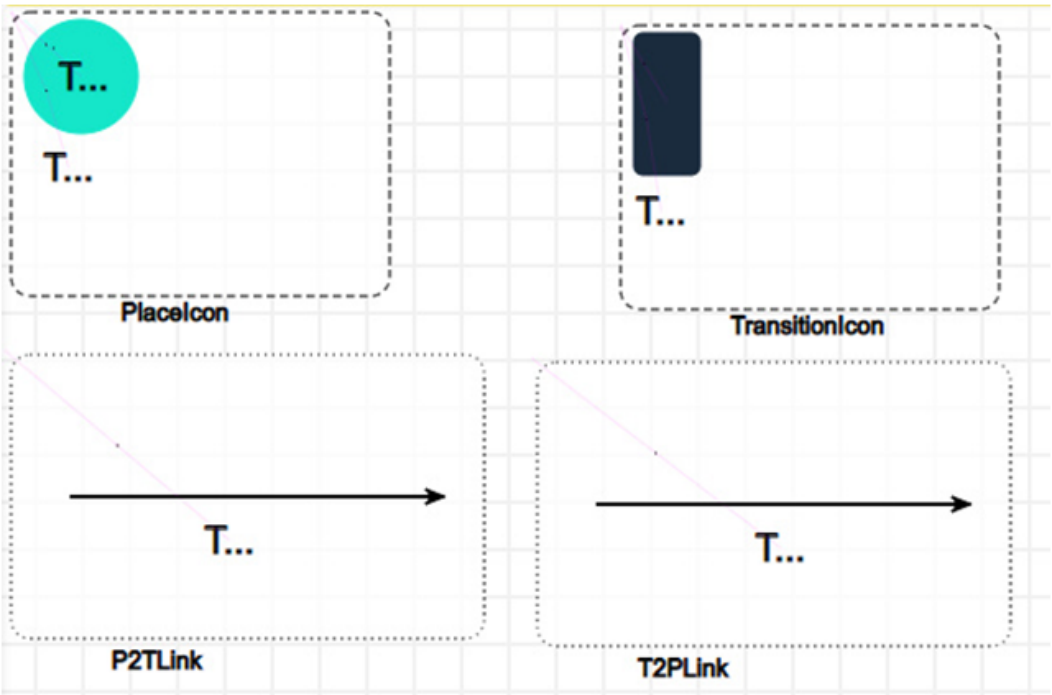


Figure 3.15 : Syntaxe concrète des Réseaux de Petri

3.3. Règles de la transformation

Définition : Idée de la transformation BPMN vers RDP

L'idée de la transformation des modèles BPMN vers les RdP est la suivante:

1. Les activités sont transformés en des transitions.
2. Les connecteurs XOR sont transformés en des places.
3. Les connecteurs AND sont transformés en des places.

Méthode : Grammaire de Graphes

Afin de produire les modèles BPMN équivalents au RdP, nous avons proposé les 9 règles suivantes [11]* :

- Règle 1 : R_Activity2Transition
- Règle 2 : R_sequencearc
- Règle 3 : R_xor2place
- Règle 4 : R_inXOR2inPL
- Règle 5 : R_outXOR2outPL
- Règle 6 : R_ANDrule
- Règle 7 : R_DeleteXOR
- Règle 8 : R_DeleteAND
- Règle 9 : R_DeleteActivity

Règle 1 R_Activity2Transition

Cette règle génère une nouvelle transition (dans RdP) pour chaque activité (dans BPMN).

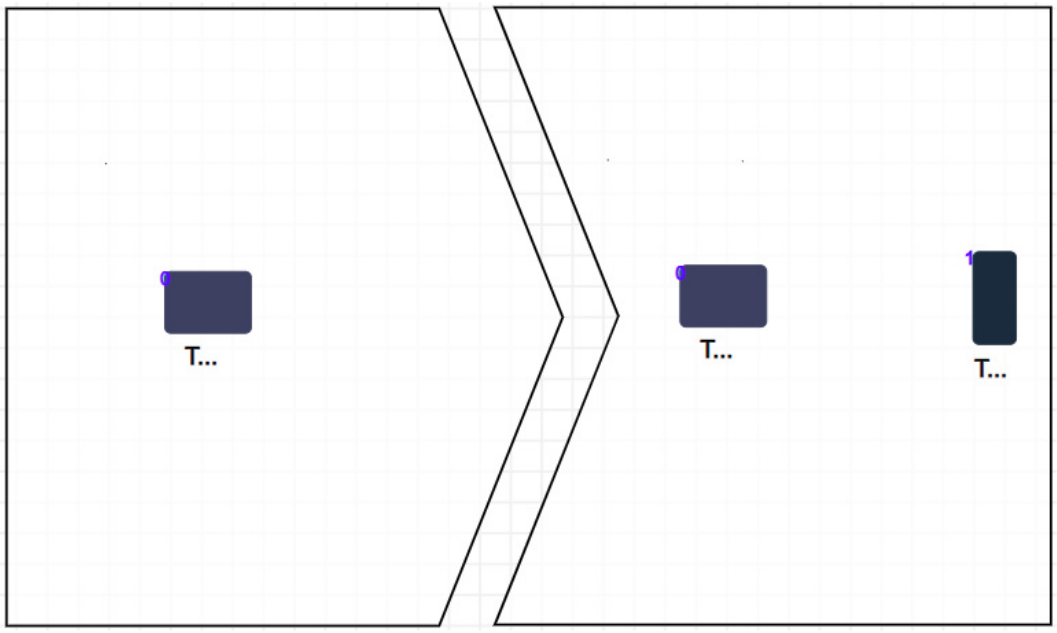


Figure 3.16 : Règle R_Activity2Transition

Règle 2 R_sequencearc

Cette règle est appliquée à chaque arc de séquence:

- Elle génère une nouvelle place avec un arc entrant et un arc sortant aux transitions.
- Elle supprime tous les arcs de séquence.

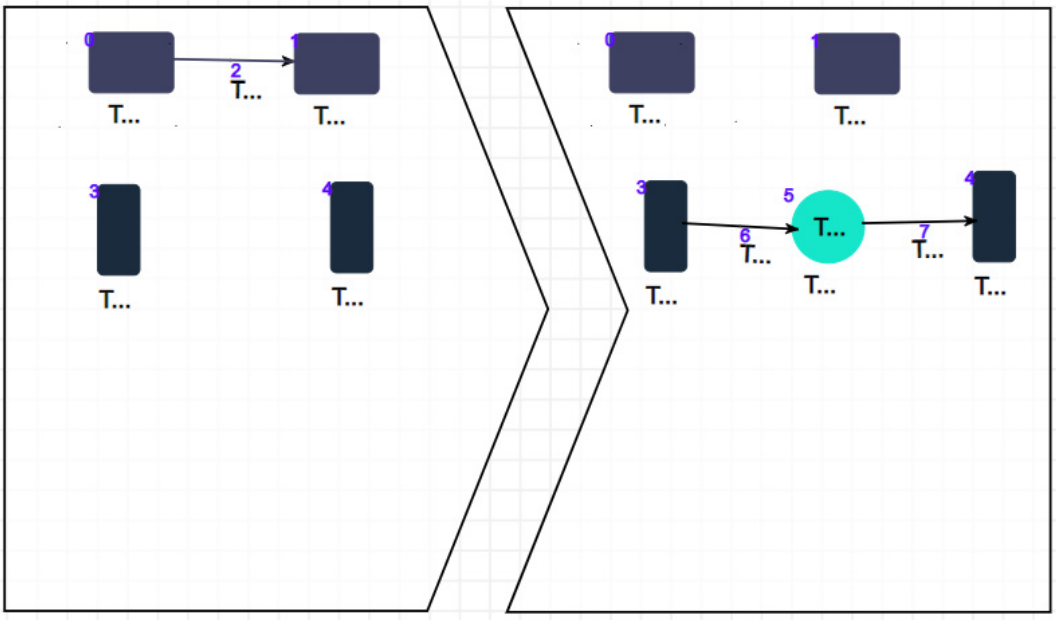


Figure 3.17 : Règle R_sequencearc

Règle 3 R_xor2place

Cette règle génère une nouvelle place pour chaque connecteur XOR.

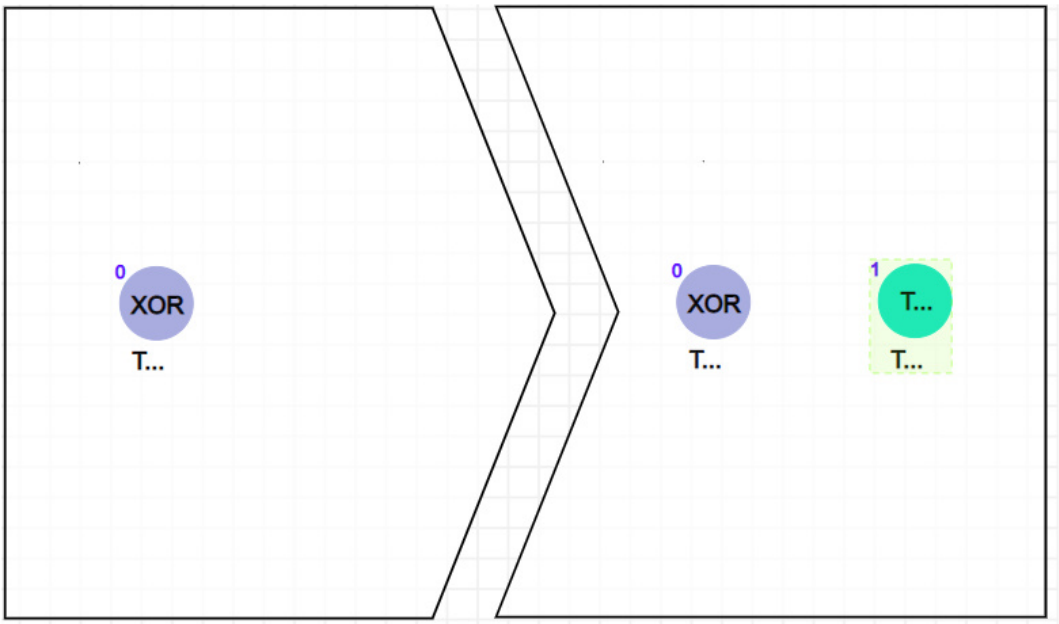


Figure 3.18 : Règle R_xor2place

Règle 4 R_inXOR2inPL

Cette règle génère un arc sortant de la transition (attachée à l'activité) vers la place et elle supprime tous les arcs (Act2XOR).

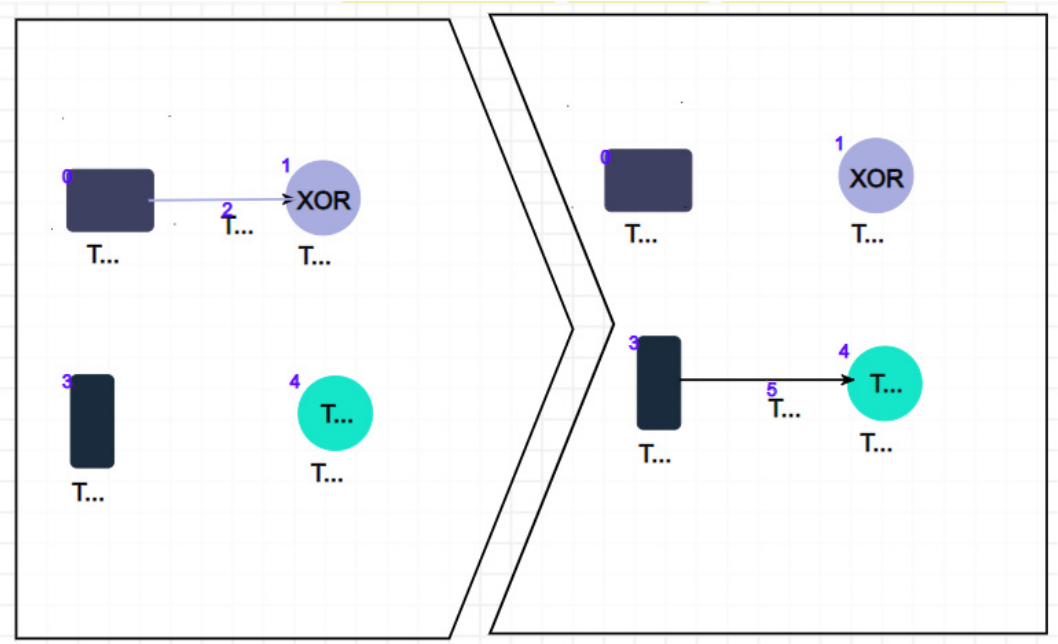


Figure 3.19 : Règle R_inXOR2inPL

Règle 5 R_outXOR2outPL

Cette règle génère un arc sortant de la place (attachée au lien XOR) vers la transition (attachée à l'activité) et elle supprime tous les arcs (XOR2Act).

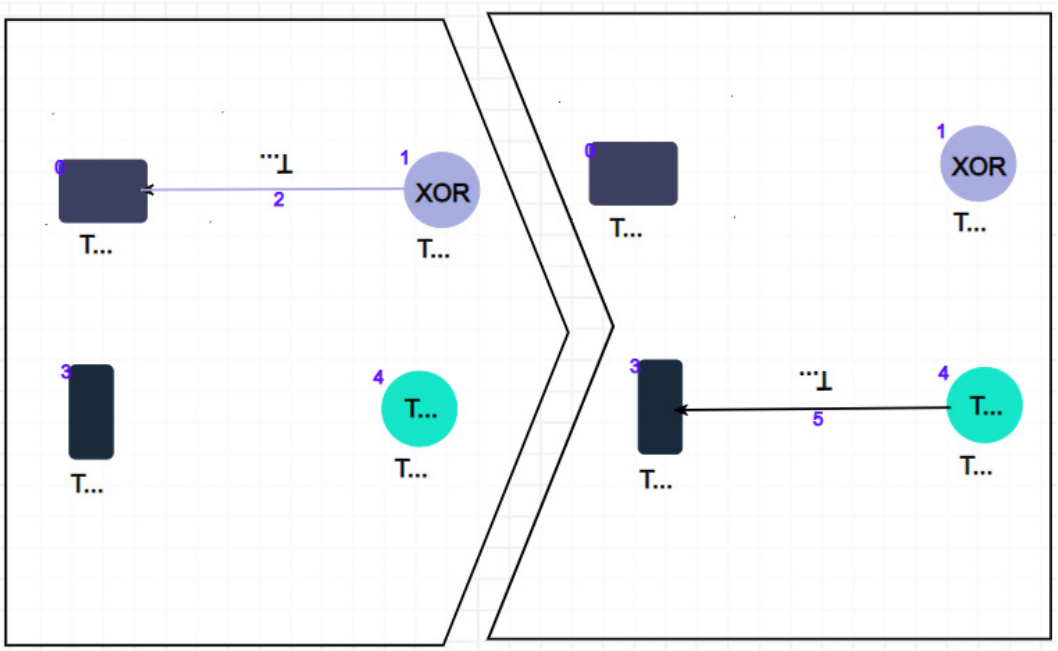


Figure 3.20 : Règle R_outXOR2outPL

Règle 6 R_ANDrule

Cette règle génère une nouvelle place avec un arc entrant et un arc sortant aux transitions (Liés aux activités).

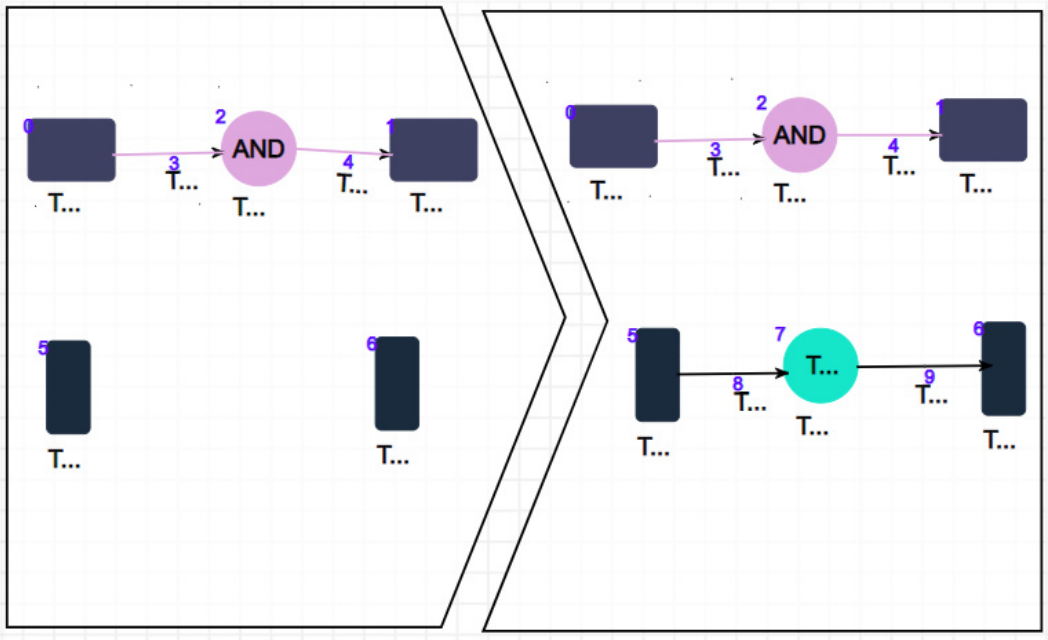


Figure 3.21 : Règle R_ANDrule

Règle 7 R_DeleteXOR

Cette règle supprime tous les connecteurs XOR du modèle BPMN.

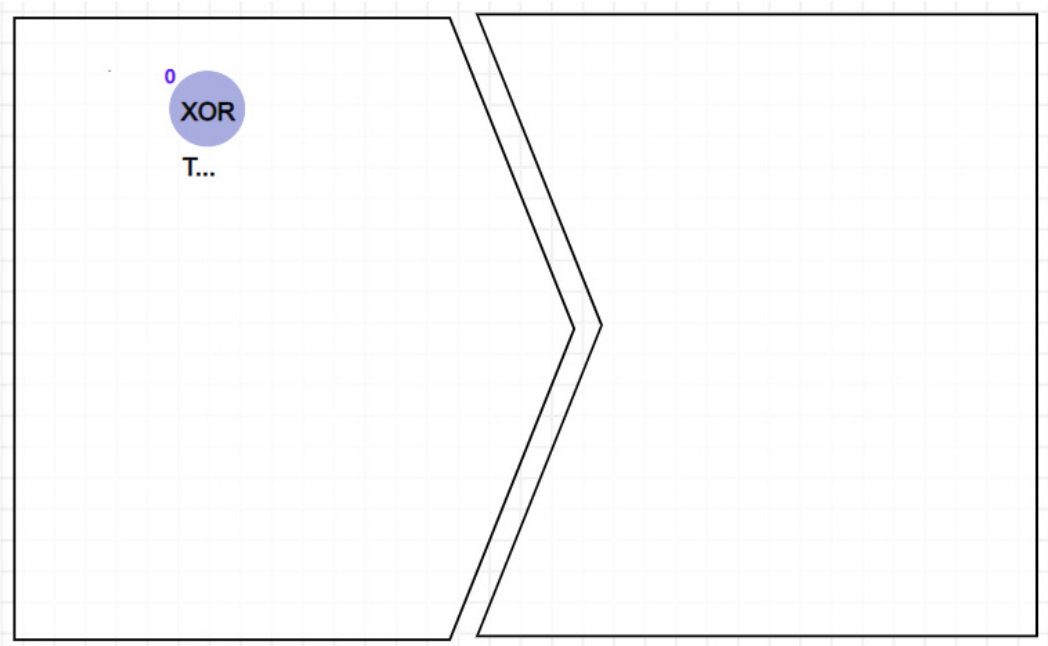


Figure 3.22 : Règle R_DeleteXOR

Règle 8 R_DeleteAND

Cette règle supprime tous les connecteurs AND du modèle BPMN.

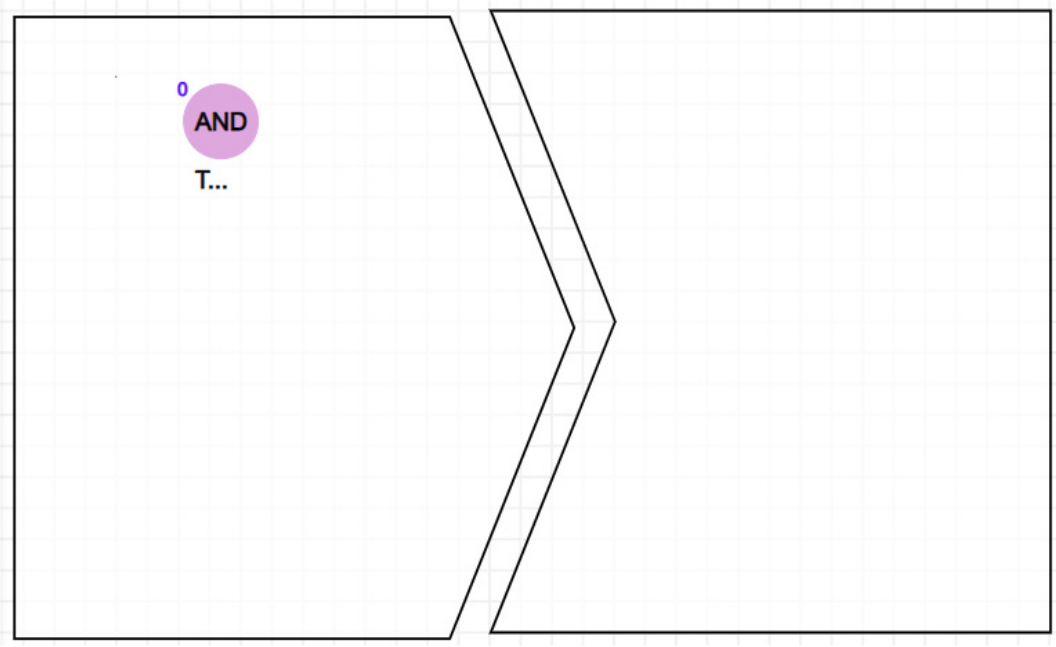


Figure 3.23 : Règle R_DeleteAND

Règle 9 R_DeleteActivity:

Cette règle supprime tous les activités du modèle BPMN.

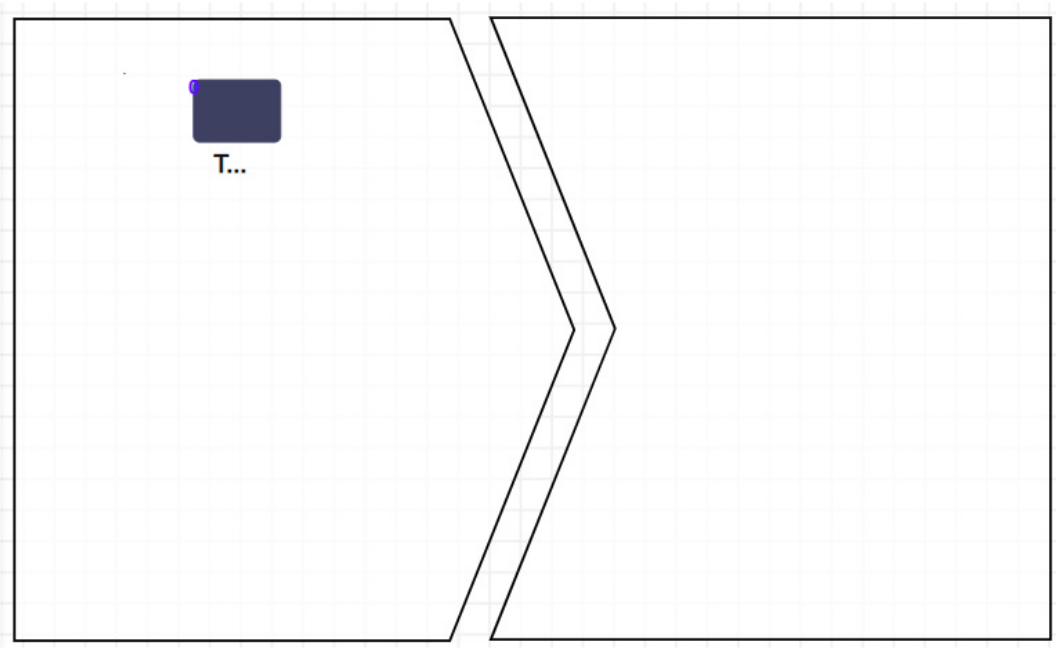


Figure 3.24 : Règle R_DeleteActivity

Processus de la transformation

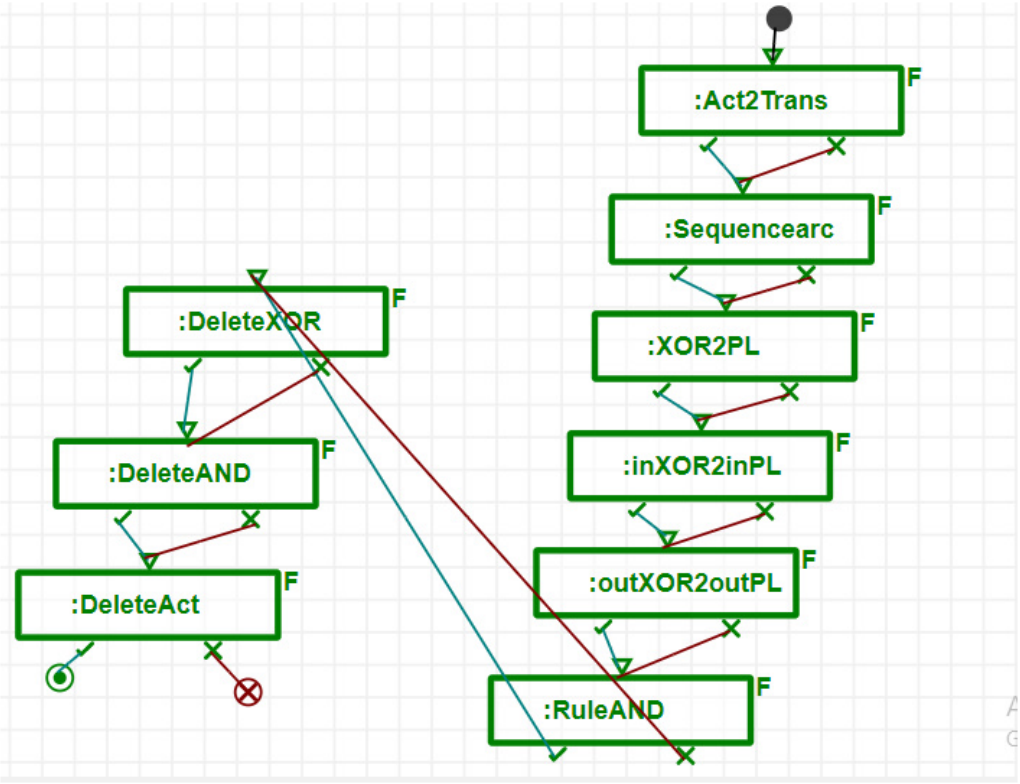


Figure 3.25 : Processus d'application des règles

La figure 3.25 montre le processus d'application des règles dans l'outil AToMPM. Ce processus est construit en utilisant le langage MoTiF.

4. Exemple de transformation d'un modèle BPMN

Exemple

Nous essayerons à travers cet exemple de montrer en détails l'application de la transformation proposée.

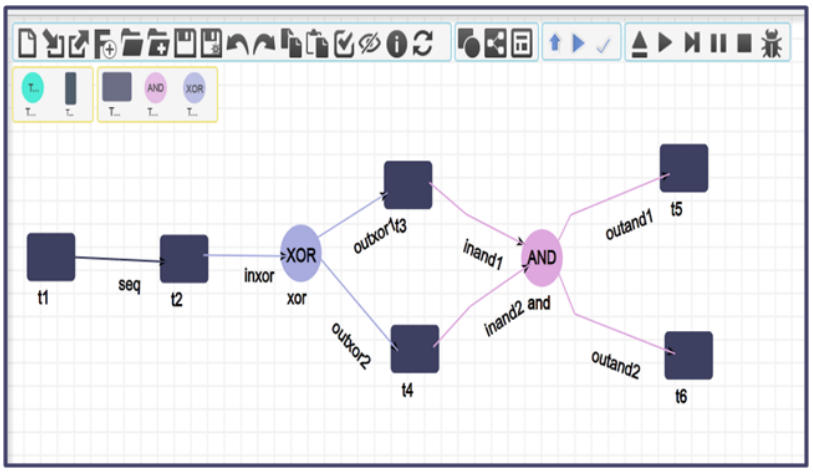


Figure 3.26 : Exemple d'un modèle BPMN source

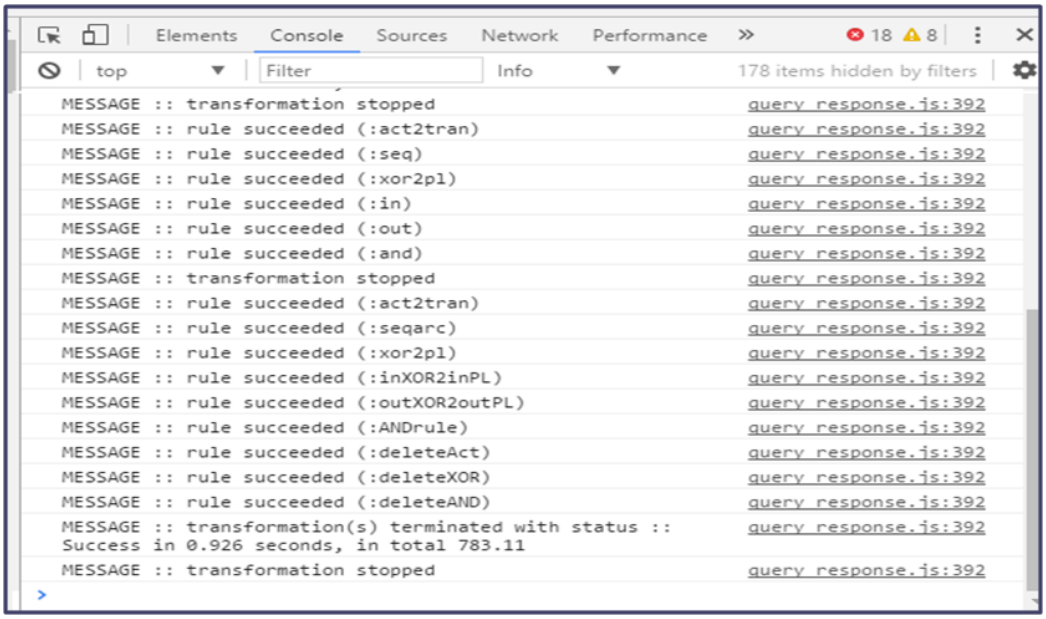


Figure 3.27 : Exécution de la transformation dans l'outil AToMPM

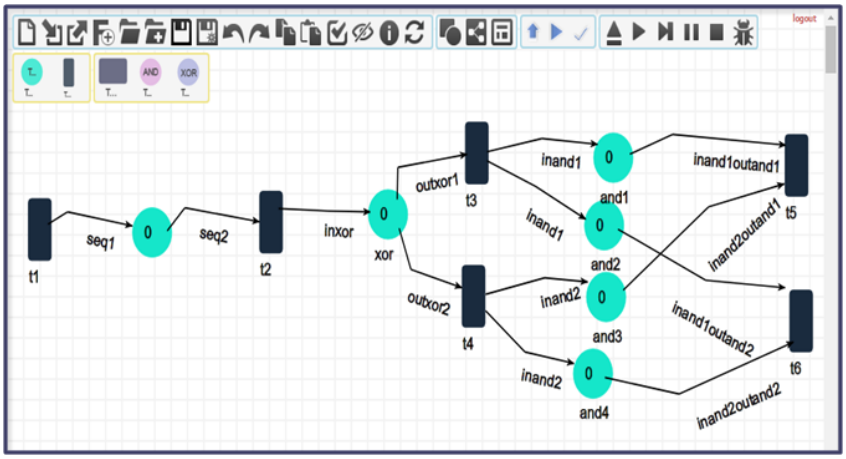


Figure 3.28 : Le réseau de Petri généré par la transformation

5. Conclusion

Dans ce chapitre, nous avons présenté les principes de transformations de graphes. Premièrement, nous avons introduit les concepts suivants : la notion de graphe, le système de transformation de graphes, les grammaires de graphes et les outils de transformation de graphes. Deuxièmement, nous avons présenté une transformation des modèles BPMN en réseaux de Petri à l'aide de l'outil AToMPM. Cette transformation a été développée dans les étapes suivantes :

1. Définir un méta-modèles pour les modèles BPMN,.
2. Définir un méta-modèles pour les réseaux de Pétri.
3. Définir une grammaire de graphes de la transformation des modèles BPMN en leur équivalent dans les réseaux de Petri.

Finalement, nous avons appliqué la transformation proposée sur un modèle BPMN.

Références

11
Makhloufi & Benkara, Mémoire de Master RSD en Informatique, Université de Constantine 2, 2017.

13
Jensen, Kurt, and Lars M. Kristensen. Coloured Petri nets: modelling and validation of concurrent systems. Springer Science & Business Media, 2009.

18
Grzegorz Rozenberg, "Handbook of Graph Grammars and Computing by Graph Transformation", Vol.1. World Scientific, 1999.

8 AToMPM home page : https://atompm.github.io/.

9
Marc Andries, Gregor Engels, Annegret Habel, Berthold Hoffmann, Hans-Jorg Kreowski, Sabine Kuske, Detlef Pump, Andy Schürr and Gabriele Taentzer, "Graph transformation for specification and programming", Science of Computer programming, vol 34, NO°1, pages 1-54, Avril 1999.