Projet de Bachelor

TPG Control Center: Tramway modeling and Simulation

par
Marius NGUENA KENGNI
(*nguenak3@etu.unige.ch*)

Superviseur:
Professeur Didier BUCHS
(*Didier.Buchs@cui.unige.ch*)

Co-Superviseur:
M. Ang CHEN
(*Ang.Chen@cui.unige.ch*)

Genève, le 15 Août 2007

Résumé

Dans ce mémoire, nous présentons une petite maquette de réseau de transport urbain inspiré du cas réel de la ville de Genève (SUISSE). Il s'agit principalement du réseau de ***tramways***, exploité par l'entreprise ***TPG*** (**T**ransports **P**ublics **G**enevois). Nous prenons en compte les cinq lignes de tram qui existent déjà sur le réseau physique actuel et le flux de véhicules (trams) sur ces lignes afin de mesurer l'impact des croisements et de la fréquence des déplacements sur ce flux de véhicules. Nous montrons précisement que la fréquence d'insertion des trams dans le réseau détermine la capacité dynamique du réseau et permet d'identifier la saturation du réseau. Pour mesurer les performances de réseau, nous les simulons et nous obtenons des résultats que nous parsons dans le langage Java : Le flux maximum de circulation et la saturation sont deux mesures de performances du réseau que nous analysons et visualisons graphiquement par l'intermédiaire de l'API Jcharts. Nous proposons ensuite un système d'aide à la régulation en situation de conflits sur le réseau, ce référentiel donne une estimation de façon fiable, des risques encourus en cas de dysfonctionnements ou de pannes. Le système complet est modélisé par les Réseaux de Petri de haut niveau : ***Colorés***$(RdpC)$, ***Temporisés***$(RdpT)$ et ***Stochastiques***$(RdpS)$.

Mots clé : Modélisation, Segments, Aiguillage, Décision, Régulation, Croisements, Temporisation, Transitions immédiates, Transitions déterministes, Transitions exponentielles, Aide à la décision, Capacité, Concurrence, Flux, Fréquence, Parseur.

Table des matières

Remerciements **3**

Introduction **4**

1 Les Transports Publics Genevois(TPG) **6**
1.1 Historique 6
1.2 Statistiques 6

2 Les principaux outils de modélisation **8**
2.1 Les Réseaux de Pétri 8
2.1.1 Les composants élémentaires. 8
2.1.2 Places, Transitions, Arcs et jetons 9
2.1.3 Exemple 1 : Les nombres Pairs et Impairs 9
2.1.4 Exemple 2 : Un carrefour de deux routes 11
2.1.5 La modélisation par les réseaux de Petri 12
2.2 Les Extensions des réseaux de Petri 13
2.2.1 Réseaux de Pétri Colorés(RdpC) 13
2.2.2 Réseaux de Pétri Temporisés(RdpT) 14
2.2.3 Réseaux de Petri Stochastiques (RdpS) 17
2.3 Le language UML (Unified Modeling Language) 19
2.3.1 Des modèles plutôt que du Code 19

3 La Modélisation du Réseau de Transport **21**
3.1 Description d'un réseau physique 21
3.2 Les règles syntaxiques et sémantiques de transformation : Réseau physique ↪ Réseau de Petri 22
3.2.1 Les Croisements 23
3.2.2 Représentation conjointe "Réseau de Petri", "Tram", "Ligne de Trams" et "Segment physique" 27
3.3 Transformation d'un modèle physique en réseau de Petri : la maquette PLAINPALAIS 28

4 Transformations, Simulations et Analyses **30**
4.1 Le principal outil de simulation : Le logiciel HPsim 30
4.2 Modèle d'analyse 32
4.3 Simulation 1 : Transformation d'une maquette en réseau de Pétri coloré 32
4.3.1 Exemple 1 : scénario MOILLESULAZ 32
4.3.2 Exemple 2 : Pliage et Dépliage (Comportements identiques) 34
4.4 Simulation 2 : Transformation d'une maquette en réseau de Pétri T-temporisé 35
4.4.1 Régulation du réseau : Détection d'un état "anormal" . 36
4.4.2 Gestion des tables horaires 39
4.4.3 Modèle Standard et Histogramme 41
4.4.4 Modèle Temps-réel et Histogramme 42
4.5 Simulation 3 : Transformation d'une maquette en réseau de Petri stochastique 43
4.5.1 Gestion du flux de trams 44
4.5.2 Les mesures de performance 46
4.5.3 Simulation d'un croisement simple : [C/2,2] 47
4.5.4 Simulation de la maquette de Plainpalais 54

5 Programme JAVA : Analyseur-de-performances **60**
5.1 Algorithme du programme 61
5.1.1 listing code création panel 1 : Flux maximum et fréquence d'insertion 65

6 Modélisation UML **69**
6.1 Le diagramme de classes 70
6.2 Les Reseaux de Petri *vs* le language UML 71

7 Conclusion et Perspectives **73**

Remerciements

Je tiens en premier lieu à remercier le bon DIEU de nous avoir offert tout ce que nous possédons.

Je remercie également tous les membres du groupe SMV (Software Modeling and Verification) du CUI (Centre Universitaire Informatique) de l'université de Genève, en particulier le Professeur Didier BUCHS et Ang CHEN, pour la disponibilité dont ils ont fait preuve tout au long de ce travail, et ceci malgré leurs nombreuses occupations professionnelles respectives.

Et c'est avec un réel plaisir que je dédie ce travail :

A ma mère et à mon père, dont la rigeur dans le travail et les intarissables conseils me servent de modèle dans la vie de tous les jours,

A mes soeurs et mon frère (Ingrid, Josee, Elodie, Clovis).

A ma très grande famille,

et à tous ceux qui me soutiennent au quotidien.

Introduction

La modélisation et la simulation d'un réseau de transports publics urbains est une tâche délicate et compliquée. En effet, les systèmes distribués tels que les réseaux de trams sont caractérisés par des comportements parallèles, asynchrones et indéterministes. Le formalisme des modèles de réseaux de Petri permet l'expression de ces comportements.

Les évènements perturbateurs et leurs conséquences peuvent nuire de façon significative à l'image d'une entreprise, à ses produits, ses employés, son organisation et à sa réputation. Les crises sont la plupart du temps imprévisibles. On peut néanmoins recenser et organiser des informations qui vont servir de référentiel pour la prévention de ces évènements incertains. Comment vérifier que le système fonctionne correctement ? Nous étudions le comportement du système (*modélisation qualitative* [BOU]), ensuite nous proposons un modèle standard qui regroupe les comportements traditionnels de notre système. Ce modèle opérateur sera utilisé pour détecter les états anormaux sur des modèles "temps-réel".

Les origines des perturbations dans les réseaux de transports urbains sont multiples et variées. On peut citer une origine extérieure comme des manisfestations publiques, des travaux de la route, ou une origine interne comme des pannes de matériel ou des retards. Les **T**ransports **P**ublics **G**enevois (***TPG***) essayent en temps réel de rétablir les conditions normales suitent à ces perturbations. Par exemple lorsqu'il y a une manifestation, les trams "adaptent" leur vitesse de déplacement et dévient parfois sur d'autres lignes. Notre objectif est de modéliser ces lignes de trams et de mesurer l'impact de ces décisions de régulations sur les flux de véhicules. Faire des mesures implique l'évaluation des performances (*modélisation quantitative* [BOU]). La notion de temps fait alors son apparition. Nous présentons les relations qui existent entre la capacité du réseau, la fréquence et la vitesse de déplacement dans le réseau.

Ce mémoire est constitué de six chapitres :

- Dans le premier chapitre, nous donnons un bref historique sur les ***TPG***.

- Le second chapitre concerne les principaux outils impliqués dans la modélisation. Nous présentons les modèles de réseaux de Petri, en particulier des caractéristiques de leurs extensions (colorés, temporisés, stochastiques). Nous introduisons également le langage de modélisation orienté objet UML (Unified Modeling Language).

- Le troisième chapitre est consacré à la transformation d'un modèle physique de trams en un modèle de réseaux de Petri. Nous donnons principalement les règles syntaxiques et sémantiques qui permettent d'exprimer tout réseau physique en réseau de Petri.

- Dans le quatrième chapitre nous introduisons brièvement le principal outil de simulation qui nous a permis de réaliser ce travail à savoir le logiciel HPsim, ensuite nous faisons quelques simulations des extensions de réseaux de Petri issues des transformations de maquettes de notre réseau de trams.

- Le cinquième chapitre présente le programme écrit en langage JAVA ***Analyseur-de-Performances***. Il s'agit d'un parseur.....

- Le sixième chapitre a pour sujet la modélisation objet de notre réseau. Le modèle UML que nous présentons est une abstraction supplémentaire de la réalité.

- Le septième et dernier chapitre fournit les conclusions et des perspectives sur cette étude.

Chapitre 1

Les Transports Publics Genevois(TPG)

1.1 Historique

"Les Transports Publics Genevois" (**TPG**) sont une entreprise publique autonome de transport en commun qui couvre la région de Genève en SUISSE ainsi que les départments francais de l'Ain et de la Haute-Savoie. Sa création remonte au 1^{er} Janvier 1977 [TPG 07] et succède à la **CGTE** (**C**ompagnie **G**enevoise des **T**ramways **E**lectriques). Elle emploie 1550 personnes et exploite un réseau en pleine expension, desservi par des *tramways*, des *autobus* et des *trolleybus*.

1.2 Statistiques

En termes de nombres [TPG 07], les Transports Public Genevois peuvent être décris par :

- 343'000 voyageurs transportés en moyenne par jour
- 125'355'000 voyageurs transportés au total par année
- 18.8 millions de Km productifs parcourus
- 1550 membres du personnel
- 392 véhicules à moteur
- 57 lignes exploitées parmis lesquelles :
  - 5 lignes de tram
  - 6 lignes de trolleybus
  - 46 lignes d'autobus.

Fig 1 : Plan du Tramway à Genève, les lignes 12, 13, 15, 16, et 17 (version du 07/08/2007)

Chapitre 2

Les principaux outils de modélisation

2.1 Les Réseaux de Pétri

Les réseaux de Petri ont été introduits en 1962 par le mathématicien Allemand Carl Adam Petri [Petri, 62]. Ils présentent trois caractéristiques importantes :

- La modélisation et la représentation graphique de comportements intégrant du parallélisme, de la synchronisation et du partage de ressources.
- La vérification des propriétés possibles car basés sur un formalisme mathématique rigoureux.
- La représentation à différents niveaux d'abstraction.

2.1.1 Les composants élémentaires.

Un ***Graphe*** simple **G** est un couple formé de deux ensembles : un ensemble **X** = {X1,X2,.....,Xn} dont les éléments sont appelés *noeuds*, et d'un ensemble **A** = {A1, A2,A3,.....,Am} dont les éléments sont appelés *arêtes*. Dans le graphe, on peut attribuer des "noms" (valeurs) aux noeuds et des pondérations aux arêtes.

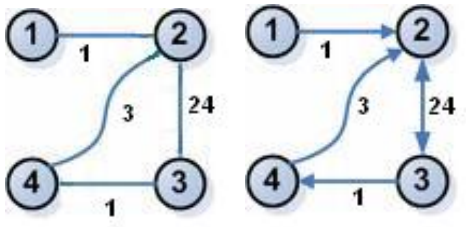


FIG. 2.1 – Exemple de graphe (gauche) et graphe orienté (droite)

2.1.2 Places, Transitions, Arcs et jetons

Les réseaux de Petri sont des graphes - des figures formées de lignes (*arêtes*) et de points de jonction de ces lignes (*noeuds*) - qui représentent des systèmes à évènements discrets. Ces graphes sont dits *orientés* et sont composés de deux types de noeuds :

- des **Places**.
- des **Transitions**.

Ces noeuds contiennent des marques appelées **jetons** et sont réliés entre eux par des arcs orientés. Un **Arc** relie soit une transition à une place, soit une place à une transition et doit absolument avoir à son extrémité une place ou une transition.

Formellement, un réseau de Petri est défini comme un quintuplet R = <P, T, A, Pre, Post> :

- P = {P1, P2, P3,...,Pn} est l'ensemble fini des places.
- T = {T1,T2,T3,...,Tm} est l'ensemble fini des transitions.
- A = {A1, A2,.....,An} est l'ensemble d'arcs A $\subset$ {P$\times$T}$\cup${T$\times$P}
- Pre : P $\times$ T $\longmapsto$ IN est l'application places précédentes
- Post : P $\times$ T $\longmapsto$ IN est l'application places suivantes

2.1.3 Exemple 1 : Les nombres Pairs et Impairs

On considère un système pour vérifier la parité des entiers naturels illustré par un Rdp à arcs inhibiteurs (figure 2.1.3a). *Un* ***arc inhibiteur*** relie une place à une transition exclusivement. On lui associe un poids de 0, en ce sens que la transition sera déclenchée si la place ne contient aucun jeton.

- La place **P0** contient un nombre de jetons dont la valeur représente le nombre à tester.

- Si la place **P0 Pair** contient un jeton, cela signifie que l'entier naturel (la valeur du nombre de jetons) en **P0** est *pair*.

- Les arcs normaux ont une flèche à leur extrémité. L'arc de poids égale à deux qui va de la place **P0** à la transition **T0** indique la consommation de deux ressources (jetons) de la place **P0**.

- Les arcs inhibiteurs ont un point à leur extrémité. L'arc **P0→T1** indique qu'on peut tirer la transition T1 uniquement quand il n'y a aucun jeton dans **P0**. L'arc **P0 pair→T1** indique qu'on peut tirer la transition T1 uniquement quand il n'y a plus aucun jeton dans **P0 pair**.

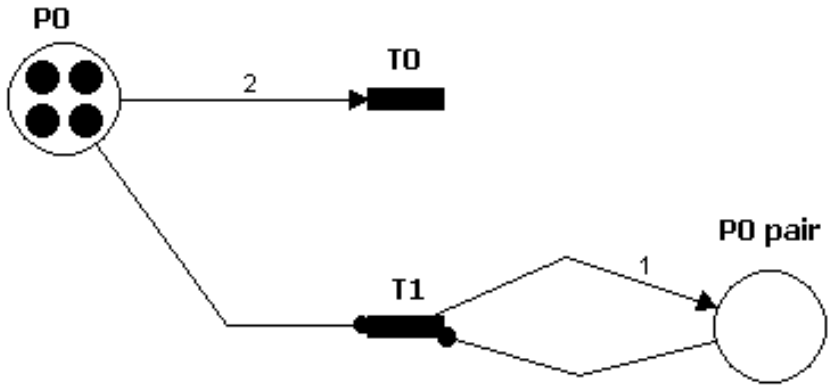


Fig 2.1.3a : exemple 1 : Modélisation des nombres Pairs ou Impair par un réseau de Petri.

On initialise (Rdp de la figure 2.1.3a) avec quatres jetons dans la place **P0**. Le marquage initial est alors M0 = (4, 0).

On tire successivement les transitions **T0**, **T0** et **T1** et on observe que les marquages intermédiaires sont respectivement M1 = (2 ,0), M2 = (0, 0), M3 = (0, 1).

Le Rdp de la figure 2.1.3b illustre, en trois étapes, l'exécution de La séquence de transitions **T0-T0-T1**.

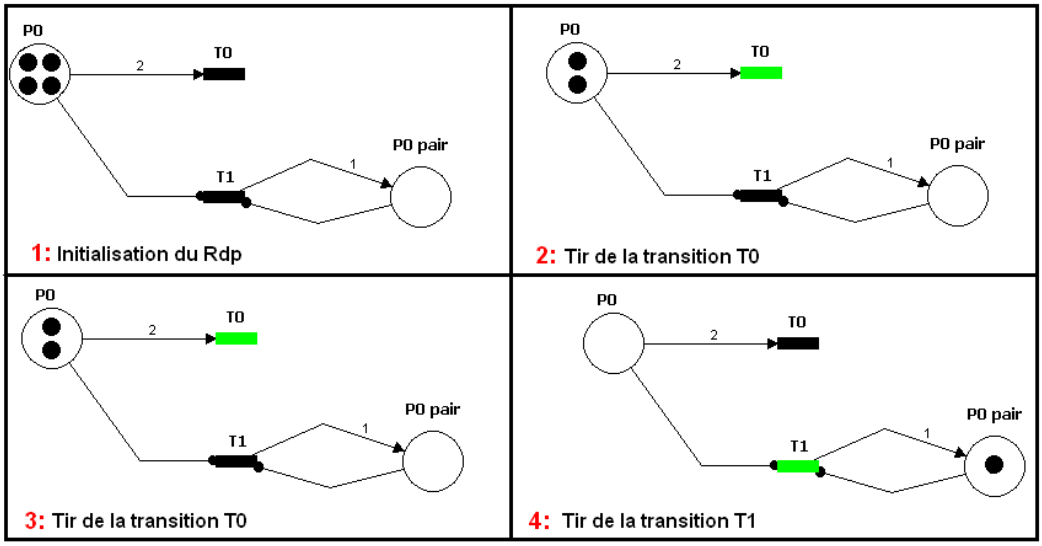


Fig 2.1.3b : Résultat exécution du Rdp de la figure 2.1.3a, Séquence de transition **T0-T0-T1**.

2.1.4 Exemple 2 : Un carrefour de deux routes

On considère maintenant un système (figure 2.1.4) qui modélise un croisement de deux routes, au sens physique, dans lequel on *fixe les directions* des véhicules : la voiture rouge (Vr) et la voiture bleue (Vb). On modélise Vr et Vb par les jetons dans les places P0 et P2 respectivement. Le jeton dans la place P1 sert de direction pour la circulation. La place P1 ne correspond à aucune route, elle existe uniquement dans le contexte des réseaux de Petri. Ce paramètre sert à gérer les conflits de croisement, les transitions ne sont jamais "*tirable*" simultanément.

Dans le Rdp (figure 2.1.4) modélisant un carrefour de la route :

1. Chacune des cinq places contient un nombre entier (positif ou nul) de ***jetons***. Le jeton représente la valeur de la variable d'état associée à la place.

2. Le marquage du réseau est l'assignation de jetons aux places du réseau. Dans notre cas, le marquage initial est : M0 = (P0, P1, P2, P3, P4) = (1, 1, 1, 0, 0). Le marquage après le tir de T0 est : M1 = (0, 1, 1, 0, 1). Le marquage après le tir de T1 est : M2 = (0, 1, 0, 1, 1).

3. Une transition est dite *validée* si toutes les places en amont (i.e en entrée) de celle-ci possède au moins un jeton.

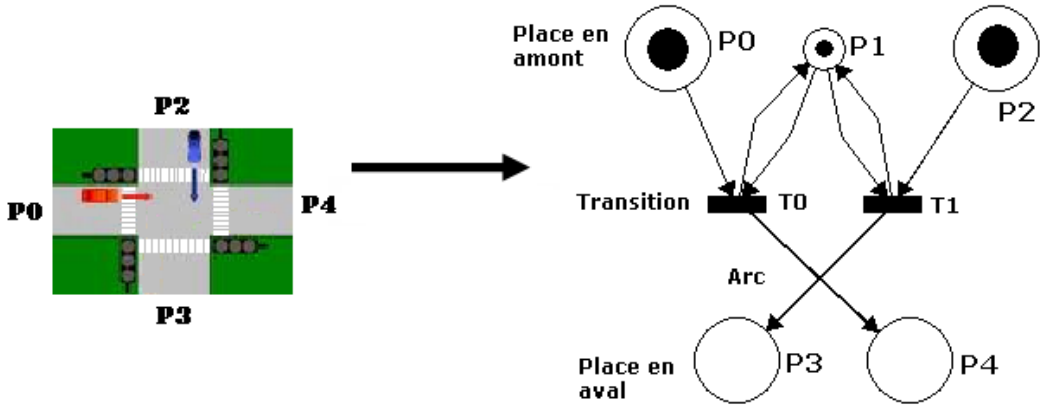


Fig 2.1.4 : A gauche un carrefour de deux routes, les flèches imposent les directions des véhicules. A droite le Rdp équivalent.

2.1.5 La modélisation par les réseaux de Petri

Les Rdp permettent de représenter graphiquement des scénarios de relations, de visualiser certains comportements simples. On pourrait par exemple visualiser l'exécution en parallèle de deux processus, après un *fork* [MENU], visualiser le comportement de deux processus qui doivent écrire sur une même zone de mémoire (partage de ressources), visualiser l'exécution d'une chaine de fabrication dans une usine (synchronisation).

La figure 2.1.5 illustre certains de ces comportements :

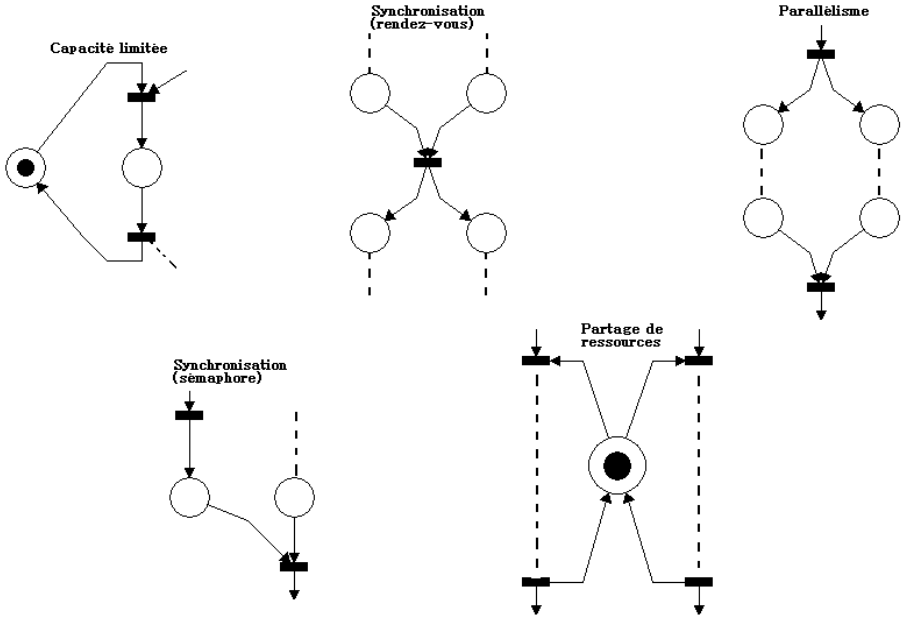


Fig 2.1.5 : Visualisation de quelques comportements par les Rdp

2.2 Les Extensions des réseaux de Petri

Les réseaux de Petri possèdent certaines extensions nécessaires à la vérification ou au contrôle d'application temps réels.

2.2.1 Réseaux de Pétri Colorés(RdpC)

Les Rdp classiques peuvent rapidement devenir "***très***" larges tant au niveau de la quantité des ressources dans les places qu'au niveau du nombre d'arcs. Avec l'extention de ces réseaux qu'on appelle "réseaux de petri colorés", on modélise de manière compacte des systèmes ayant des composantes au comportement indentiques. on réduit ainsi la taille des modèles creés.

les principales caractéristiques des RdpC :

- Les jetons sont "typés" par des ***couleurs*** en quantité finie. En d'autres termes, on donne des valeurs aux types de chaque jeton.

- On associe à chaque transition un ensemble de couleurs "valides". Ce sont les couleurs de franchissement de transition.

– Les places peuvent contenir plusieurs marques(types) distintes.

– On spécifie une *relation* entre les jetons consommés et les jetons produits. En d'autre termes, la valeur des jetons produits doit se référer à la valeur des jetons consommés.

Le réseau de Petri de la figure 2.2.1 modélise l'opération d'addition, la place "Somme P0" contient un jeton dont la valeur est la somme des valeurs des jetons de la place "P0".

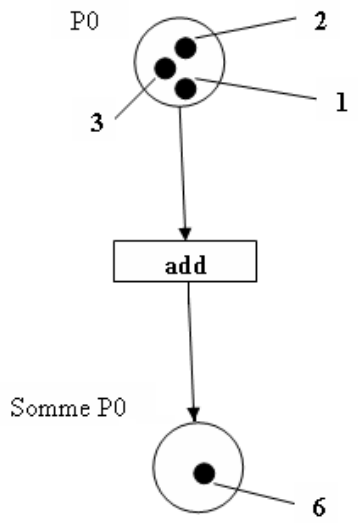


Fig 2.2.1 : Exemple de Réseau de Petri coloré.

2.2.2 Réseaux de Pétri Temporisés(RdpT)

Les RdpT introduisent la notion *d'information temporelle sur la durée des activités*. Dans ce contexte, une durée représente l'intervalle de temps [Tmin ; Tmax] pendant lequel une transition peut être franchie. Ces réseaux sont utiles pour évaluer les performances d'un système et sont caractérisés par :

– La référence unique du temps qui est fournie par les données du système : les RdpT dépendent d'un environnement.

– La puissance d'expression supérieure aux Rdp classiques.

– Deux modèles :

1. Les Rdp T-temporisés [BUC 04], les durées constantes sont attachées aux transitions.
2. Les Rdp P-temporisés [BUC 04], les durées constantes sont attachées aux places.

Il est à noter que ces modèles de RdpT comportent deux ***sémantiques temporelles*** :

– Une sémantique **FORTE** [BON 01] dans laquelle les contraintes temporelles (principalement la contrainte maximale) peuvent forcer le tir d'une transition : *La transition doit être tirée* !

– Une sémantique **FAIBLE** [BON 01] où, malgré le fait que toutes les conditions nécessaires au franchissement d'une transition soient réunies, cette dernière peut ne pas être tirée : *La transition peut être tirée !*.

Les types de transitions.

Dans ce projet, nous avons considéré les modèles de Rdp T-temporisés. Il existe trois variantes temporelles de paramètres à transitions :

– Les transitions *immédiates*. Ce sont les transitions similaires à celles des Rdp classiques, lorsque les conditions de tir de la transition sont réunies (toutes les places en amont de la transition ont des jetons !), les jetons sont directement disponibles après le tir de la transition.

– Les transitions *déterministes*. Ce sont des transitions retardées. Après le tir d'une transition, les jetons sont disponibles uniquement au temps $t \geq \delta$, δ étant la durée de franchissement de la dite transition.

– Les transitions *stochastiques*. Avec ces transitions on gère des évènements incertains. Les durées de franchissement sont des variables aléatoires qui suivent des lois de probabilité. Dans la section suivante (Chap 2.2.3), nous présentons plus en détails les types de transitions stochastiques.

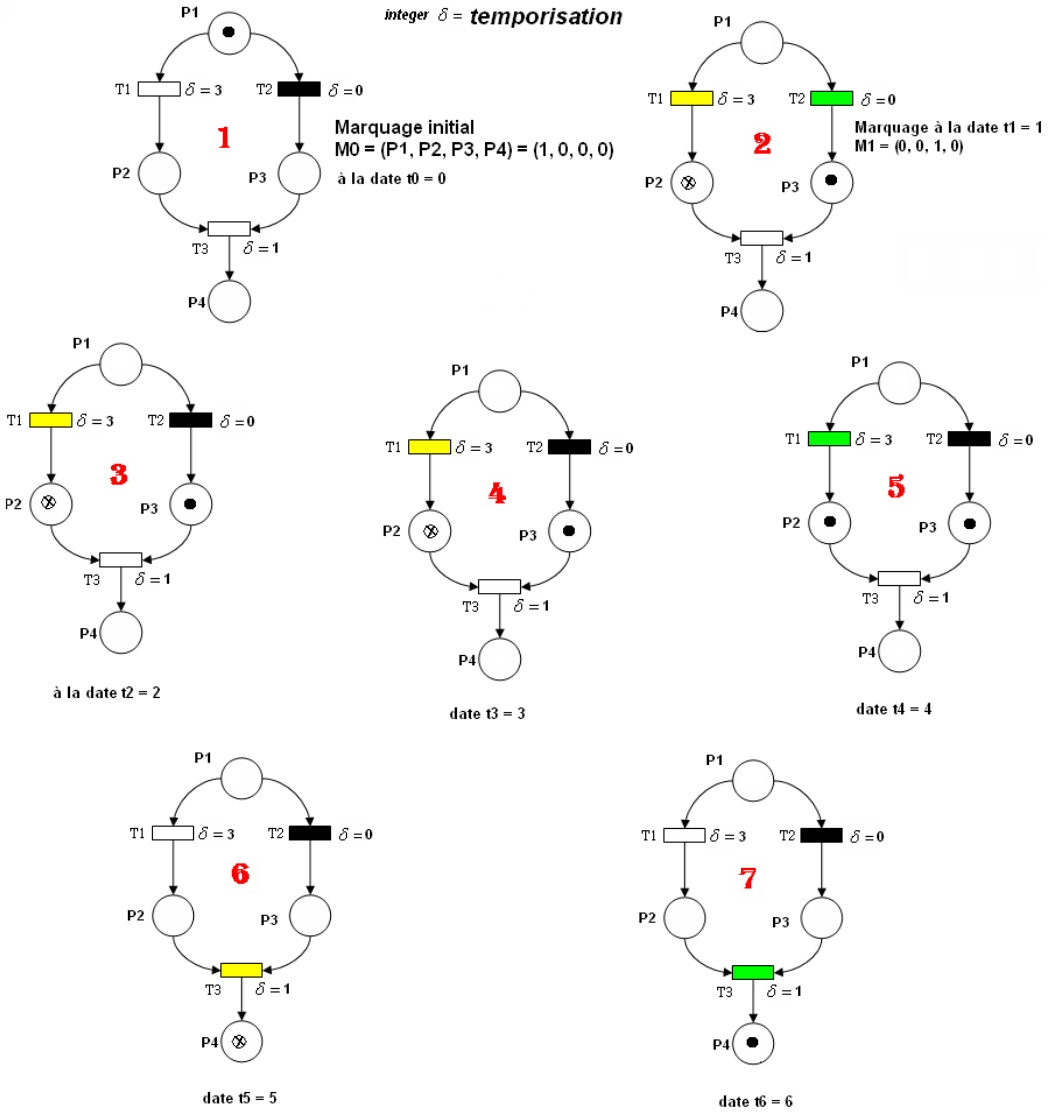


Fig 2.2.2 : Exemple de fonctionnement Réseau de Petri T-temporisé.

Description des RdpT de la figure 2.2.2, Les marquages intermédiaires nous renseignent sur l'état de notre système à des instants précis. La simulation se fait avec le logiciel HPsim.

- Le Rdp numéro 1 : On initialise notre réseau à la date t0 = 0, le marquage est alors M0 = (1, 0, 0,0).

- Le Rdp numéro 2 : "Sensibilisation" de la transition T1 et "tir" de la transition T2, marquage M1 = (0, 0, 1, 0). Le jeton est "*disponible*"

dans la place P3. Le jeton de la place P2 est "*réservé*"(non disponible).

- Le Rdp numéro 3 : La transition T1 est en état de "sensibilisation", marquage M2 = (0, 0, 1, 0). Le jeton est "*disponible*" dans la place P3. Le jeton de la place P2 est "*réservé*" (non disponible).

- Le Rdp numéro 4 : La transition T1 est toujours en état de "sensibilisation", marquage M3 = (0, 0, 1, 0). Le jeton est "*disponible*" dans la place P3. Le jeton de la place P2 est "*réservé*"(non disponible).

- Le Rdp numéro 5 : "tir" de la transition T1, marquage M4 = (0, 1, 1, 0). Le jeton de la place P2 est maintenant "disponible".

- Le Rdp numéro 6 : On sensibilise la transition T3, marquage M5 = (0, 0, 0, 0). Le jeton est "réservé" dans la place P4.

- Le Rdp numéro 7 : On tire la transition T3, marquage M6 = (0, 0, 0, 1). Le jeton est maintenant "disponible" dans la place P4.

2.2.3 Réseaux de Petri Stochastiques (RdpS)

Les RdpS sont aussi une extension "temporelle" des Rdp ordinaires. A chaque transition, on associe une variable aléatoire qui spécifie la durée qui sépare l'instant de sensibilisation de la transition et l'instant de tir de la dite transition.
L'ensemble des valeurs aléatoires X(t), avec **t** comme paramètre réel, décrit un processus stochastique. On distingue :

- *les processus stochastiques à temps discret* [LEF 05] pour lesquels l'espace du paramètre temporel **t** est l'ensemble T = {0, 1, 2, 3,.....}.

- *les processus stochastiques à temps continu* [LEF 05] oú T = [0, ∞[.

Les processus stochastiques [RechOP] modélisent en temps continu des phénomènes physiques parmis lesquels on peut citer :

- Le nombre de clients dans une file d'attente à un instant donné.

- Le nombre de trams arrivant à un arrêt pendant un intervalle de temps [0, t].

– La détection d'un signal de communication.

– etc...

Exécution de transitions Stochastiques.

Le pouvoir d'expression d'un modèle RdpS est en grande partie exprimé par les variables aléatoires temporelles associées aux différents types de transitions. Ces transitions vont être par la suite (cf Chap 4) paramétrisées à l'aide des informations de l'outil de simulation (HPsim) que nous avons choisi. On distingue :

– Des transitions ***uniformément distribuées*** [HPsim tools].
La variable aléatoire associée suit une loi de distribution uniforme dans un intervalle donné. Par exemple, une distribution dans l'intervalle [1, 2].

– Des transitions ***exponentielles*** [HPsim tools].
La variable aléatoire associée suit une loi de distribution exponentielle de taux λ. La fonction densité de cette loi est : f(t)=$\lambda e^{-\lambda t}$, t $\geq$ 0.

Pour une distribution de probabilité exponentielle, la fonction :
Prob[T$\leq$ t] = 1 - $e^{-\lambda t}$, décrit la probabilité pour que la durée de franchissement soit inférieure à **t**.
Si un paramètre t suit une loi exponentielle de taux λ, le temps moyen sera $\frac{1}{\lambda}$. La figure 2.2.3 représente f(t)=$\lambda e^{-\lambda t}$ pour $\lambda = 1$.

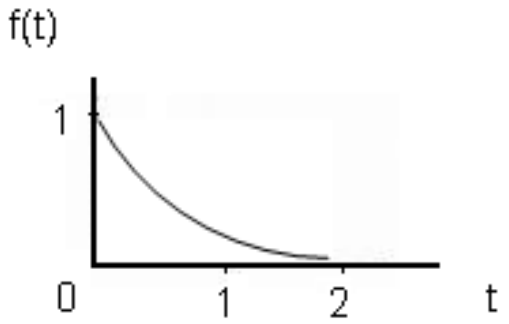


Fig 2.2.3 Loi exponentielle $\lambda = 1$.

Analyse d'un Rdp Stochastique.

La sémantique d'un réseau de Petri stochastique est définie selon deux critères importants :

- La sélection de la prochaine transition à franchir :

  1. par **compétition** [GAL 97], franchissement de la transition dont la variable aléatoire est statistiquement la plus petite.
  2. par **présélection** [GAL 97], on attribue aux transitions des probabilités de franchissement.

- La *mémoire temporelle* ou l'influence de franchissements sur les franchissement successeurs.

Les fonctions de densité de probabilité sur les intervalles de temps attribués aux transitions de ces RdpS permettent de manipuler des systèmes ou les informations traitées sont connues avec ***incertitude****. Dans notre contexte l'incertitude peut être une perturbation du réseau ou un besoin de flexibilité du système.*

2.3 Le language UML (Unified Modeling Language)

UMl - version 1.1 de novembre 1997 [UML 00], des auteurs J. BOOCH, I. JACOBSON et J. RUMBAUGH, est le langage de modélisation orienté objet le plus connu et le plus utilisé au monde. Il fournit une panoplie d'outils permettant de représenter, avec des outils informatiques, des composants du monde réel : c'est une simplification (abstraction) de la réalité. Cet outil de modélisation est également un bon moyen, de maitriser la complexité de notre réseau de trams et de mieux comprendre son fonctionnement, ce qui est indispensable avant toute éventuelle implémentation.

2.3.1 Des modèles plutôt que du Code

La modélisation avec UML regroupe trois concepts [O'REI 06] importants :

1. Les ***vues*** qui montrent diverses projections d'une même représentation.

2. Les ***diagrammes*** définissent les données et les traitements qui leurs sont attribués.

3. Le ***modèle*** qui est une abstraction de la réalité. L'abstraction est un processus qui consiste à identifier les caractéristiques intéressantes d'une entité, en vue d'une utilisation précise.

Ces trois concepts sont intimement liés : *Un* ***modèle*** *est une vue subjective de la réalité et une* ***vue*** *est constituée d'un ou plusieurs* ***diagrammes***.

On distingue deux types de vues :
- Les **vues statiques** qui représentent le système physiquement. Ce sont les diagrammes d'objets, les diagrammes de classes, les diagrammes de cas d'utilisation, les diagrammes de déploiements.

- Les **vues dynamiques** qui décrivent le comportement du système. Ce sont les diagrammes d'activité, les diagrammes de séquence, les diagrammes de collaboration, les diagrammes d'états-transition.

Cette modélisation permet de découvrir un système et son environnement, d'étudier les besoins des utilisateurs qui veulent interagir avec ce système et de dialoguer efficacement avec ceux-ci, même si ils ne sont pas *informaticiens*. Les interactions avec les systèmes informatiques sont représentées par des processus métiers. Ces processus décrivent l'ensemble d'activités exécutées dans une entreprise pour servir un client. Dans le domaine du genie logiciel, les modèles informatiques dérivent des modèles métiers [GLog 06].

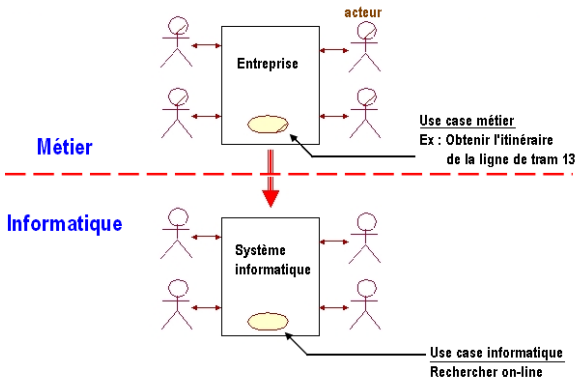


Fig 5 : Vue métier / Vue informatique

Chapitre 3

La Modélisation du Réseau de Transport

Dans le réseau de transports urbains de la ville de Genève que nous modélisons cohabitent plusieurs types de véhicules, des autobus, des trolleybus et des trams. Ces véhicules partagent parfois les mêmes voies pour un trajet donné. Les trams roulent sur des voies ferrées (paires de rails) spécialement conçues à cet effet. Le tracé d'un tramway est assez particulier, les lignes qui le composent se croisent plusieurs fois et s'entremêlent pour finalement n'irriguer qu'un secteur restreint de la ville : c'est le syndrome "*Spaghetti*".

3.1 Description d'un réseau physique

Au sens physique, nous représentons un réseau de trams par :

1. Un ensemble de ***segments***. Les segments sont des voies ferrées équipées de paires de rails.

2. Un ensemble de ***trams***.

3. Un ensemble de ***croisements*** qui servent de liaisons entre segments.

Nous présentons, dans la figure 3.1, une maquette du lieu dit "***PLAINPALAIS***", c'est un carrefour où se croisent quatres des cinq lignes de trams qui existent sur ce réseau : les lignes 12, 13, 15 et 17 , ces lignes partagent les mêmes segments en ce lieu.

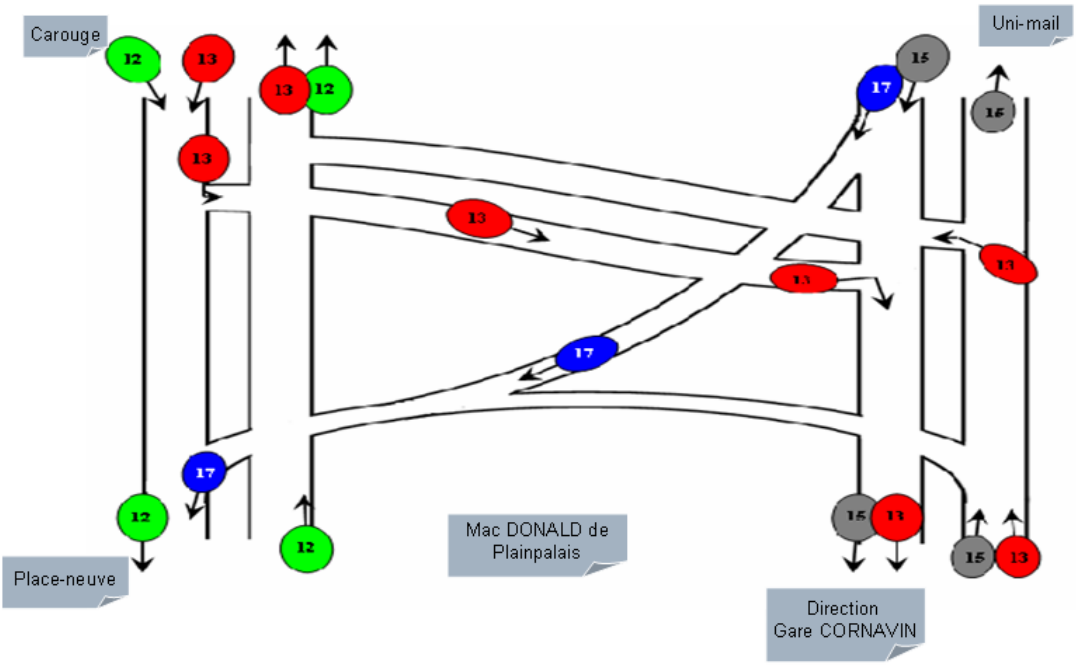


Fig 3.1 Segments de trams au carrefour de PLAINPALAIS à Genève.

3.2 Les règles syntaxiques et sémantiques de transformation : Réseau physique ↪ Réseau de Petri

Pour faire de la modélisation, il est important d'identifier les plus petites entités qui composent notre système. Nous donnons ainsi, pour chaque entité du modèle de réseau physique, son équivalent dans le modèle de réseau de Petri et tout cela suivant trois principales règles :

1. **Segments** = **Places**.
2. **Trams** = **Jetons**.
3. **Liens entre les segments** = **Transitions**.

Nous considérons que les *liens entre les segments* peuvent être des *Arrêts de tram* qui sont également modélisés par les transitions. On parle ainsi de Transitions *logiques* car on suppose que tout arrêt de tram pourrait être déplacé.

3.2.1 Les Croisements

Dans un réseau de trams, les segments (paires de rail) se croisent en plusieurs points. Ces points sont appelés ***croisements***, ils peuvent très rapidement devenir "*spaghettis*". Il est donc important d'optimiser la construction des croisements.

Fig 3.2.1 : Des segments de lignes de trams

Nous définissions des croisements de base selon la syntaxe :
[C/N *entrées*, **M** *sorties***]**. Il faut noter que les croisements que nous présentons sont "sans choix" pour les trams, en ce sens que les directions sont imposées sur les figures que nous présentons. Ce qui illustre bien le fonctionnement d'un réseau de trams, en effet on ne saurait envisager un réseau dans lequel les trams choisissent eux mêmes leurs directions.

Un réseau physique contient un ou plusieurs croisements de base :

- **[C**/1, 1**]** pour croisement 1 entrée, 1 sortie (c'est une paire de rails !).
- **[C**/1, 2**]** pour croisement 1 entrée, 2 sorties.
- **[C**/2, 1**]** pour croisement 2 entrées, 1 sortie.
- **[C**/2, 2**]** pour croisement 2 entrées, 2 sorties.
- **[C**/3, 3**]** pour croisement 3 entrées, 3 sorties.
- etc.....

Voici quelques exemples avec les Rdp équivalent :

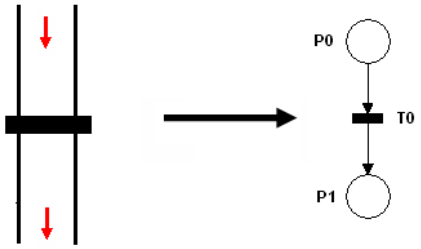


Fig 3.2.1a A gauche, une paire de rails (Segment). A droite le Rdp équivalent.

Dans la figure ci-dessus, la partie de gauche illustre le cas le plus simple d'une paire de rails (Un segment) divisé en deux parties par une barre noire (Un arrêt de Tram) et la partie de droite représente le Rdp équivalent. Les deux parties du segment sont modélisées par les places **P0** et **P1** et l'arrêt de Tram est modélisé par la transition **T0**.

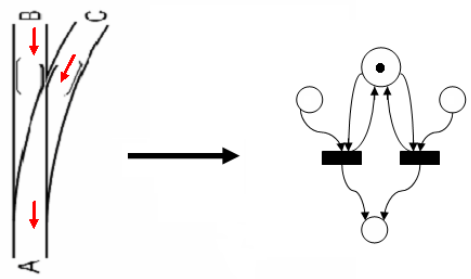


Fig 3.2.1b croisement [C/2,1] 2 entrées, 1 sortie

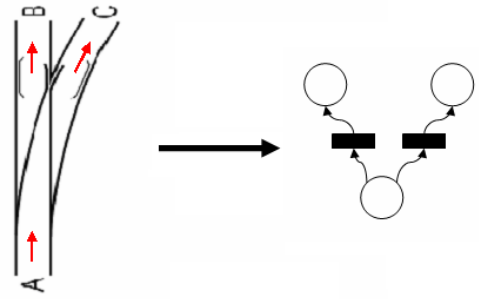


Fig 3.2.1c croisement [C/1,2] 1 entrée, 2 sorties

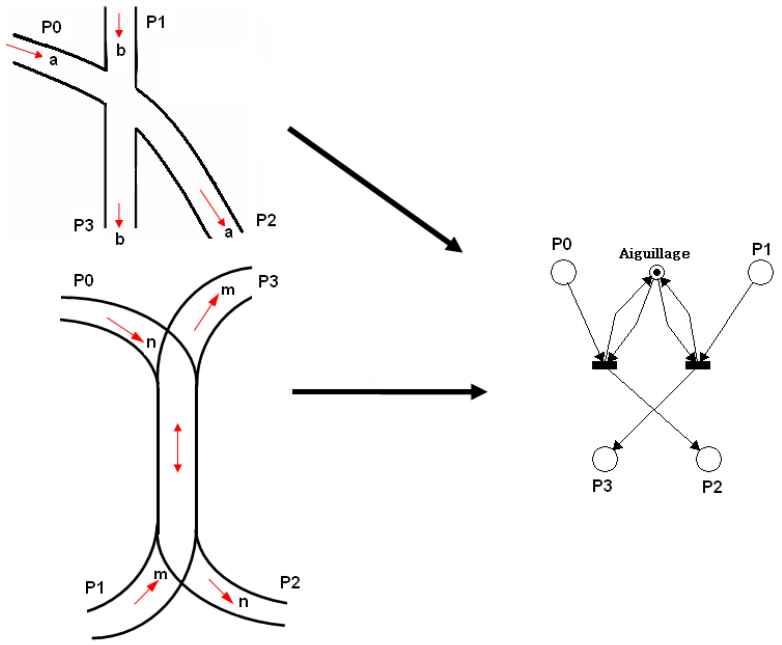


Fig 3.2.1d croisement [C/2,2] 2 entrées, 2 sorties

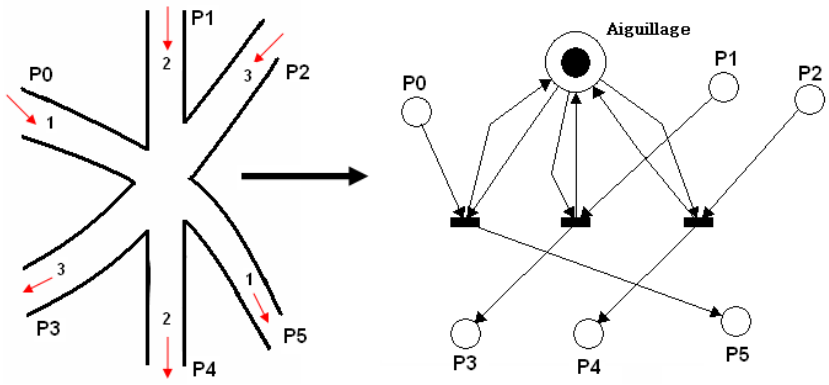


Fig 3.2.1e croisement [C/3,3] 3 entrées, 3 sorties.

Après avoir modélisé tous les croisements élémentaires qui peuvent composer un réseau, nous pouvons maintenant identifier (figure 3.2.1d) les croisements qui figurent sur notre maquette : Il s'agit de l'exemple de la figure 3.1 pour le modèle de Plainpalais.

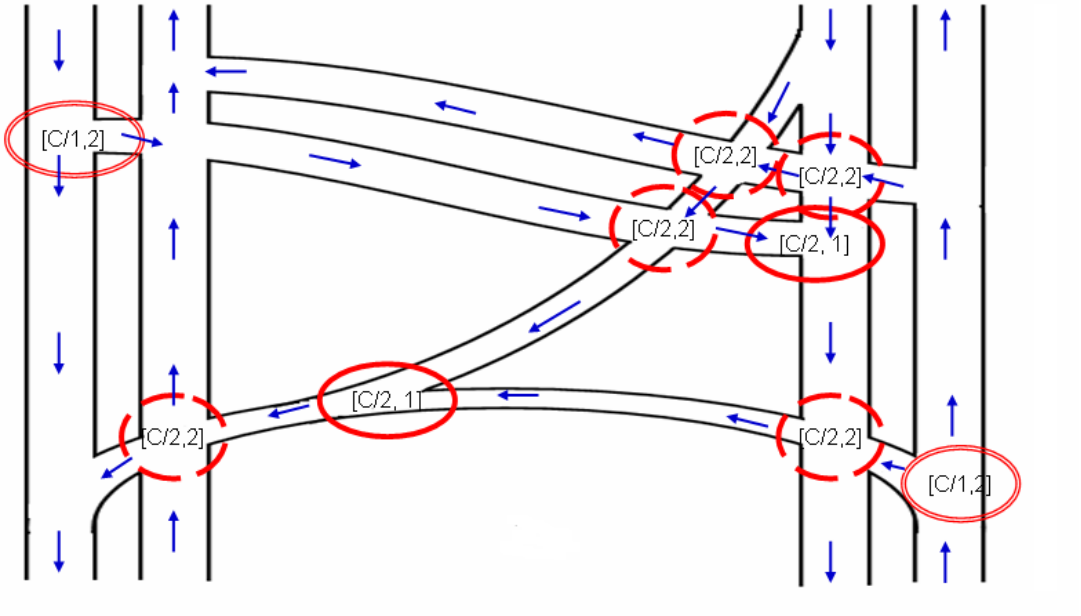


Fig 3.2.1d Le réseau physique Plainpalais et ses croisements.

3.2.2 Représentation conjointe "Réseau de Petri", "Tram", "Ligne de Trams" et "Segment physique"

Le formalisme des Rdp, par sa représentation graphique, est bien adaptée [SUP 06] à l'étude du fonctionnement des systèmes parallèles tels que les réseaux de transports. Il peut être intéresant d'avoir une vue d'ensemble de "*ce qu'on veut modéliser*" et de l'"*outil de modélisation*". C'est dans cette optique que nous avons proposé la modélisation conjointe (figure 3.2.2) d'un réseau de Petri, d'une ligne de trams et d'un segment physique avec ses trams. Cette figure identifie clairement ces entités, ainsi on a un segment (couleur noire) qui se divise en deux voies, un Rdp (couleur bleue), des jetons colorés dans le Rdp qui ont la valeur de "Trams", deux lignes de Trams. Pour déterminer une ligne de trams, on attribue des couleurs aux transitions, la ligne *12* est modélisée par les transitions **T0** et **T1**. Et la ligne *15* est modélisée par la transition **T2**.

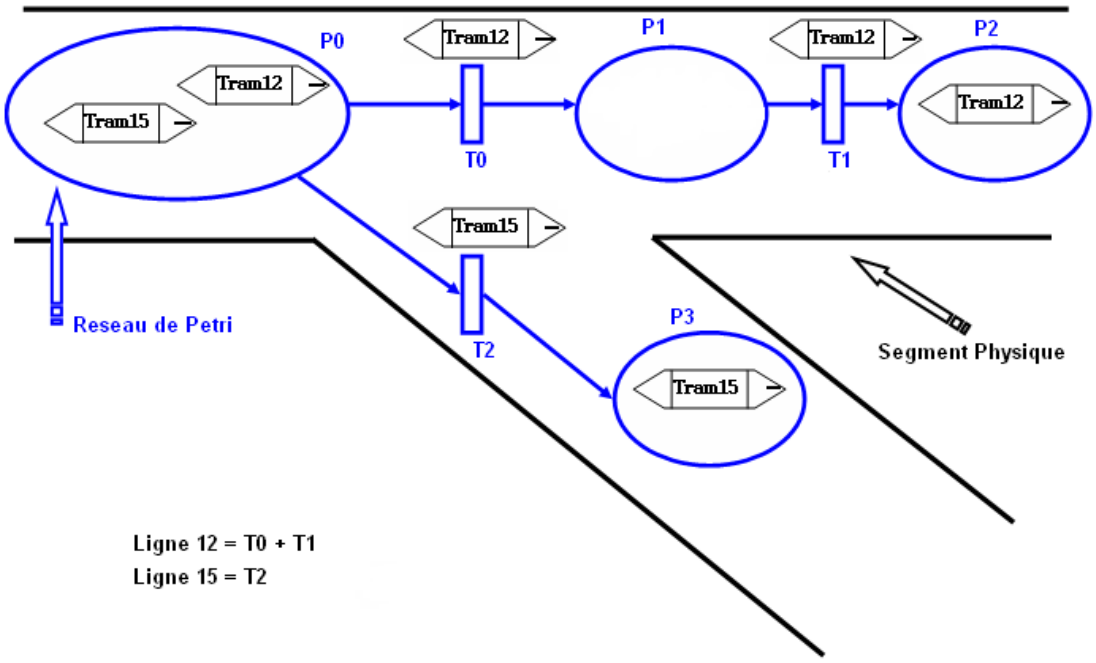


Fig 3.2.2 : Vue conjointe Réseau de Petri, Trams, Ligne de Trams, Segment physique.

3.3 Transformation d'un modèle physique en réseau de Petri : la maquette PLAINPALAIS

Dans la section précédente, nous avons défini les différents croisements de base. Tout réseau physique peut ainsi être transformé en Rdp par l'identification de ses composants (segments, croisements) et leur mise en commun. Cette transformation est illustrée par les figures 3.3.1 et 3.3.2.

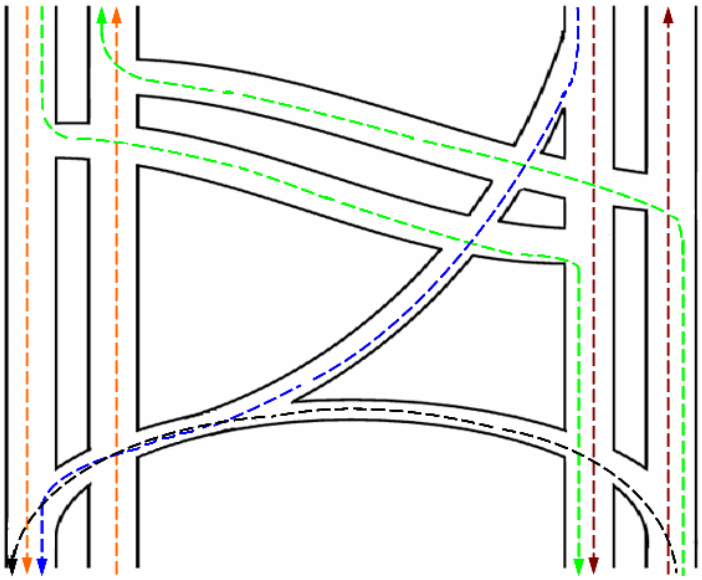


Fig 3.3.1 : Tramway carrefour Plainpalais (Genève)

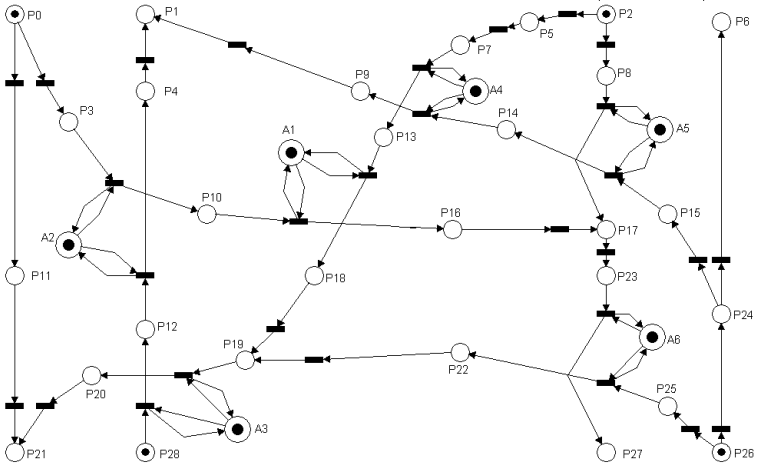


Fig 3.3.2 : Réseau de petri équivalent carrefour Plainpalais (Genève)

Description du Rdp de la figure 3.3.2 :

- Les transitions représentent les arrêts de trams.
- Les places sont les segments(rails de lignes de trams).
- Les jetons représentent les trams.
- Les places A1, A2,..., An ne correspondent à rien dans la réalité. Elles existent uniquement dans le contexte des Rdp et servent ici d'aiguillage dans les croisements de voies de trams.
- La ligne de tram **12** (direction *Moillesulaz*) est modélisée par les places : P0, P11, P21.
- La ligne de tram **12** (direction *Bachet-de-Pesay*) est modélisée par les places : P28, P12, P4, P1.
- La ligne de tram **13** (direction *Nations* via *Carouge*) est modélisée par les places : P0, P3, P10, P16, P17, P23, P27.
- La ligne de tram **13** (direction *Palettes* via *Carouge*) est modélisée par les places : P26, P24, P15, P14, P9, P1.
- La ligne de tram **15** (direction *Nations* via *Acacias*) est modélisée par les places : P2, P8, P17, P23, P27.
- La ligne de tram **15** (direction Palettes via *Acacias*) est modélisée par les places : P26, P24, P6.
- La ligne de tram **17** (direction *Gare des Eaux-Vives*) est modélisée par les places : P2, P5, P7, P13, P18, P19, P20, P21.

Chapitre 4

Transformations, Simulations et Analyses

Dans ce chapitre, nous introduisons le logiciel de simulation HPsim. Ensuite, nous appliquons, à des maquettes du réseau de trams Genevois, quelques extensions des modèles de Rdp dont la puissance d'expression est supérieure à celle des Rdp classiques :

- Transformation d'une maquette en réseau de Petri coloré(RdpC) et RdpC à file d'attente.
- Transformation d'une maquette en réseau de Petri temporisé(RdpT).
- Transformation d'une maquette en réseau de Petri stochastique(RdpS).

4.1 Le principal outil de simulation : Le logiciel HPsim

Le logiciel HPsim [HPsim tools] du docteur '**Henryk Anschuetz** (Allemagne) est un bon outil de simulation des réseaux de Petri. Il fonctionne sous windows XP et permet de :

- créer et simuler des Rdp places/transition.
- faire des animations avec des jetons.

– créer et simuler des Rdp avec places à capacité limitée.

– créer et simuler des Rdp temporisés (transitions immédiates, transitions déterministes).

– créer et simuler des Rdp stochastiques (transitions à variable aléatoire suivant une loi de distribution uniformément ditribuée, transitions à variable aléatoire suivant une loi de distribition exponentielle).

– obtenir le vecteur des marquages step by step, le vecteur est fournit en sortie comme information dans un fichier excel "*nomDuFichier*-OUT.csv".

– ...Mais malheureusement, on ne peut pas créer des Rdp colorés !

La figure 4.1 montre l'interface utilisateur du logiciel HPsim.

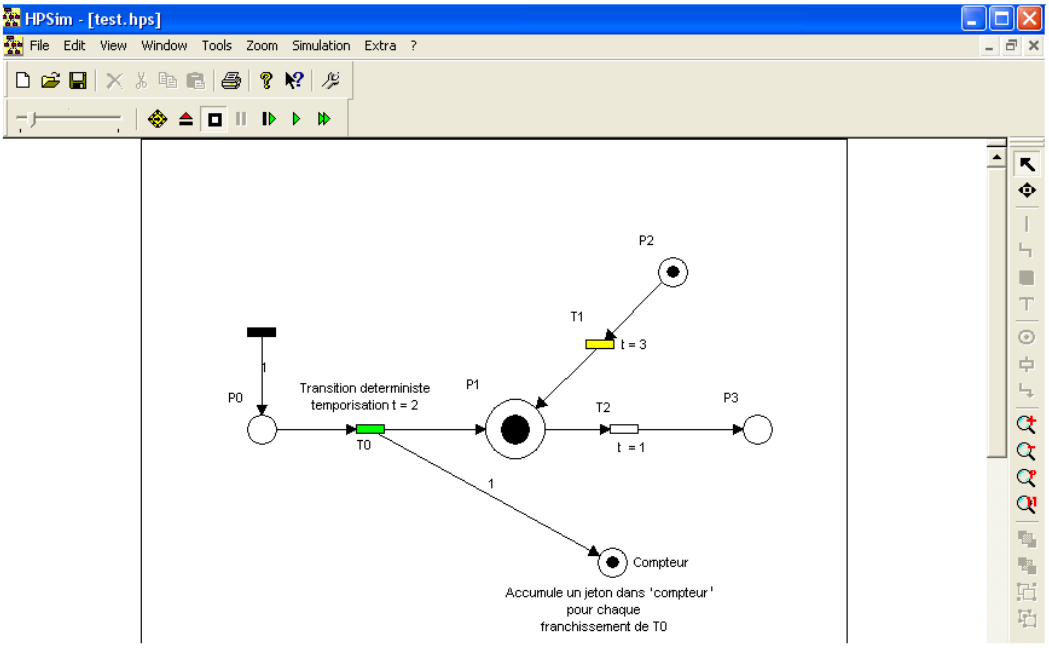


Fig 4.1 : Interface utilisateur de l'éditeur de reseau de Petri HPsim.

4.2 Modèle d'analyse

Notre analyse repose sur des entrées, une boite noire (Les logiciels de simulation) et des sorties. Etant donné un marquage initial M0, nous analysons, étape par étape le comportement du réseau à travers les marquages qui représentent les états du système.

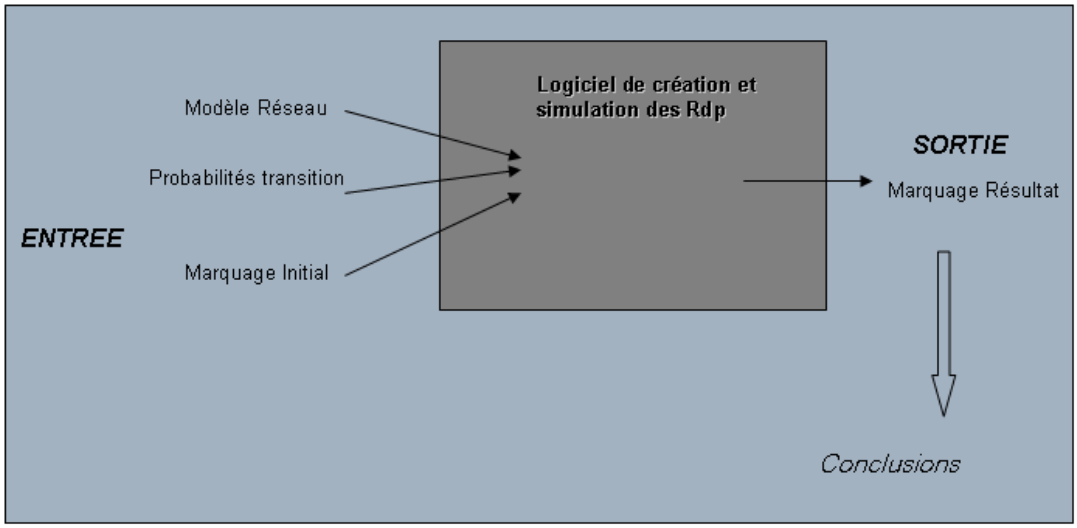


Fig 4.2 : Schéma scénario d'analyse

4.3 Simulation 1 : Transformation d'une maquette en réseau de Pétri coloré

4.3.1 Exemple 1 : scénario MOILLESULAZ

Nous présentons dans cet exemple un scénario de régulation des TPG au niveau de la douane de Moillesulaz (figure 4.3.1a) qui relie la SUISSE (par la ville de Genève) à la FRANCE (par la ville d'Annemasse) : *Le changement de ligne pour un véhicule (tram).*
A cet endroit précis il existe deux arrêts qui portent le même nom. Le changement de ligne est caractérisé par le fait que lorsque les trams se trouvent à ces deux arrêts, le numéro de ligne avec lequel ils y sont arrivés ne sont pas toujours conservés lorsqu'ils en repartent. Par exemple, un tram de la ligne 12 arrive à l'arrêt "Moillesulaz" marqué du numéro 12 et repart de cet arrêt avec le numéro 16, qui représente la ligne 16.

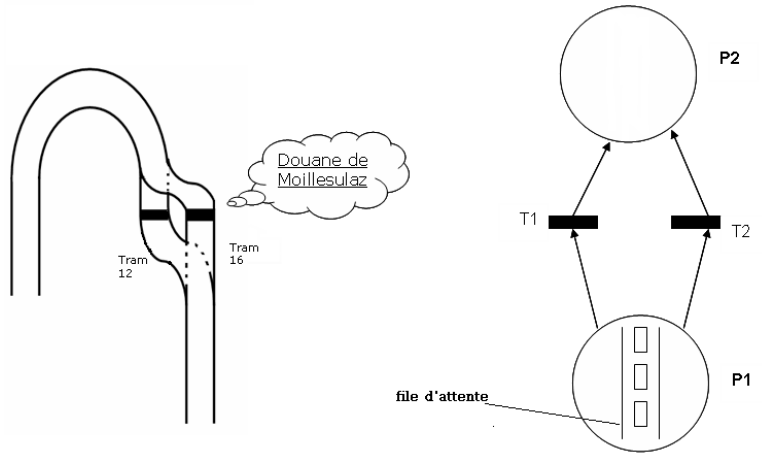


Fig 4.3.1a : A gauche, les lignes de tram à la douane de Moillesulaz. A droite, le Rdp coloré à file d'attente équivalent.

La syntaxe des couleurs est donné par la paire <"*nom du tram*" , *nombre*> ou "*nom du tram*" représente le numéro du tram et *nombre* représente le n'ième tram, dans un ordre croissant, de la ligne concernée. Nous modélisons le changement de ligne par le tir de transition. En effet les transitions T1 et T2 modélisent respectivement les lignes 12 et 16. Ce qui signifie que pour le tir la transition T1 (figure 4.3.1b), seule la couleur <12, nombre> est "validée" parmis toutes les couleurs qui se trouvent dans la place en amont de T1. Et lorsqu'on tir la transition T2, seule la couleur <16, nombre> est "validée".

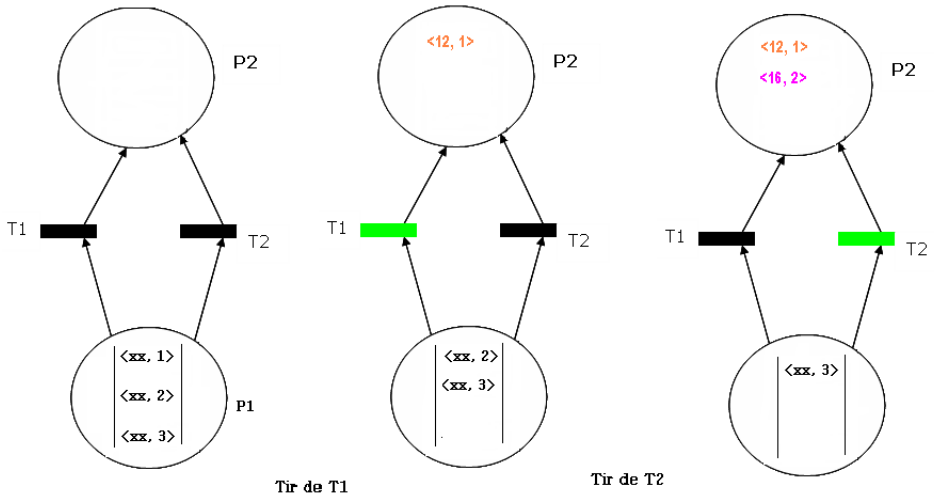


Fig 4.3.1b RdpC à file d'attente, tir des transitions T1 et T2.

4.3.2 Exemple 2 : Pliage et Dépliage (Comportements identiques)

La méthode de dépliage permet d'identifier automatiquement les comportements équivalents d'un système, et la méthode de pliage permet de les représenter de facon compacte. Le dépliage d'un système fini est infini [CHA 06] (une explosion du nombre de places et transitions dépliées est possible !), ce qui rend parfois la modélisation difficile. On devrait pouvoir définir des techniques de découpage fini qui préservent suffisament d'informations pour modéliser le système.

Nous allons considérer les lignes 12 (Bachet-de-pesay vers Moillesullaz) , 16 (Gare Cornavin vers Moillesullaz) et 17 (Bachet-de-pesay vers Gare des Eaux-Vives) qui partagent le même itinéraire à partir de l'arrêt *Bel-Air* jusqu'à l'arrêt *Roches* : [Bel-Air(cité) ↪ Molard ↪ Rive ↪ Terrassière ↪ Villereuse ↪ Roches].

Les RdpC de gauche de la figure 4.3.2 modélisent cet itinéraire avec ses lignes de tram.

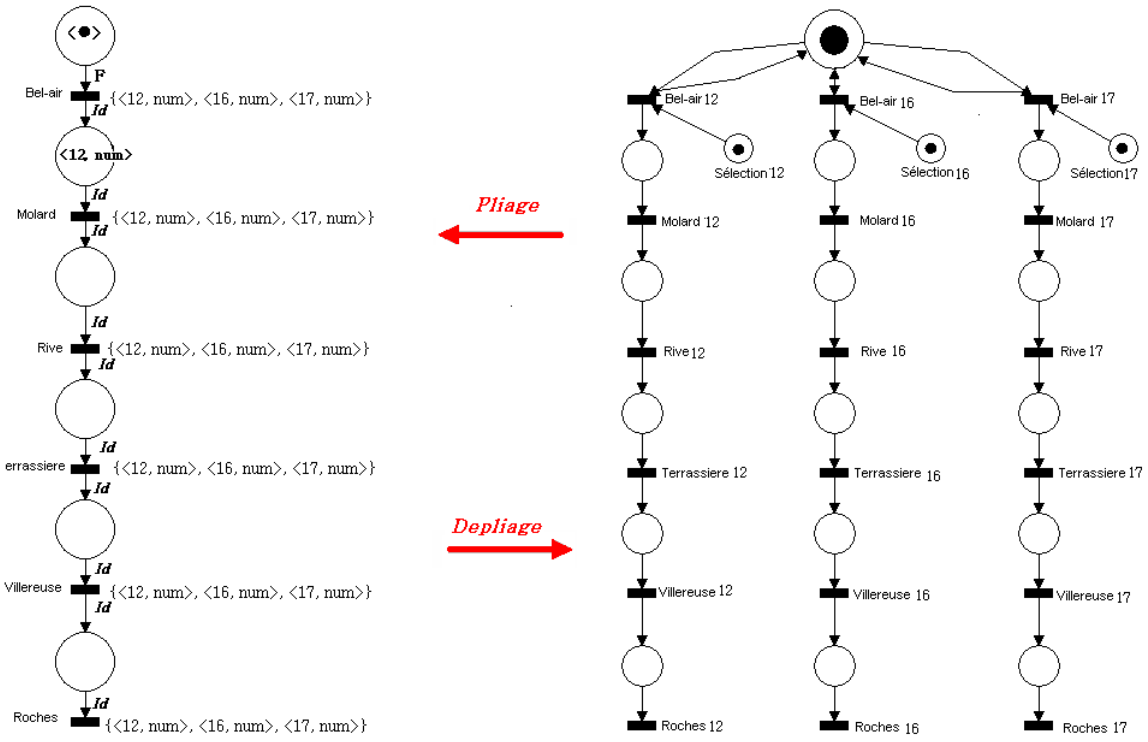


Fig 4.3.2 RdpC : Pliage et Dépliage.

Dans le RdpC de gauche (figure 4.3.2), on associe à chaque transition (ce sont les arrêts de tram) un ensemble de couleurs "valides". Ce sont les trams qui ont le droit de passer cette transition. La fonction Identité (***Id***) permet de conserver, d'une place vers une transition ou d'une transition vers une place, la ressource à traiter.
Le domaine des couleurs est l'ensemble,
D = {<•>, <12, num>, <16, num>, <17, num> } tel que :

- <•> est marque neutre
- <12, num> signifie tram numéro "num" de la ligne 12
- <16, num> signifie tram numéro "num" de la ligne 16
- <17, num> signifie tram numéro "num" de la ligne 17.

La fonction (***F***) établit une correspondance entre chaque couleur de la transition et les couleurs de la place telle que :

- ***F***(<12, num>) = <•>
- ***F***(<16, num>) = <•>
- ***F***(<17, num>) = <•>.

Le Rdp de droite (figure 4.3.2) est un dépliage de celui de gauche. Pour chacune des places ou transitions, on crée autant d'instances que son domaine de couleurs contient d'éléments. Ensuite on connecte ces places et transitions en "dépliant" les fonctions des couleurs.
Cependant, le dépliage d'un réseau n'est pas toujours l'expression du modèle d'origine due à une mauvaise interprétation de celui-ci.

4.4 Simulation 2 : Transformation d'une maquette en réseau de Pétri T-temporisé

"*Aider à la gestion de crise, c'est agir avant, pendant et après la crise*" [DUSS 03]. Il est important de bien noter la différence en termes de performances entre un réseau classique idéal (sans fuites d'informations) et un réseau temps-réel qui est un mélange <pas toujours apprecié !> de contraintes temporelles très souvent soumises à des évènements incertains. Dans un premier temps, nous modélisons les informations qui sont fournies par les tables horaires [TPG 07] des lignes de trams telles qu'elles existent. Le modèle standard issu de cette modélisation sera notre référentiel de base pour la gestion des conflits. Nous considérons ensuite un scénario dans lequel il existe des "perturbations" : Le modèle temps réel.

4.4.1 Régulation du réseau : Détection d'un état "anormal"

Un état "anormal" est une perturbation [FLA] (figure 4.4.1a).

Fig 4.4.1a : Une perturbation (un accident).

La **régulation** désigne l'ensemble des actions appliquées à un système de telle sorte qu'une valeur produite en sortie par celui-ci soit égale à la valeur consignée (celle qui est véritablement attendue !). Pour tout réseau de transport urbain, il est primodial de concevoir une bonne politique d'exploitation. Cette politique passe par la mise en place d'un système de *régulation* dont la tâche peut devenir complexe selon le nombre d'informations qu'il traite. La complexité de la tâche du régulateur nécessite un système d'aide à la décision (***SAD***) [KOW 06]. L'un des principaux objectifs du SAD est d'aider le régulateur à respecter au mieux les tables d'horaires. Le régulateur gère un ensemble de données parmis lesquelles :

- *Les horaires de circulation des différents trams* : Les horaires des trams sont inscrits, en fonction des itinéraires, dans des *tables horaires*.

- *Quelques caractéristiques des perturbations* :
  la ligne de tram et le tram impliqués dans le conflit.
  Le nombre de trams impliqués dans le conflit.
  Les lignes de tram connectées à la ligne perturbée.
  Le type de la perturbation qui peut être un retard, une avance, une panne, une manisfestation, un accident.
  l'horaire précis de la perturbation.
  Il est aussi important de noter la période (heure de pointe, heure creuse) à cet instant.

– *Les actions de base de régulation* : En période de crise, on pourrait envisager un changement de ligne pour des tram. On pourrait aussi ralentir le tram, l'accélérer ou même l'arrêter. A Genève la plus grande cause [TPG 07] de retards observés est liée à la circulation.

Certaines actions peuvent être générées automatiquement par le système sans intervention du régulateur. Ces actions font partie du processus de régulation que nous proposons dans la figure 4.4.1b :

– **Phase 1 :** Le processus commence tout naturellement par l'observation du réseau et des planifications théoriques des horaires qui lui sont attribuées.

– **Phase 2 :** Cette phase marque le début de l'analyse de l'état du système. On prévoit qu'un évènement déclencheur doit durer un minimum de temps pour faire intervenir le régulateur. Supposons par exemple qu'une poussette de bébé reste bloquée à la descente du tram, cette situation pourrait entrainer quelques secondes de retard qui ne nécessite pas forcément l'intervention du régulateur.

– **Phase 3 :** Notre travail est plus impliqué à ce niveau du processus de régulation, plus précisément pour la description de la perturbation. En effet, une situation "anormale" est déclenchée par comparaison du Rdp T-temporisé issu de l'état du système en temps-réel avec le modèle de Rdp théorique. Nous avons modélisé et simulé quelques horaires (section 4.4.2) avec un Rdp T-temporisé et nous présentons l'histogramme issu de l'analyse du réseau. Ces modèles vont être considérés comme référentiel. Le modèle temps-réel revèle les caractéristiques de la perturbation.

– **Phase 4 :** Nous avons cité (section 4.4.1) quelques exemples de solutions preétablies aux crises. Le régulateur filtre parmis ces solutions celle qui est adaptée à la perturbation. Lorsque la solution proposée ne règle pas totalement le problème, on retourne à la phase 3.

– **Phase 5 :** Le cycle du processus est maintenant en phase terminale, l'action qui a été retenue est validée et exécutée.

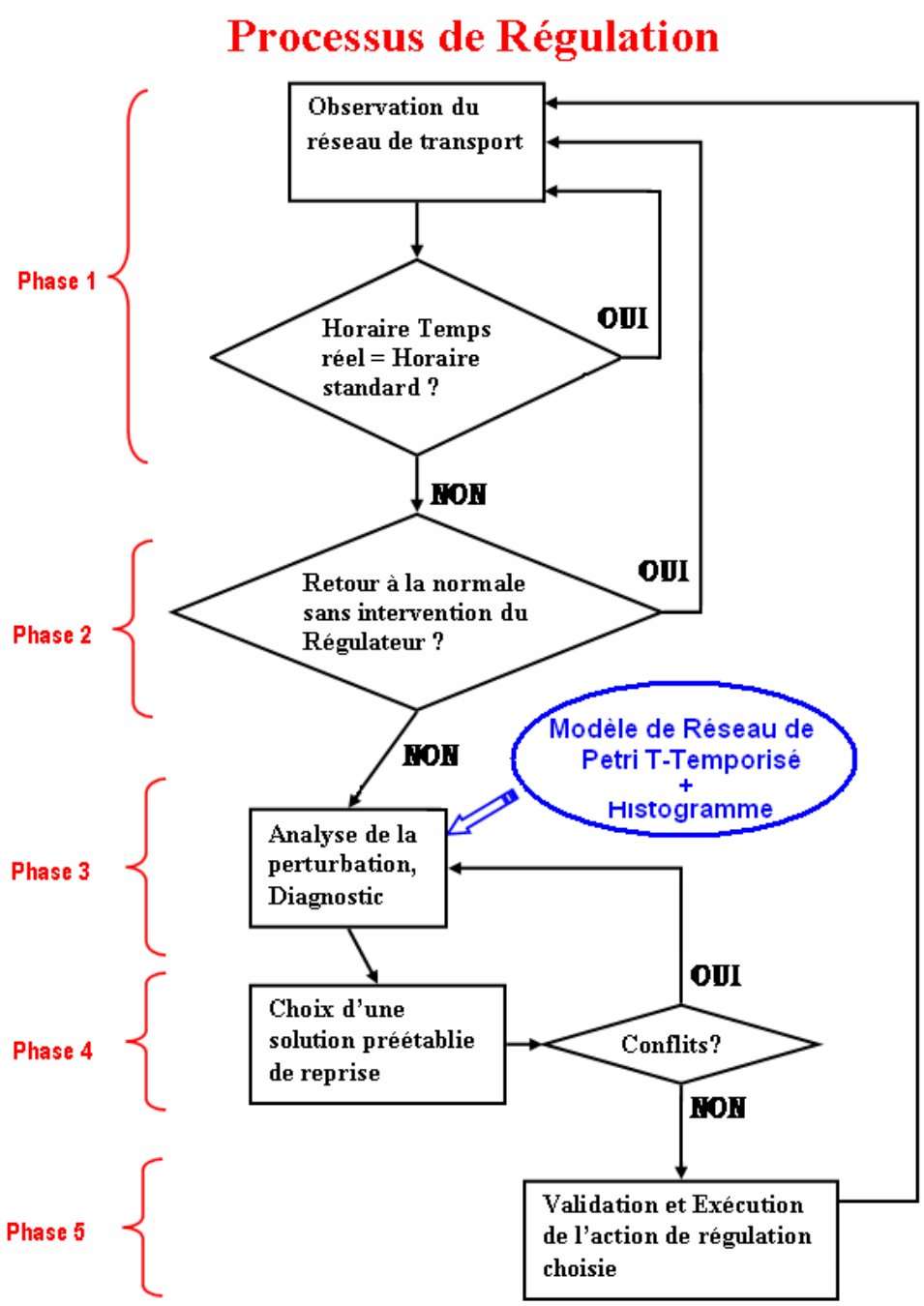


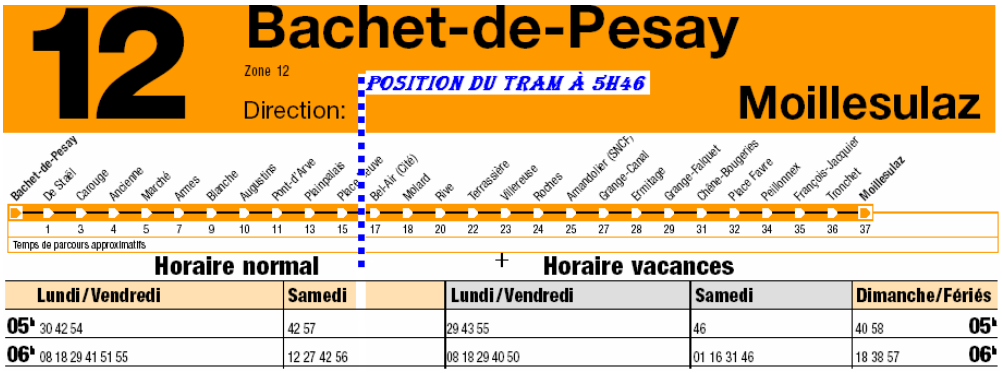
Fig 4.4.1b : Un modèle de processus de régulation

4.4.2 Gestion des tables horaires

La gestion des conflits passe par la prévention. Nous exploitons les résultats de la simulation des tables horaires pour identifier avec précision les éventuels évènements incertains qui peuvent se produire sur le réseau.

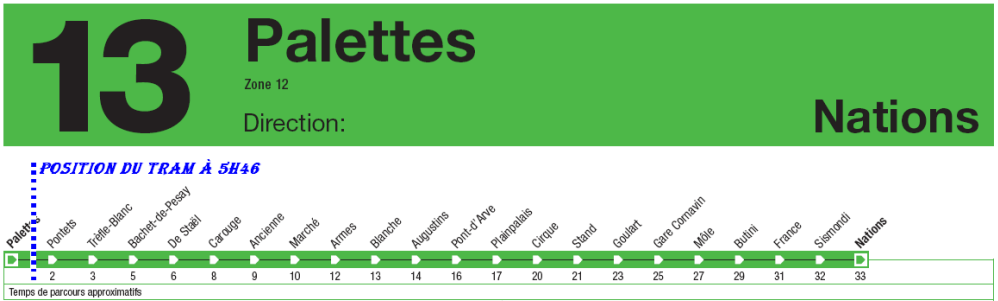
Pour la modélisation des tables horaires, on attribue à chaque transition une temporisation. La temporisation est la durée de franchissement des transition, qui représente la durée de parcourt approximatif entre différents arrêts de tram. La transition est dite ***déterministe***.

Nous avons considéré les tables horaires de trois lignes de trams qui se croisent au carrefour de *Plainpalais* et simulé les trajets parcourus par les véhicules. Les départs initiaux des véhicules sont donnés par les tables horaires, figure 4.4.1. On considère le temps de départ à 5h46 pour les trois lignes, et nous simulons et analysons les positions quelques minutes après chaque départ.



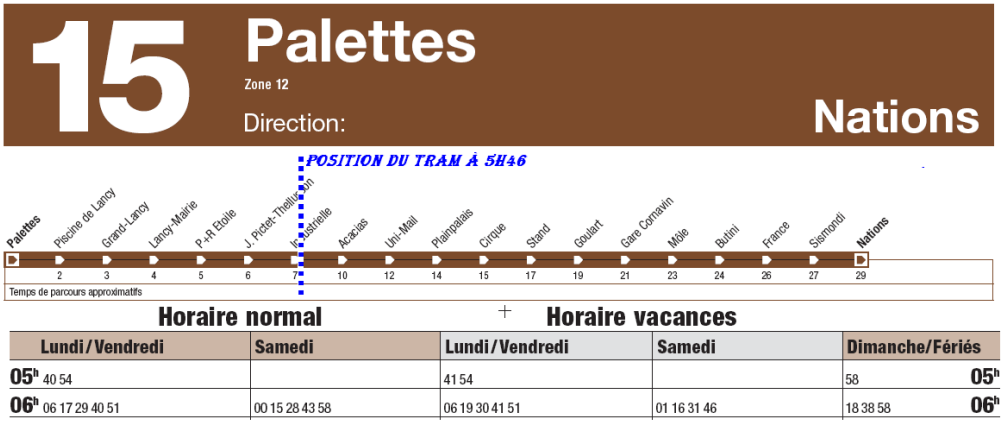
Horaire normal + **Horaire vacances**

Lundi / Vendredi	Samedi	Lundi / Vendredi	Samedi	Dimanche/Fériés	
05h 30 42 54	42 57	29 43 55	46	40 58	**05**h
06h 08 18 29 41 51 55	12 27 42 56	08 18 29 40 50	01 16 31 46	18 38 57	**06**h



Horaire normal + **Horaire vacances**

Lundi / Vendredi	Samedi	Lundi / Vendredi	Samedi	Dimanche/Fériés	
05h a 45 56	a 47	a 45 56	a 49	a 44	**05**h
06h 08 20 31 41 53	02 17 31 46	07 19 32 41 51	04 19 34 49	a 04 23 43	**06**h



Horaire normal + **Horaire vacances**

Lundi/Vendredi	Samedi	Lundi/Vendredi	Samedi	Dimanche/Fériés
05h 40 54		41 54		58 **05**h
06h 06 17 29 40 51	00 15 28 43 58	06 19 30 41 51	01 16 31 46	18 38 58 **06**h

Fig 4.4.2 Tables horaires : ligne 12(Bachet-de-Pesay→Moillesulaz), ligne 13(Palettes→Nations) et ligne 15(Indutrielle→Nations).

Un exemple de fonctionnement :.

Nous avons crée un Rdp pour modéliser ces tables horaires. Toutes les places qui modélisent les segments, ont une capacité d'un jeton, on suppose qu'il ne peut y avoir qu'un seul tram par segment. Les transitions qui modélisent les arrêt de trams ont des temporisations telles que définies par les tables horaires (figure 4.4.1). Après simulation, en considérant la position des trams à 5h46, il est intéressant de constater que, sept minutes (figure 4.4.2b) après le départ de chaque véhicule, deux trams ne se trouvent pas au même moment dans une place. Il en est de même 10 minutes (figure 4.4.2c) après le départ de chaque véhicule, les conflits sont gérés par les retards respectifs attribués à chaque transition.

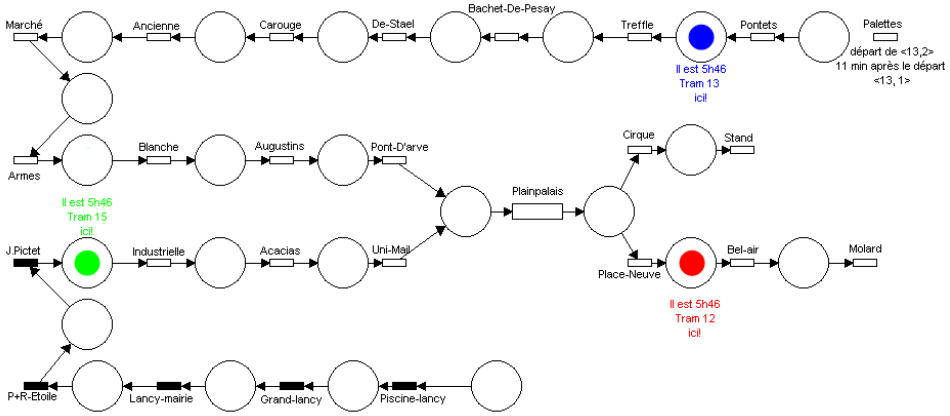


Fig 4.4.2a Tables horaires lignes 12, 13, 15 au temps 5h46

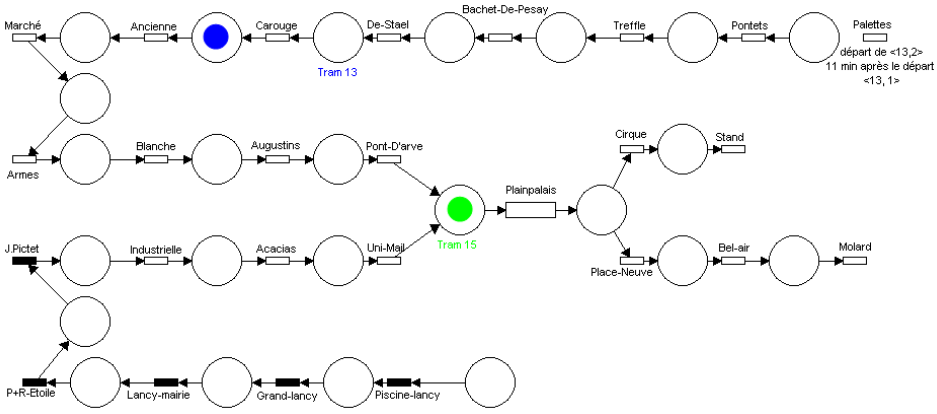


Fig 4.4.2b Tables horaires lignes 12, 13, 15 au temps 5h53

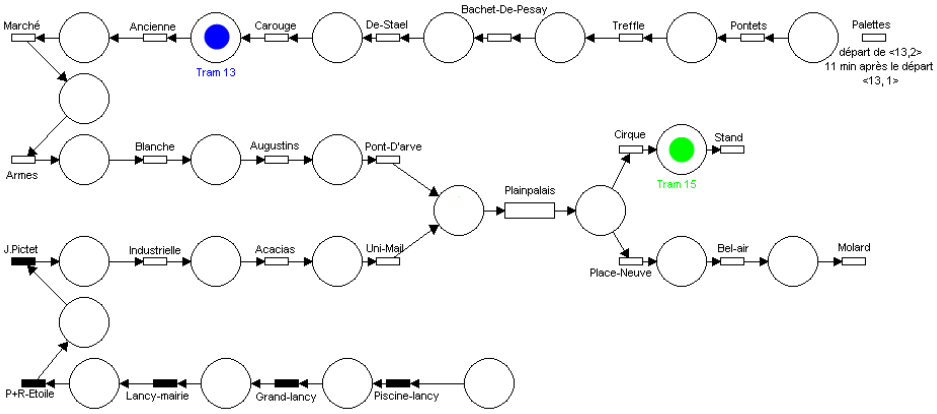


Fig 4.4.2c Tables horaires lignes 12, 13, 15 au temps 5h56.

Dans notre contexte, on parle de sémantique *Forte*. Les contraintes de temps attribuées aux transitions imposent le franchissement de la dite transition. Les transitions déterministes doivent être tirées pour que notre modèle exprime au mieux le traffic des trams sur les différentes lignes.

4.4.3 Modèle Standard et Histogramme

L'histogramme de la figure 4.4.3 donne une vue générale du positionnement des différents véhicules à des instants précis. Nous considérons que tous les trams suivent cette cadence dans les conditions normales (théoriques) : C'est le référenciel du modèle standard. On suppose que le régulateur du système d'exploitation est doté de capteurs qui donnent les positions des trams à tout instant. Le temps initial est choisi au temps t0 = 5h46. Le deuxième tram de la ligne 12 démarre 12 minutes (à 5h58) après son prédécesseur et suit

la séquence Place-neuve→ Bel-air→ Molard. Le deuxième tram de la ligne 15 démarre 14 minutes(à 6h00) après son prédecesseur et suit la séquence Industrielle→ Acacias→ Uni-mail

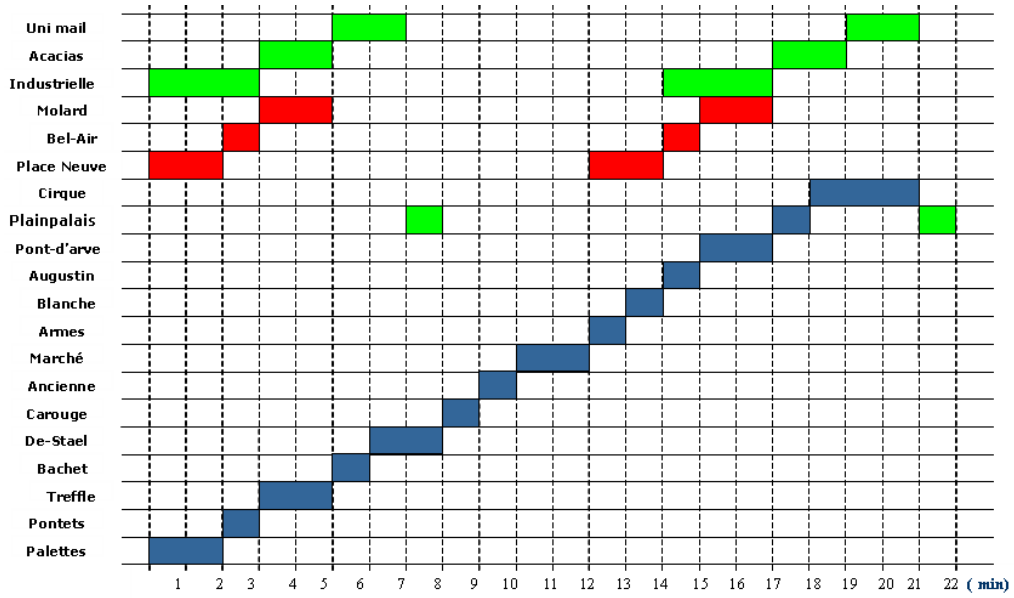


Fig 4.4.3 histogramme tables horaire lignes 12, 13 et 15. Positions :départ à 5h46

4.4.4 Modèle Temps-réel et Histogramme

Dans cette section, on va créer une "perturbation" Figure 4.4.4a : retard sur la ligne 12.

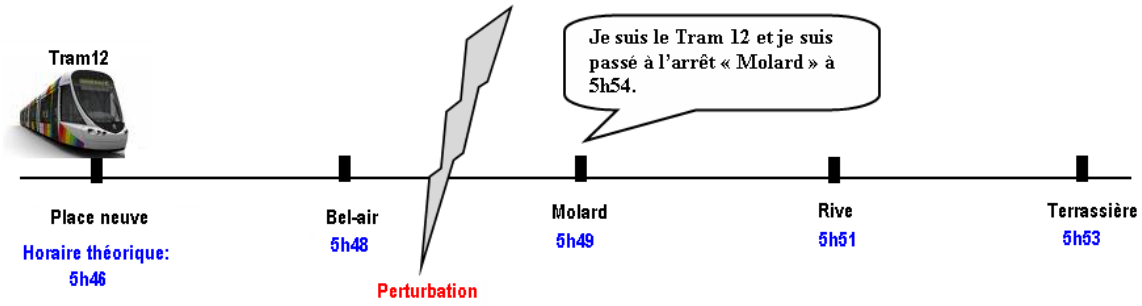


Figure 4.4.4a : Perturbation de la ligne 12.

Est-ce que chaque tram respecte le marquage théorique ? NON. Lorsqu'on compare les histogrammes des figures 4.4.3 et 4.4.4b on identifie avec précision à partir de quel instant la pertubation a eu lieu, quelle est la ligne de tram concernée : *la ligne 12*. On peut ainsi faire une estimation des temps de parcours qui vont suivre et éventuellement adopter une autre solution de régulation plus appropriée.

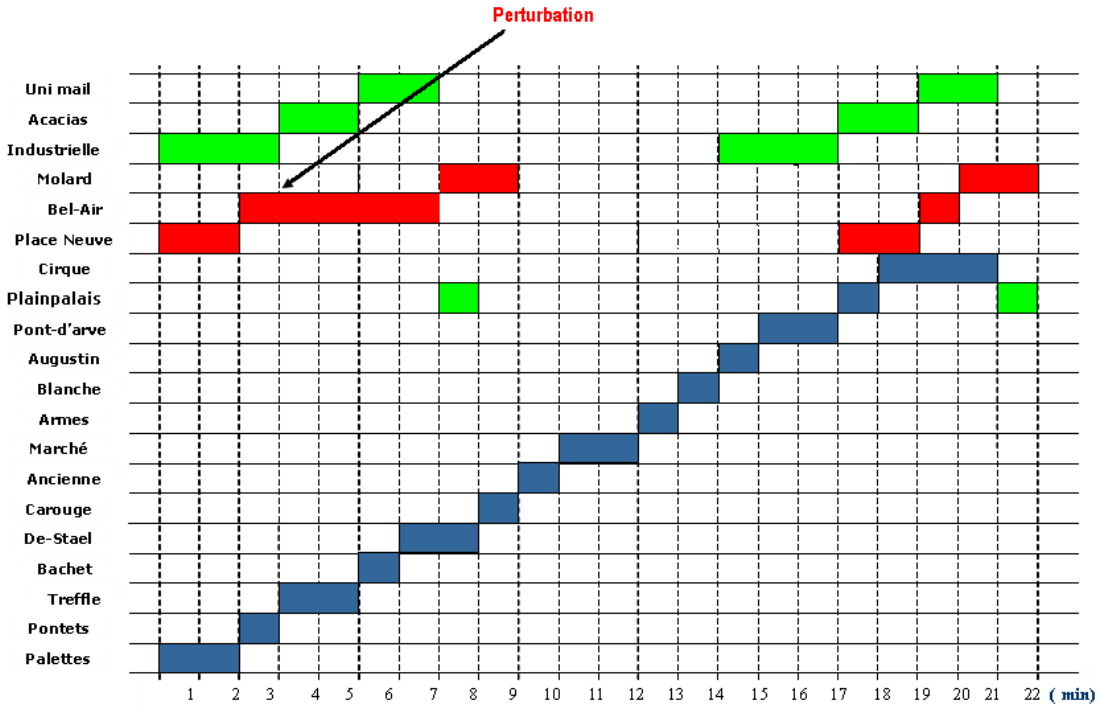


Fig 4.4.4b histogramme tables horaire lignes 12 ("*Pertubée*"), 13 et 15.
Positions :départ à 5h46

4.5 Simulation 3 : Transformation d'une maquette en réseau de Petri stochastique

Nous avons présenté, dans le chapitre 3, le modèle physique de réseau de transport urbain par trams. Ces réseaux sont caractérisés par des croisements qui, en fonction du débit de trams, influencent la *capacité dynamique* (**CapD**) du réseau. La **CapD** représente le nombre de trams encore en circulation sur le réseau après un certain temps. En effet, les trams marquent un temps de passage relativement élevé (par rapport à un trajet sans "obstacles") lors

des passages à des croisements. Par exemple, si l'on considère un tram à un arrêt de tram précédant un croisement, son temps d'arrêt est plus ou moins long en rapport avec la vitesse de déplacement du véhicule qui se trouve déjà dans le segment partagé du croisement concerné. Nous pouvons donner une estimation de ce temps, mais en réalité il dépend des évènements incertains. Les RdpS permettent de modéliser et de simuler ces évènements à caractères aléatoires, nous allons précisement attribuer aux transitions des temporisations. Une temporisation attribuée à une transition exponentielle représente la moyenne [HPsim tools] des temps de distribution (Exemple d'une fonction de loi de distribution exponentielle f(t)=$\lambda e^{-\lambda t}$, avec t = temporisation).

4.5.1 Gestion du flux de trams

Dans les réseaux de transports urbains, les trams partagent très souvent leurs voies avec d'autres types de véhicules. Tel est le cas de la ville de Genève. Il est extrèmement important de maintenir un écoulement de flux optimal pour garantir une bonne cohabitation des véhicules sur le réseau.

- Quelle est la fréquence d'insertion (**FreqI**) des trams dans le réseau ?
- Quelle est le flux maximum (**FluxMax**) de déplacement des trams sur le réseau ?
- Quelle est la vitesse moyenne commerciale (**Vmoy**) des véhicules sur le réseau ?
- Quelle est la capacité absolue (**CapA**) du réseau ?
- Quelle est la capacité dynamique (**CapD**) du réseau en régime continu ?
- Quel est le nombre de segments sur le réseau ?
- Combien de lignes de tram figurent sur un réseau donné ?

Les éléments de réponse à ces questions ont des liens que nous allons mettre en évidence avec un modèle de Rdp.

En effet
$\longmapsto$ la **CapA** est donnée par le nombre de segments sur le réseau.
$\longmapsto$ la **CapD** est donnée en fonction de la **FreqI** et du nombre total de trams dans les segments.
$\longmapsto$ la **Vmoy** est donnée par la moyenne des vitesses (**V**) tel que :

$$\mathbf{V} = \frac{distance \times Temps}{nombreDeTrams}$$

$\longmapsto$ le **FluxMax** est donné par le rapport du nombre de trams qui sont sortis du réseau et du Temps (le temps de simulation).

$$\mathbf{FluxMax} = \frac{nombreDeTramsSortants}{Temps}$$

Nous définissons un scénario de simulation selon lequel :

1. On produit des jetons (trams) dans le réseau à des fréquences bien précises. Cette opération va être réalisée en identifiant des *transitions deterministes* sous le logiciel HPsim, et on garde une trace du nombre de ces jetons dans des places (exemple nom de place = "*EntrantsP0*").

2. On consomme les jetons (trams) produits avec des *transitions immédiates*, et on garde une trace du nombre de ces jetons dans des places (exemple nom de place = "*SortantsP0*"). Ces jetons représentent les trams qui ont traversés tout le réseau et sont *arrivés à destination*. On considère qu'un tram est *arrivé à destination* lorsqu'il sort du réseau sujet à la simulation.

3. Pour chaque croisement du réseau, on identifie ses transitions auxquelles on attribue des temporisations exponentielles. Cette opération permet modéliser les évènements à caractères aléatoires qui peuvent subvenir dans cette partie du réseau.

4. On lance la simulation en continu dans un intervalle de temps prédéfini.

5. Enfin, on range soigneusement les résultats dans un tableau et on les analyse.

Il est important de définir les conditions d'arrêts d'une simulation. Ce paramètre doit être raisonnable, être suffisament indiquatif et contenir assez d'informations pour l'analyse des résultats.
Selon les objectifs initiaux, on pourrait citer quelques exemples de conditions d'arrêt :

- Le nombre de transitions tirées.
- Le nombre de jetons (trams) dans une place au temps t_i.
- Le temps d'arrêt de la simualtion.
- Le nombre de "pas" de simulation.
- etc...

Pour notre travail, nous avons choisi *le temps d'arrêt de la simualtion* comme condition d'arrêt de la simulation.

4.5.2 Les mesures de performance

Nous définissons deux mesures essentielles pour évaluer les performances de notre système :

1. Le flux maximum (**FluxMax**) de circulation. Cette mesure représente le nombre maximum de trams par milliseconde (trams/ms) qui sortent du réseau ("arrivent à destination"). i.e un tram toutes les 2×10^{-5} minutes

2. La saturation du réseau. On considère que chaque place, qui représente un segment de tram, contient 1 seul jeton (tram). Il y a saturation du réseau lorsque *Nombre-de-Jetons* > 1 (la valeur du nombre de jetons dans une ou plusieurs places est strictement supérieure à un).

Les paramètres pour le calcul de ces mesures de performances sont consignés dans un tableau tel que celui de la figure 4.5.2. dans lequel les variables aux différentes colonnes sont :

- Colonne 1 : "**nbLigne**" = le nombre de lignes qui figurent dans le réseau.

- Colonne 2 : "**tpsSimul**" = le temps d'arrêt de la simulation.

- Colonne 3 : "**frequence**" = la fréquence d'insertion des trams dans le réseau.

- Colonne 4 : "**nbEntrants**" = le nombre de trams qui ont été insérés dans le réseau.

- Colonne 5 : "**nbSortants**" = le nombre de trams qui sont sortis du réseau.

- Colonne 6 : "**nbCirculation**" = le nombre de trams encore en circulation sur le réseau aprés un certain temps : c'est la capacité dynamique (**CapD**)du réseau.

- Colonne 7 : "**nbAttente**" = le nombre de trams en attente d'insertion sur le réseau. Ce nombre indique une saturation du réseau.

Tableau des paramètres de mesures de performances						
nb-Ligne	**tps-Simul**	**frequence**	**nb-Entrants**	**nb-Sortants**	**nb-Circulation**	**nb-Attente**
5	10ms	1 tram/ms	50	18	32	9
//	//	1 tram/2ms	...	...	...	...
//	//	1 tram/3ms	...	...	...	...
//	//	1 tram/4ms	...	...	...	...
//	20ms	1 trams/ms	...	...	...	...
....	...	...	...	...	...	...

Fig 4.5.2. Modèle exemple du tableau des paramètres pour les mesures de performances

4.5.3 Simulation d'un croisement simple : [C/2,2]

Dans cette section on va considérer le cas d'un réseau simple composé *uniquement* d'un croisement à deux entrées et deux sorties **[C**/2, 2**]**. La figure 4.5.3 modélise ce réseau. On a rajouté quatre transitions ("*Production Tram 12*", "*Production Tram 13*", "*Consommation Tram 12*" et "*Consommation Tram 13*") et six places ("EntrantsP0", "EntrantsP1", "Buffer12", "Buffer13", "SortantsP2", "SortantsP3").

Les hypothèses de départ - simulation [C/2,2]

Pour la simulation, nous avons considéré certains paramètres initiaux pour notre Rdp (figure 4.5.3) :

1. La bulle bleue représente le croisement sujet de la simulation. On considère que l'information est incertaine aux arrêts de tram représentés par les transitions T0 et T1. On suppose que l'incertitude est due a des perturbations sur le réseau.

2. La condition d'arrêt de la simulation est définie par *le temps d'arrêt de la simulation* (T_s = 10ms, millisecondes). C'est l'intervalle de temps pendant lequel on observe l'évolution des jetons (trams) dans le réseau.

3. Nous produisons (bulle rose) des trams à des fréquences différentes, et nous simulons le modèle de Rdp pendant 10ms sous HPsim qui fournit en sortie le vecteur de marquages du réseau qui représentent les états du système.

4. Les places **P0**, **P3** représentent les segments de la ligne de tram 13.

5. Les places **P1**, **P2** représentent les segments de la ligne de tram 12.

6. Les transitions **T0** et **T1** sont des arrêts de trams. Ces deux transitions dites *exponentielles* auxquelles on associe une même temporisation (t = 1 ms). Cette temporisation représente la moyenne des temporisations d'une distribution de loi exponentielle.

7. Les transitions "***Production Tram 12***" et "***Production Tram 13***" modélisent la création de ressources dans le réseau. Ces deux transitions sont dites **déterministes**, i.e qu'on leur attribue une temporisation constante qui va définir la **fréquence** de tir de ces transitions. Par exemple si la temporisation d'une transition est égale à 1 ms, cela signifie qu'un tram arrive toutes les millisecondes.

8. Les places "***Buffer13***" et "***Buffer12***" sont des conteneurs de jetons à capacité illimitée, ils mettent en attente les trams dans le cas ou la fréquence de production est trop élevée. Ces places modélisent la *saturation* du réseau.

9. Les transitions "***Consommation Tram 12***" et "***Consommation Tram 13***" modélisent la consommation de ressources du réseau. Ces transitions sont *immédiates*, dès qu'un jeton arrive dans les places **P2** et **P3**, il est *consommé* suite au tir de ces transitions.

10. Les places "***EntrantsP0***" et "***EntrantsP1***" sont des compteurs à capacité illimitée qui gardent une trace du nombre total de trams entrants dans le réseau

11. Les places "***SortantsP2***" et "***SortantsP3***" ont également une capacité illimitée et indiquent le nombre de ressources (trams) qui ont traversé les arrêts T0 et T1 respectivement : Ce sont des compteurs qui gardent une trace nombre total de trams sortants du réseau.

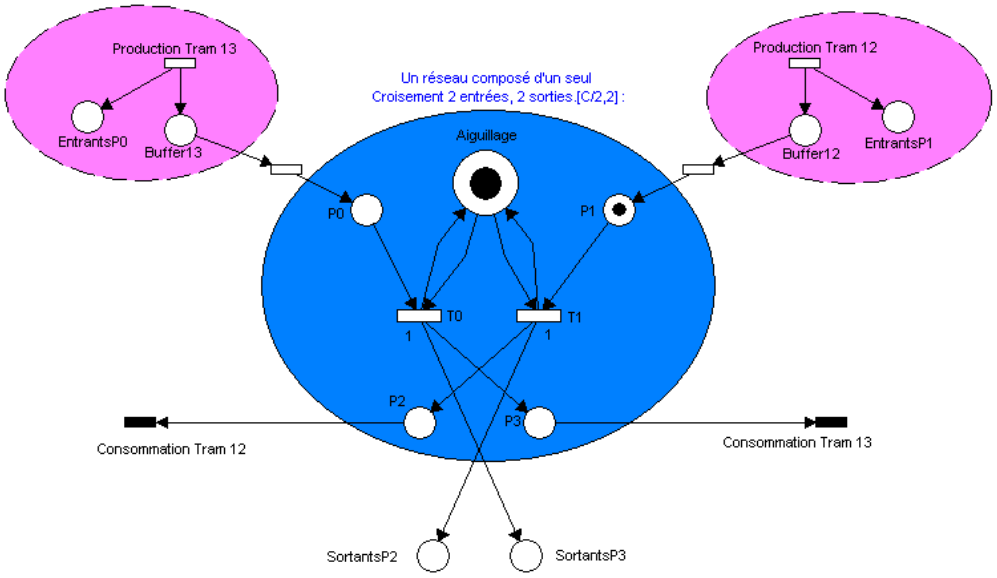


Fig 4.5.3 Rdp croisement 2 entrées, 2 sorties [C/2,2] : simulation des transitions stochastiques (T0 et T1).

Nous avons choisi quatre fréquences d'insertion différentes pour produire nos trams dans le réseau. Chacune de ces fréquences marque une étape de notre simulation selon la description suivante :

- **Etape 1 :** Nous associons aux transitions T0 et T1 une temporisation $\delta = 1$ms. La fréquence de trams est alors 1 tram/ 1ms. Le résultat après 10ms est fournit par le Rdp en haut à gauche de la figure 4.5.3a.

- **Etape 2 :** Nous associons aux transitions T0 et T1 une temporisation $\delta = 2$ms. La fréquence de trams est alors 1 tram/ 2ms. Le résultat après 10ms est fournit par le Rdp en haut à droite de la figure 4.5.3b.

- **Etape 3 :** Nous associons aux transitions T0 et T1 une temporisation $\delta = 3$ms. La fréquence de trams est alors 1 tram/ 3ms. Le résultat après 10ms est fournit par le Rdp en bas à gauche de la figure 4.5.3c.

- **Etape 4 :** Nous associons aux transitions T0 et T1 une temporisation $\delta = 4$ms. La fréquence de trams est alors 1 tram/ 4ms. Le résultat après 10ms est fournit par le Rdp en bas à droite de la figure 4.5.3d.

Analyse des résultats obtenus - simulation [C/2,2]

La simulation d'un croisement simple, modélisé par le Rdp de la figure 4.5.3 donne des résultats différents en fonction des fréquences d'insertion des trams dans le réseau. Les Rdp des figures 4.5.3.a, 4.5.3.b, 4.5.3.c et 4.5.3.d illustrent certains de ces résultats.

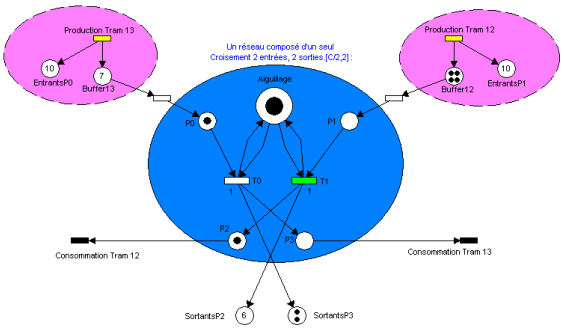


Fig 4.5.3.a : Rdp résultat après 10ms. (fréquence de tir de "Production Tram 12" et Production Tram 13" = 1 tram/ms).

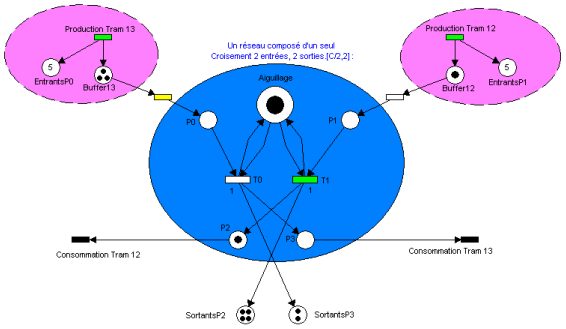


Fig 4.5.3.b : Rdp résultat après 10ms. (fréquence de tir de "Production Tram 12" et Production Tram 13" = 1 tram/2ms).

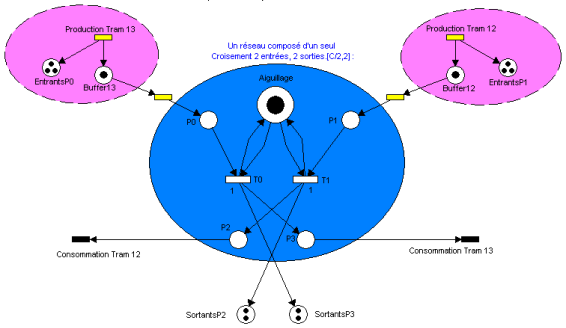


Fig 4.5.3.c : Rdp résultat après 10ms. (fréquence de tir de "Production Tram 12" et Production Tram 13" = 1 tram/3ms).

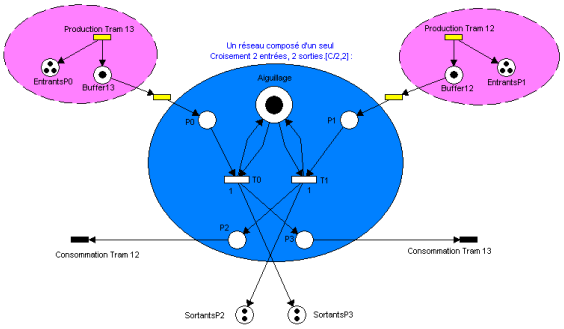
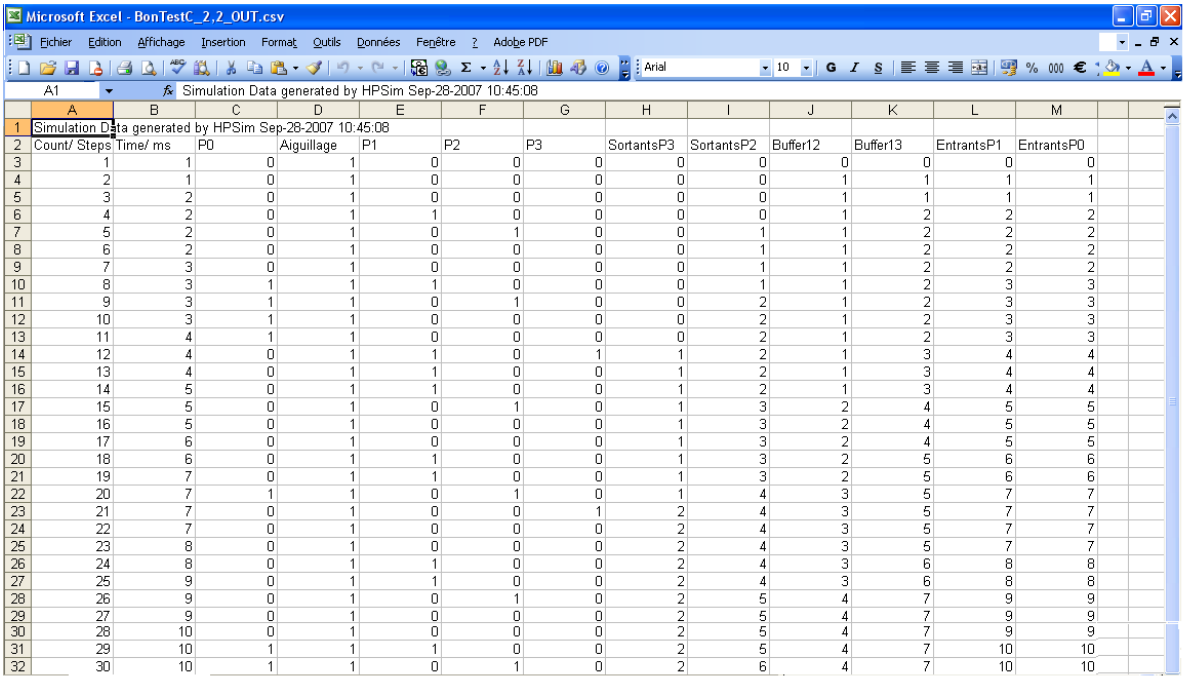


Fig 4.5.3.d : Rdp résultat après 10ms. (fréquence de tir de "Production Tram 12" et Production Tram 13" = 1 tram/4ms).

Le logiciel HPsim [HPsim tools] fournit en sortie un tableau excel (figure 4.5.3.e) qui représente, pour chaque pas de simulation les différents vecteurs de marquages des Rdp jusqu'à la fin de la simulation. La distribution des jetons dans les places est appelée le *marquage* du Rdp



Microsoft Excel - BonTestC_2,2_OUT.csv

Fichier Edition Affichage Insertion Format Outils Données Fenêtre ? Adobe PDF

A1 Simulation Data generated by HPSim Sep-28-2007 10:45:08

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Simulation Data generated by HPSim Sep-28-2007 10:45:08												
2	Count/ Steps	Time/ ms	P0	Aiguillage	P1	P2	P3	SortantsP3	SortantsP2	Buffer12	Buffer13	EntrantsP1	EntrantsP0
3	1	1	0	1	0	0	0	0	0	0	0	0	0
4	2	1	0	1	0	0	0	0	0	1	1	1	1
5	3	2	0	1	0	0	0	0	0	1	1	1	1
6	4	2	0	1	1	0	0	0	0	1	2	2	2
7	5	2	0	1	0	1	0	0	1	1	2	2	2
8	6	2	0	1	0	0	0	0	1	1	2	2	2
9	7	3	0	1	0	0	0	0	1	1	2	2	2
10	8	3	1	1	1	0	0	0	1	1	2	3	3
11	9	3	1	1	0	1	0	0	2	1	2	3	3
12	10	3	1	1	0	0	0	0	2	1	2	3	3
13	11	4	1	1	0	0	0	0	2	1	2	3	3
14	12	4	0	1	1	0	1	1	2	1	3	4	4
15	13	4	0	1	1	0	0	1	2	1	3	4	4
16	14	5	0	1	1	0	0	1	2	1	3	4	4
17	15	5	0	1	0	1	0	1	3	2	4	5	5
18	16	5	0	1	0	0	0	1	3	2	4	5	5
19	17	6	0	1	0	0	0	1	3	2	4	5	5
20	18	6	0	1	1	0	0	1	3	2	5	6	6
21	19	7	0	1	1	0	0	1	3	2	5	6	6
22	20	7	1	1	0	1	0	1	4	3	5	7	7
23	21	7	0	1	0	0	1	2	4	3	5	7	7
24	22	7	0	1	0	0	0	2	4	3	5	7	7
25	23	8	0	1	0	0	0	2	4	3	5	7	7
26	24	8	0	1	1	0	0	2	4	3	6	8	8
27	25	9	0	1	1	0	0	2	4	3	6	8	8
28	26	9	0	1	0	1	0	2	5	4	7	9	9
29	27	9	0	1	0	0	0	2	5	4	7	9	9
30	28	10	0	1	0	0	0	2	5	4	7	9	9
31	29	10	1	1	1	0	0	2	5	4	7	10	10
32	30	10	1	1	0	1	0	2	6	4	7	10	10

Fig 4.5.3.e : Output Fichier.csv, Résultats simulation du Rdp (figure 4.5.3)

Pour chacune des quatre fréquences d'insertion des trams dans le réseau, nous avons effectué plusieurs simulations et nous avons retenu la moyenne des valeurs des résultats obtenus. Le tableau de la figure 4.5.3.f regroupe ces résultats. Lorsque nous comparons les valeurs de ce tableau, nous remarquons que pour une fréquence de 1 tram/2ms on commence à voir une **saturation** du réseau. Cette saturation est représentée par l'accumulation des jetons dans les places "Buffer12" et "Buffer13" (voir figure 4.5.3). Par exemple, dans la troisième ligne du tableau (figure 4.5.3.f), lorsqu'on insère vingt trams dans le réseau à une fréquence de 1 tram/ms, après 10 ms, on observe qu'il y a 11 trams en "Attente d'insertion". Par contre dans la cinquième ligne de ce tableau, pour vingt trams insérés à une fréquence de 1 tram/3ms, on observe qu'il n'y a aucun tram en "Attente d'insertion".

On peut ainsi conclure que $1tram/3ms \leq fréquence\ idéale \leq 1tram/2ms$ (la fréquence idéale pour notre réseau est comprise entre 1 tram/3ms et 1 tram/2ms).

Tableau des paramètres de mesures de performances						
nb-Ligne	tps-Simul	frequence	nb-Entrants	nb-Sortants	nb-Circulation	nb-Attente
2	10ms	1 tram/ms	20	7	2	11
2	10ms	1 tram/2ms	10	6	2	2
2	10ms	1 tram/3ms	6	5	1	0
2	10ms	1 tram/4ms	4	4	0	0
2	20ms	1 tram/ms	40	17	1	22
2	20ms	1 tram/2ms	20	14	1	5
2	20ms	1 tram/3ms	14	13	0	1
2	20ms	1 tram/4ms	10	10	0	0
2	30ms	1 tram/ms	60	30	1	29
2	30ms	1 tram/2ms	30	22	1	7
2	30ms	1 tram/3ms	20	19	0	1
2	30ms	1 tram/4ms	16	14	1	1

Fig 4.5.3.f : Tableau mesures des performances pour un réseau simple composé d'un croisement de deux lignes de trams

Après avoir analysé la saturation du réseau, nous considérons une deuxième mesure des performances : Le flux maximum (**FluxMax**) de circulation dont la valeur est donnée par le rapport du nombre total de trams qui sortent du réseau ("*nbSortants*") et le temps de simulation ("*tpsSimul*") :

$$\textbf{FluxMax} = \frac{nbSortants}{tpsSimul}$$

Dans le tableau de la figure 4.5.3.f, lorsqu'on lit :

- Les lignes 2, 6 et 10, on remarque pour une fréquence d'insertion égale à 1 tram/ms, **FluxMax** = 1 tram/ms (ligne 6 du tableau).

- Les lignes 3, 7 et 11, on remarque pour une fréquence d'insertion égale à 1 tram/2ms, **FluxMax** = 0.73 tram/ms (ligne 11 du tableau).

- Les lignes 4, 8 et 12, on remarque pour une fréquence d'insertion égale à 1 tram/3ms, **FluxMax** = 0.65 tram/ms (ligne 8 du tableau).

- Les lignes 5, 9 et 13, on remarque pour une fréquence d'insertion égale à 1 tram/4ms, **FluxMax** = 0.5 tram/ms (ligne 9 du tableau).

Le flux maximum ne dépend pas de l'intervalle de temps pendant lequel on prend des mesures de performances (le temps de simulation) mais de la fréquence d'insertion des trams dans le réseau.

Frequence et Vitesse de déplacement

Le paramètre temporel est commnun aux mesures de la fréquence et de la vitesse de déplacement selon les formules :

- Vitesse = $\frac{distance}{Temps}$
- Fréquence = $\frac{NombreDeTrams}{Temps}$

Nous considérons la vitesse moyenne commerciale intiale qui est égale à 18km/h (5 $\times 10^{-3}$m /ms) sur le réseau Genevois [TPG 07]). Pour chaque fréquence, on calcule la vitesse correspondante (voir tableau de la figure 4.5.3g).

La courbe de la figure 4.5.3.h montre l'évolution de la vitesse moyenne en fonction de la fréquence.

On observe que lorsque la fréquence augmente, la vitesse diminue dans le réseau.

Frequence	**Vitesse**
1 tram/ms	**V** initiale = (5 $\times 10^{-3}$m /ms)
1 tram/2ms	(2.5 $\times 10^{-3}$m /ms)
1 tram/3ms	(1.6 $\times 10^{-3}$m /ms)
1 tram/4ms	(1.2 $\times 10^{-3}$m /ms)

Fig 4.5.3.g : Tableau frequence et vitesse de déplacement.

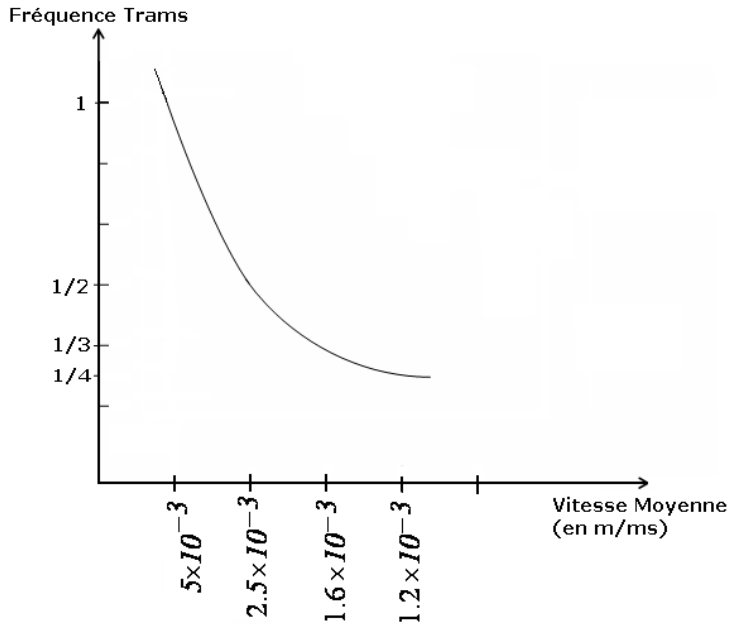


Fig 4.5.3.h : Courbe résultats frequence - vitesse moyenne.

4.5.4 Simulation de la maquette de Plainpalais

Cette simulation est la généralisation de la précédente, en effet, nous avons simulé un croisement simple à deux entrées et deux sorties. Nous allons maintenant analyser le comportement de la maquette de Plainpalais (figure 3.3.2) qui comporte plusieurs croisements.

Les hypothèses de départ - simulation *Plainpalais*

Le scénario de simulation de tout le réseau de Plainpalais est similaire à celui que nous avons effectué pour un croisement simple, sauf que pour ce réseau il existe des arrêts de trams avant et aprés certains croisements et pour représenter fidèlement la réalité nous nous sommes inspirés des tables horaires [TPG 07] pour approximer les durées de parcourt.

Ce scénario de simulation est défini comme suit :

On crée tout dabord des transitions pour *produire* et *consommer* des trams dans le réseau. On définit une fréquence de production, la consommation est immédiate. Pour chaque tram consommé, on crée des places compteurs qui comptabilisent leurs mouvements. On lance la simulation pendant

10ms et on analyse les résultats.

Nous illustrons les initialisations par le Rdp de la figure 4.5.4a dont voici le descriptif :

- La durée de simulation est toujours égale à 10 ms.

- Les transitions pour la production des trams : les transitions *déterministes* "**Prod-13**", "**Prod-17**", "**Prod-12**", "**Prod-15**", servent à produire dans le réseau les trams des lignes 13, 17, 12 et 15. Pour chaque étape, on attribue à ces transitions des fréquences de production notamment (1tram/ms, 1tram/2ms, 1tram/3ms, 1tram/4ms).

- Les transitions pour la consommation des trams : Ce sont des transitions *immédiates* "**Cons-17**", "**Cons-12 et 13**", "**Cons-13 et 15**", elles servent à consommer (sortir du réseau) les trams des lignes 17, 12, 13 et 15.

- Les transtions exponentielles ("**Te**") : On modélise les évènements aléatoires grâce aux transitions ("**Te**") qui se situent aux croisements. Ce sont des transitions exponentielles auxquelles nous avons associé une temporisation $\boldsymbol{t}$ = 1ms, cette valeur représente la variable temporelle d'une distribution [f(t)=$\lambda e^{-\lambda t}$] de loi exponentielle avec un temps égale à $\boldsymbol{t}$ comme moyenne, c'est le paramètre à fournir pour une transition exponentielle dans le logiciel HPsim [HPsim tools].

- Les transitions aux arrêts de trams : Ce sont les transitions "**Arrêt tram 12 (aller)**", "**Arrêt tram 12 (retour)**", "**Arrêt tram 13**", "**Arrêt tram 17**", "**Arrêt tram 13 et 15**". Ces transitions sont déterministes et la temporisation qu'on leur attribu est donnée par les tables horaires des TPG [TPG 07]. Il s'agit notamment des arrêts des trams 12, 13, 15 et 17 qui se situent au quartier *Plainpalais*.

- Toutes les places du réseau dont le label commence par la lettre "**p**" représentent les segments de trams.

- Les places "**Buffer**−" : Ces places représentent les jetons (trams) en attente d'insertion. Ce sont les places "**Buffer 13a**", "**Buffer 13b**", "**Buffer 12**", "**Buffer 17**", "**Buffer 15**".

- Les places compteurs "**Entrants**" : Ce sont les places "**Entrants 13a**",

"**Entrants 13b**", "**Entrants 17**", "**Entrants 15**" et "**Entrants 12**", elles servent de compteur pour comptabiliser le nombre de trams insérés dans le reseau.

- Les places compteurs "**Sortants**" : Ce sont les places "**Sortants 12**", "**Sortants 13**", "**Sortants 13 et 15**" et "**Sortants 17**", elles précèdent les transitions pour la consommation des trams et indiquent la quantité de ressources qui ont franchi cette partie du réseau.

- Les places **A1**, **A2**, **A3**, **A4**, **A5** et **A6** ne représentent aucun segment de place dans la réalité, elles existent uniquement dans le contexte des Rdp et servent d'aiguillage.

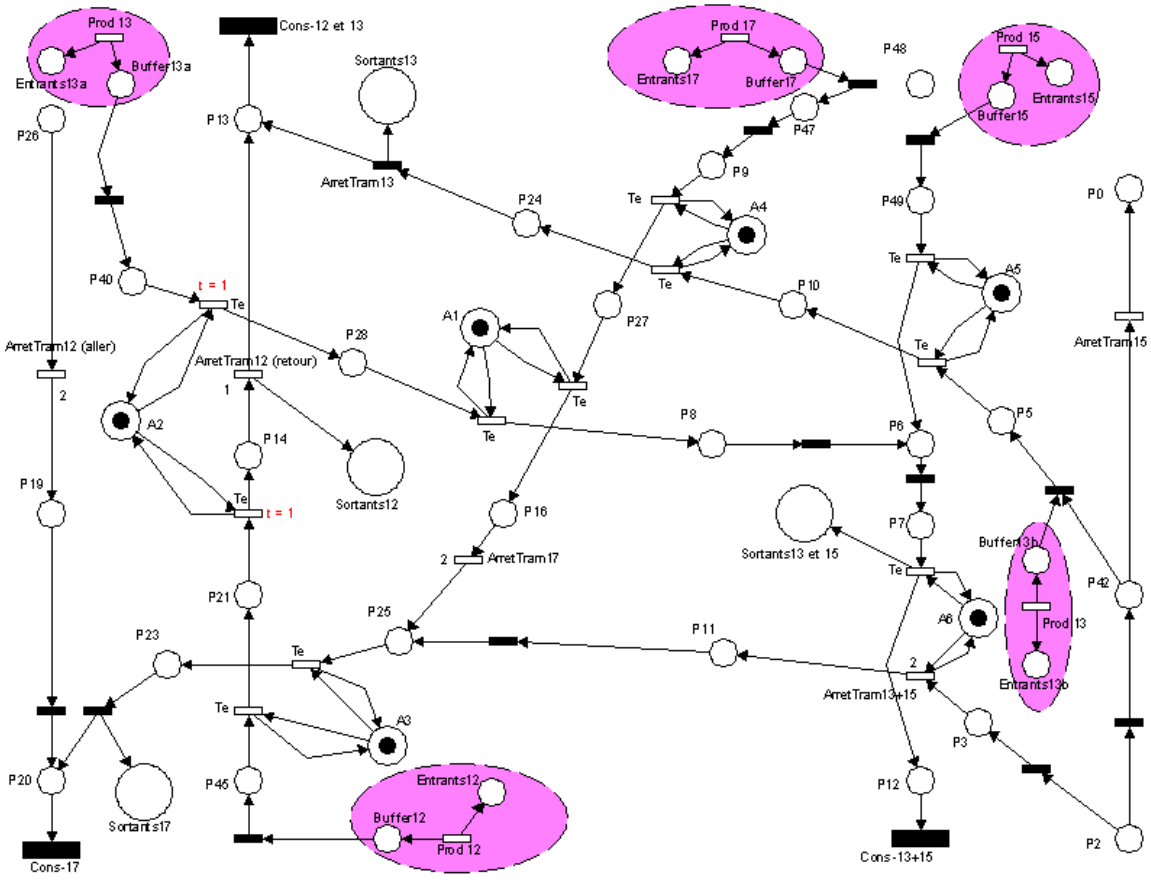


Fig 4.5.4a : Initialisations, simulation stochastique de plusieurs croisements, maquette de Plainpalais.

Analyse des résultats obtenus - simulation *Plainpalais*

Nous consignons les résultats des simulations dans un tableau (figure 4.5.4.b), tout comme nous l'avons fait, dans la section précédente, pour un réseau composé d'un seul croisement. Les résultats obtenus convergent sensiblement vers les mêmes conclusions, en effet, dans les lignes 4, 8, et 12 de ce tableau, le nombre ("**nbAttente**") de trams "en attente d'insertion" dans ce réseau est respectivement égale à 0, 1 et 2. On déduit que la fréquence de production idéale est égale à 1 tram /2ms. C'est la fréquence pour éviter une saturation du réseau.

Pour chaque fréquence de production, on obtient un flux maximum de trams. Par exemple pour une fréquence égale à 1 tram/4ms (lignes 6, 10 et 14 du tableau),

FluxMax = $\frac{nbSortants}{tpsSimul}$ = 1.3 trams /ms (Dernière ligne du tableau).

Tableau des paramètres de mesures de performances						
nb-Ligne	tps-Simul	frequence	nb-Entrants	nb-Sortants	nb-Circulation	nb-Attente
5	10ms	1 tram/ms	50	30	10	10
5	10ms	1 tram/2ms	25	18	7	0
5	10ms	1 tram/3ms	15	14	1	0
5	10ms	1 tram/4ms	10	8	2	0
5	20ms	1 tram/ms	100	64	12	24
5	20ms	1 tram/2ms	50	41	8	1
5	20ms	1 tram/3ms	35	29	6	0
5	20ms	1 tram/4ms	25	19	6	0
5	30ms	1 tram/ms	150	97	11	42
5	30ms	1 tram/2ms	75	67	6	2
5	30ms	1 tram/3ms	50	45	5	0
5	30ms	1 tram/4ms	40	36	4	0

Fig 4.5.4b : Tableau mesures des performances pour un réseau composé cinq lignes de trams

On observe les mouvements des trams de chaque ligne et l'évolution du réseau en régime continu. La figure 4.5.4c montre l'état du système après 10ms de simulation, pour une fréquence de production égale à 1 tram/ms. Ce réseau est numériquement représenté par la troisième ligne du tableau précédent (figure 4.5.4b). En effet, dans les colonnes du tableau, les paramètres **nbEntrants**, **nbSortants**, **nbAttente** et **nbCirculation** sont donnés par :

- "**nbEntrants**" = $\sum$ jetons "**Entrants**–" : [le nombre de jetons (trams) produits est égale à la somme des jetons dans les places "**Entrants 13a**", "**Entrants 13b**", "**Entrants 17**", "**Entrants 15**" et "**Entrants 12**"]

- "**nbSortants**" = $\sum$ jetons "**Sortants**–" : [le nombre de jetons (trams) consommé est égale à la somme des jetons dans les places "**Sortants 12**", "**Sortants 13**", "**Sortants 13 et 15**" et "**Sortants 17**"]

- "**nbAttente**" = $\sum$ jetons "**Buffer**–" : [le nombre de jetons (trams) en attente d'insertion est égale à la somme des jetons dans les places "**Buffer 13a**", "**Buffer 13b**", "**Buffer 12**", "**Buffer 17**", "**Buffer 15**"]

- "**nbCirculation**" = $\sum$ jetons "**P**–" : [le nombre de jetons (trams) en attente d'insertion est égale à la somme des jetons dans les places segments (les places dont le label commence par la lettre **P**)]

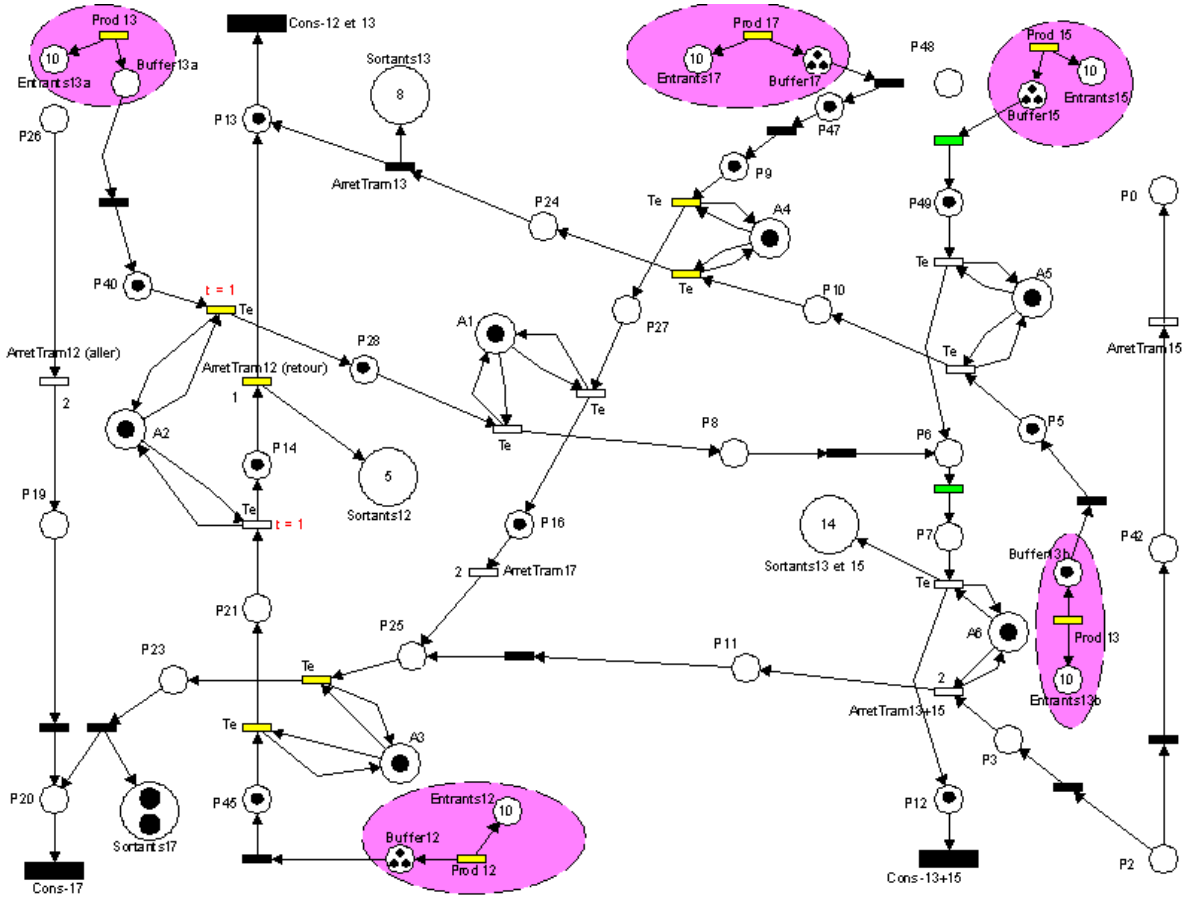


Fig 4.5.4c : Résultat, simulation stochastique de plusieurs croisements , fréquence de production 1tram/ms, (maquette de Plainpalais).

Chapitre 5

Programme JAVA : Analyseur-de-performances

La quantité importante des valeurs obtenues après simulation rend parfois la tâche d'analyse fastidieuse. Pour une meilleure compréhension des résultats obtenues, nous avons écrit en langage Java un parseur sémantique ("***Analyseur-de-performances***"). Ce parseur est destiné à récupérer les informations contenues dans des fichiers au format "*.csv*" (**C**omma **S**eparated **V**alues) et de leur donner un sens en terme de performances de réseau. Les fichiers analysés sont fournis par le logiciel HPsim pour chaque simulation et ont été au préalable rangés dans un répertoire. Chaque répertoire contient les fichiers résultats de simulation pour réseau composé d'un nombre de lignes précis. Les simulations que nous avons effectuées dans la section 4.5 du chapitre 4 concernent deux principaux réseaux : un réseau composé de *deux* lignes de trams et un réseau composé de *cinq* lignes de trams, on dispose ainsi de deux principaux répertoires de fichiers.

le programme Java "*Analyseur-de-performances*" permet aussi de visualiser, via des graphes, ces mesures de performances. Pour cela, nous avons utiliser ***JCharts***. L'**API** (Application Programming Interface) ***JCharts*** [Jcharts] propose un ensemble de classes pour la génération de graphiques. Elle propose aussi tous les outils nécessaires pour intégrer ces graphiques via Swing, servlets/JSP, etc...

5.1 Algorithme du programme

1. On parcourt un répertoire contenant les fichiers (.csv) issus de la simulation. Pour chaque fichier, on le traite (la classe TraiteFichierTexte.java).

2. On parcourt simultanément la première et la dernière ligne du fichier. Pour chaque élément (**Elt**) de ligne séparé par un point virgule, on compare le premier caractère ("***car***").

   - Si ***car***(**Elt**) = "**P**", le nombre de trams encore en circulation dans le réseau (**nbCircule**) est égale à la somme de toutes les valeurs de la dernière ligne du fichier qui sont dans les colonnes correspondant aux éléments de la première ligne commencant par le caractère "**P**".

   - Si ***car***(**Elt**) = "**B**", le nombre de trams en attente d'insertion dans le réseau (**nbAttente**) est égale à la somme de toutes les valeurs de la dernière ligne du fichier qui sont dans les colonnes correspondant aux éléments de la première ligne commencant par le caractère "**B**".

   - Si ***car***(**Elt**) = "**E**", le nombre de tram insérés dans le réseau (**nbEntrants**) est égale à la somme de toutes les valeurs de la dernière ligne du fichier qui sont dans les colonnes correspondant aux éléments de la première ligne commencant par le caractère "**E**".

   - Si ***car***(**Elt**) = "**S**", le nombre de tram qui sortent du réseau le réseau (**nbSortants**) est égale à la somme de toutes les valeurs de la dernière ligne du fichier qui sont dans les colonnes correspondant aux éléments de la première ligne commencant par le caractère "**S**".

3. Dans un tableau à six colonnes (figure 5.1.a), on regroupe respectivement des valeurs :

   - 1ère colonne : le nombre de lignes du réseau (**nbLignes**) qui est égale au nombre d'éléments qui commencent par le caractère "**E**". Comme exemple, on peut citer les éléments "**EntrantsP0**" et "**EntrantsP1**" du fichier de la figure 4.5.3.e.
   - 2ième colonne : le temps de simulation ("**tempsSimulation**") qui est tout simplement le second élément de la première ligne du fichier (figure 4.5.3.e).
   - 3ième colonne : **nbEntrants**.

- 4ième colonne : **nbEntrants**.
- 5ième colonne : **nbCirculation**.
- 6ième colonne : **nbAttente**.

4. Les données du tableau de la figure 5.1.a sont les paramètres pour générer des graphiques.

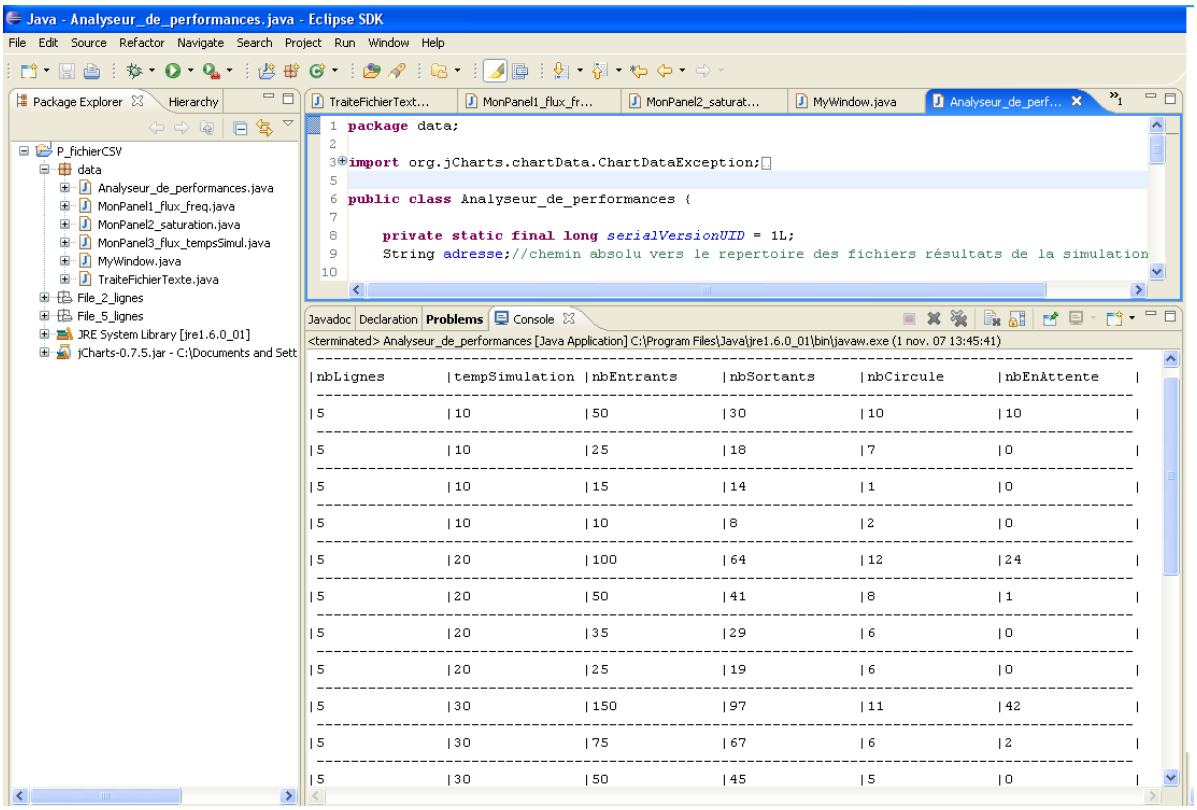


Fig 5.1.a : Exécution du programme "*Analyseur-de-performances*", Screen Shot Ecclipse, tableau à six colonnes.

Les graphiques des figures 4.5.b et 4.5.c illustrent respectivement les mesures de performances du réseau composé de *deux* lignes de trams et du réseau composé de *cinq* lignes de trams.

- Panel 1 : On représente le flux maximum de trams à travers le réseau en fonction de la fréquence d'insertion.

- Panel 2 : on représente le flux maximum à travers de réseau en fonction du temps de simulation.

– Panel 3 : Sur l'axe des ordonnées, on a le nombre de trams en attente d'insertion (**nbEnAttente**) et sur l'axe des abcisses, on a les fréquences d'insertion. Ce grapique permet de visualiser la fréquence ideale pour éviter une saturation du réseau (accumulation des jetons (trams) dans une place).

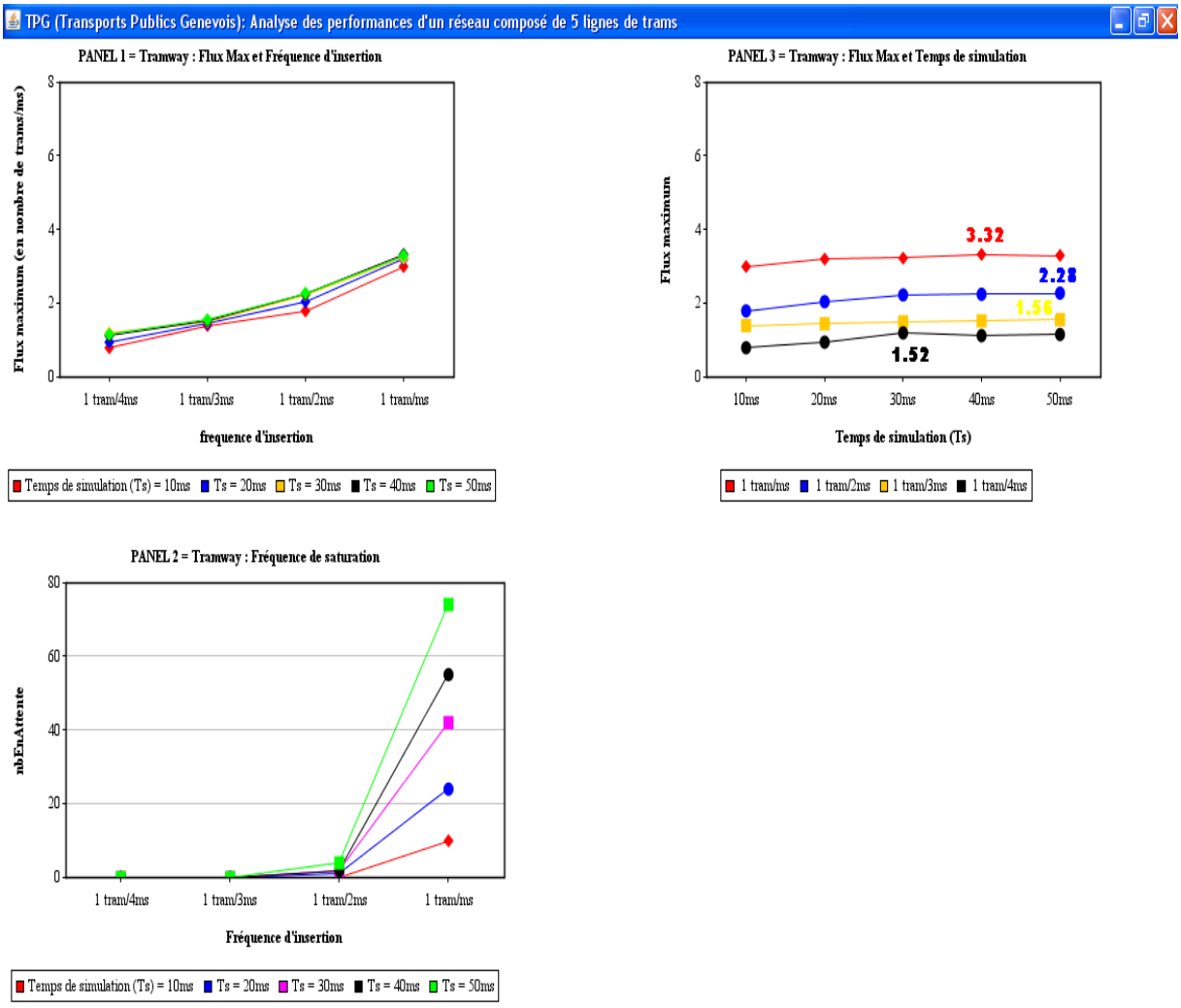


Fig 5.1.b : Exécution du programme "*Analyseur-de-performances*", Réseau composé de 5 lignes de trams, Mesures de performances.

Description de la figure 5.1.b

L'observation du panel 3 indique, pour chaque fréquence, les différents flux maximum. On observe qu'après avoir simuler le réseau pendant 10, 20, 30, 40 et 50 ms, **FluxMax** = 3.32 tram/ms pour une fréquence d'insertion égale à 1 tram/ms. Et pour une fréquence d'insertion égale à 1 tram/4ms, **FluxMax** = 1.2 tram/ms.

L'observation du panel 2 indique que la fréquence d'insertion idéale se situe entre 1 tram/3ms et 1 tram/2ms. En effet, lorsque la fréquence d'insertion supérieure ou égale à 1 tram/2ms, le nombre de trams en attente augmente considérablement, ce qui représente une saturation du réseau.

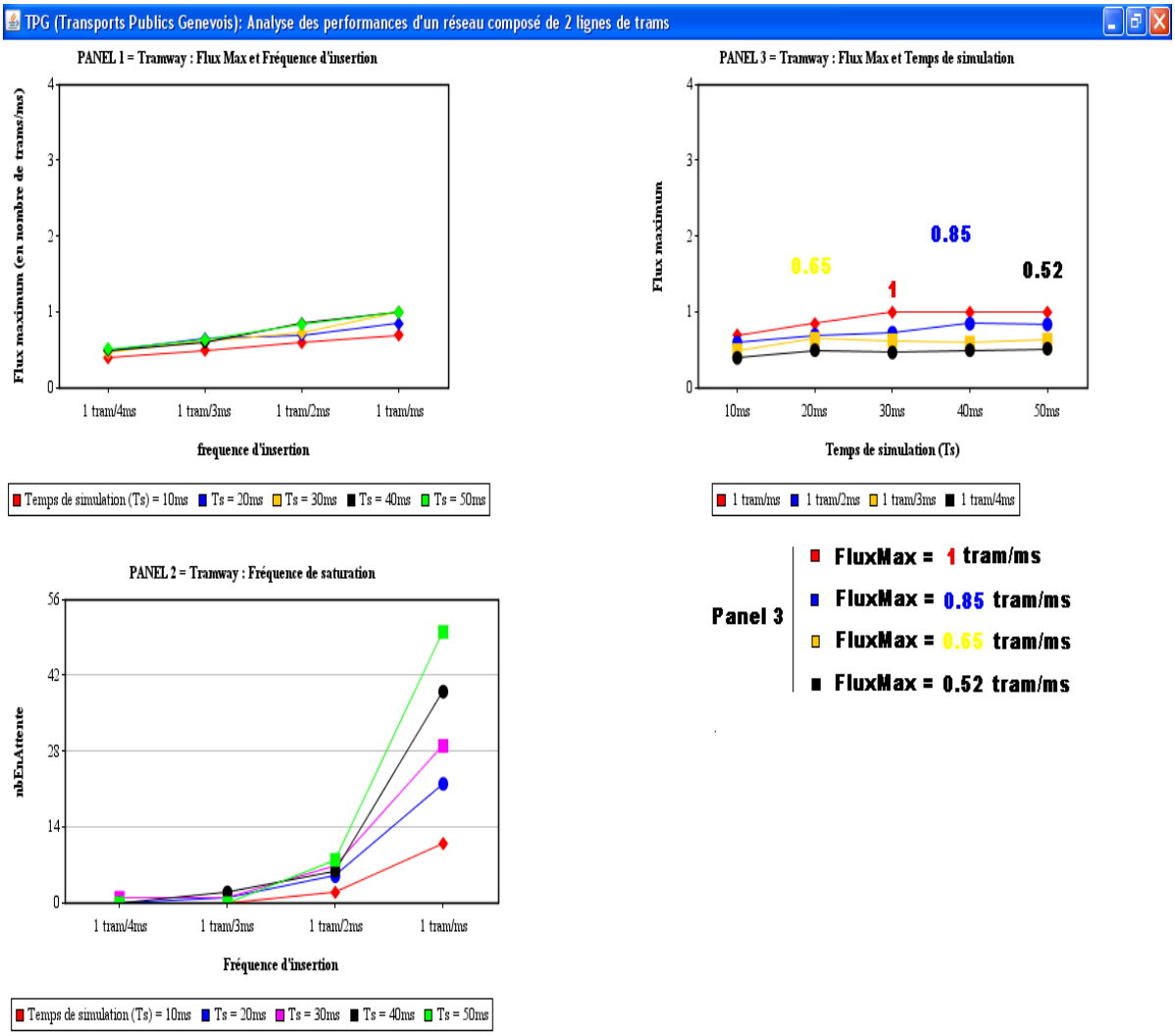


Fig 5.1.c : Exécution du programme "*Analyseur-de-performances*", Réseau composé de 2 lignes de trams, Mesures de performances.

5.1.1 listing code création panel 1 : Flux maximum et fréquence d'insertion

```
package data;

import org.jCharts.axisChart.*; import org.jCharts.types.*; import
org.jCharts.chartData.*; import org.jCharts.properties.*;

import java.awt.*; //BasicStroke, Color, Shape, Paint, Stroke
import java.util.Vector;

import javax.swing.*;


public class MonPanel1_flux_freq extends JPanel {

    private static final long serialVersionUID = 1L;

    AxisChart axisChart;

    //Constructeur :
    public MonPanel1_flux_freq() {

        super();
        setBackground(Color.white);
    }
//  ************************************************************************
    public void construireUnAxisChart(String adresse) throws
                                  ChartDataException, PropertyException {

        //-----------------------------------------------------------------
        String[] xAxisLabels = {" 1 tram/4ms", "1 tram/3ms",
                                " 1 tram/2ms", " 1 tram/ms"};
        String xAxisTitle= "frequence d'insertion";
        String yAxisTitle= "Flux maximum (en nombre de trams/ms)";


        /*double[][] data = new double[][]{ {0.4, 0.5, 0.6, 0.7},
                                          {0.5, 0.65, 0.7, 0.85},
                                          {0.46, 0.63, 0.73, 1.0} }; */
```

```
//%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

        //données calculées de chaque flux (reseau de 2 lignes)
        String title = "PANEL 1 = Tramway : Flux Max et Fréquence d'insertion";
        DataSeries dataSeries =
            new DataSeries( xAxisLabels,xAxisTitle,yAxisTitle,title );
        TraiteFichierTexte tf = new TraiteFichierTexte();
        Vector<Double> vip = tf.getVectFlux(adresse);
        Double [][] temp = tf.genererMatrice(vip);
        Double [][] tempInverse = tf.inverseElementsLignesMatrice(temp);

        final double[][] data =
            new double[tempInverse.length][tempInverse[0].length];

        for (int i = 0; i < tempInverse.length; i++) {
            for(int j = 0; j < tempInverse[0].length; j++) {
                data[i][j] = tempInverse[i][j];
            }
        }

        String[] legendLabels = { "Temps de simulation (Ts) = 10ms",
                "Ts = 20ms",
                "Ts = 30ms",
                "Ts = 40ms",
                "Ts = 50ms"};

        //generateur aleatoire de 3 couleurs
        //Paint[] paints = TestDataGenerator.getRandomPaints(3);
        Paint[] paints = {Color.red, Color.blue, Color.orange,
                          Color.black, Color.green};
        Stroke[] strokes = new Stroke[ 5 ];
        /*strokes[ 0 ]= new BasicStroke( 3.5f, BasicStroke.CAP_ROUND,
        BasicStroke.JOIN_ROUND, 5f, new float[]{ 5f, 5f, 10f, 5f}, 4f );*/
        strokes[ 0 ]= new BasicStroke( 1.0f );
        strokes[ 1 ]= new BasicStroke( 1.0f );
        strokes[ 2 ]= new BasicStroke( 1.0f );
        strokes[ 3 ]= new BasicStroke( 1.0f );
        strokes[ 4 ]= new BasicStroke( 1.0f );
        Shape[] shapes= {
                PointChartProperties.SHAPE_DIAMOND,
```

```
        PointChartProperties.SHAPE_DIAMOND,
        PointChartProperties.SHAPE_DIAMOND,
        PointChartProperties.SHAPE_DIAMOND,
        PointChartProperties.SHAPE_DIAMOND};

;

LineChartProperties lineChartProperties =
        new LineChartProperties( strokes, shapes );

AxisChartDataSet axisChartDataSet= new AxisChartDataSet(
        data,
        legendLabels,
        paints,
        //ChartType.BAR,
        ChartType.LINE,
        lineChartProperties );

dataSeries.addIAxisPlotDataSet( axisChartDataSet );

ChartProperties chartProperties = new ChartProperties();
AxisProperties axisProperties = new AxisProperties();
LegendProperties legendProperties= new LegendProperties();

axisChart = new AxisChart( dataSeries,
        chartProperties,
        axisProperties,
        legendProperties,
        450,
        350 );
Graphics g = this.getGraphics();
Graphics2D g2 = (Graphics2D) g;

axisChart.setGraphics2D(g2) ;
//Pour afficher ou non les "labels" de l'axe des abcisses
//axisProperties.getXAxisProperties().setShowAxisLabels(false);
//...de l'axe des ordonnées
//axisProperties.getYAxisProperties().setShowAxisLabels(false);
```

```
}

    //*****************************************************************
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        Graphics2D g2 = (Graphics2D) g;

        axisChart.setGraphics2D( g2) ;
        try {
            axisChart.render();
        }
        catch (ChartDataException cde) {
            System.out.println( cde.toString() );
        }
        catch (PropertyException pe) {
            System.out.println( pe.toString() );
        }
    }
    //*****************************************************************

}//end class
```

Chapitre 6

Modélisation UML

Les démarches de modélisation varient selon les praticiens. Parmis les langages graphiques qui existent, il y a le langage UML dont l'utilisation est assez générale. A travers la panoplie des modèles de ce langage, on représente une abstraction supplémentaire de la realité. Dans notre contexte, la réalité n'est autre que le réseau de trams proprement dit. Pour ce projet, nous nous limiterons au *diagramme de classes* pour les objets métiers.

Quelques définitions [O'REI 06] :

- Une ***classe*** décrit un ensemble d'objets similaires du point de vue :
  - de leur structure (les attributs).
  - de leur comportement (les opérations).
  - de leurs relations à l'extérieur(les associations).

- Le ***diagramme de classes*** représente la structure des données de classes et relations entre ces classes.

- Un ***objet*** est une instance d'une classe, il possède les comportements de la classe à laquelle il appartient.

Afin de construire toute la structure des données de notre réseau de transports, il est important d'identifier au préalable les entités à représenter. Pour le réseau de trams, on aimerait représenter :

- Le *réseau de tram* (au sens physique !).
- Un segment.
- Une ligne de tram.
- Une connection de segment.

– Un arrêt de tram.
– Un croisement.

6.1 Le diagramme de classes

Les classes définissent une abstraction, un type abstrait qui permettra plus tard d'instancier des objets. Le diagramme qui compose ces classes décrit le modèle général et permet de séparer les composants du système, ce qui peut s'avérer efficace pour structurer un travail de développement. Le modèle de la figure 5.2 est composé des classes suivantes :

1. La classe "**TramWayNet**" modélise un réseau de trams dans son aspect général. On considère que le réseau est composé d'une liste de trams, des lignes de trams et des segments.

2. La classe "**Connector**" modélise la liaison entre segment, on utilise le concept d'héritage pour définir deux sortes de liens : les arrêts de trams et les croisements tels que nous les avons définis au chapitre 3, (section 3.2.1).

3. La classe "**Segment**". A un segment, on affecte :
   – zéro ou deux "*connector*" (lien de segments). Pour un arrêt de tram par exemple, il ne peut y avoir que deux segments de tram connectés au maximum. Un segment peut être composé de plusieurs petits segments(liste de segments) et joue le rôle de début ou de fin de lien .

4. La classe "**Crossing**", c'est un croisement avec comme attributs la liste des segments en entrée et en sortie.

5. La classe "**TramStop**" modélise les arrêts de trams. Ses attributs sont le segment en entrée et le segment en sortie.

6. La classe "**TramLine**" est utilisée pour représenter une ligne de tram à laquelle on associe plusieurs arrêts de trams. Les arrêts sont les éléments d'une liste ordonnée, ce qui nous donne la direction de la ligne.

6.2 Les Reseaux de Petri *vs* le language UML

Du point de vue structurel, la modélisation du réseau de trams par les Rdp et le langage UML présente des similitudes. En effet, nous avons eu besoins d'identifier les entités du monde réel qui étaient à représenter :

- Des **segments**.
- Des **trams**.
- Des **croisements**.

Cependant, le modèle de Rdp proposé (figure 3.3.2), concernait *uniquement des trams* . Si l'on considère tout le réseau de transports urbains, qui comporte d'autres types de véhicules, on peut imaginer que le Rdp équivalent serait moins "lisible" en terme de croisements.

Ainsi, après avoir consideré ces deux techniques de modélisation, il apparaît clairement que l'approche UML est particulièrement bien adapté pour décrire les aspects statiques ou structurels d'un système. Il existe aussi des modèles UML pour montrer le fonctionnement d'un système : les *diagrammes d'état*. Cependant, les réseaux de Petri, de part leur fonctionnement asynchrone, sont bien adaptés pour modéliser des systèmes à caractère distribué.

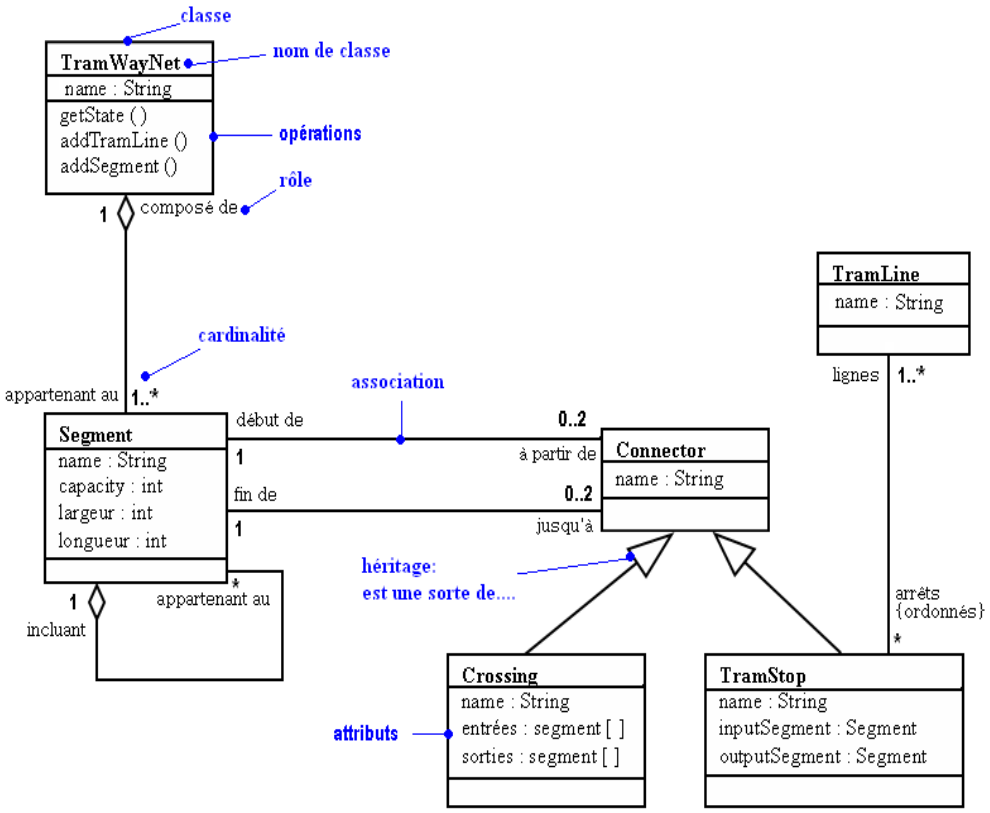


Fig 5.2 Diagramme de classes, Réseau de trams.

Chapitre 7

Conclusion et Perspectives

Dans ce mémoire, nous avons présenté un modèle du réseau de transport public urbain de la Ville de Genève. L'utilisation du formalisme des réseaux de Petri pour modéliser ce système nous a permis d'avoir une représentation graphique, de faire des analyses en temps continu (dynamiquement). Nous avons défini une syntaxe et une sémantique pour la transformation de tout réseau, au sens physique, en réseau de Petri.

Le logiciel HPsim s'est révélé être assez complet et pratique pour simuler les différents modèles, nous avons pu exploiter des informations utiles fournies par son analyse.

Ce travail a été réalisé en considérant les trams d'un réseau de traffic urbain. Nous avons pu vérifier le fonctionnement des horaires des trams....Mais que pouvons nous dire du réseau dans son ensemble qui comporte des Autobus et des Trolley-bus ? Avec ces deux nouveaux paramètres, des contraintes supplémentaires s'ajoutent à la modélisation. On peut généraliser le système d'aide à la décision basé sur les Rdp à des réseaux de transport en commun multimodaux : Tramway, Autobus, Trolley-bus.

La modélisation du réseau de trams avec le language UML présente des similitudes avec la modélisation par les Rdp, et pour rendre cette étude complète, on pourrait implémenter et expérimenter les modèles proposés.

Cette expérience nouvelle et unique m'a permis d'utiliser et d'appliquer, à un système temps réel, l'un des outils informatiques qui nous a été enseigné pendant ces deux dernières années. Et ces dernières lignes marquent la *fin* de plusieurs semaines de travail.....ou plutôt le *début* !

Bibliographie

[BUC 04] Didier BUCHS, Pascal RACLOZ, Réseaux de Petri, 1995-2004.

[UML 00] Grady BOOCH, James RUMBAUGH et Ivar JACOBSON, Le guide de l'utilisateur UML-2000, ISBN 2212091036, 2000

[TPG 07] URL http ://www.tpg.ch/

[Jcharts] URL http ://jcharts.sourceforge.net/

[Petri, 62] Carl ADAM PETRI, "Communication avec des Automates", Thèse de doctorat, Université de Darmstadt, BONN, 1962

[HPsim tools] URL http ://www.winpesim.de

[KOW 06] Yvan KOWALSKI, "Aide à la décision par l'analyse sémantique et la simulation des interactions dans l'organicube, modèle qualitatif général d'audit pour les entreprises", Thèse de Doctorat, Université de FRIBOURG (SUISSE), 31 janvier 2006.

[RechOP] Robert FAURE, "RECHERCHE OPERATIONNELLE et PHENOMENES ALEATOIRES...", Dunod, Sciences SUP, ISBN : 2100076760

[BON 01] Patrice BONHOMME, "Reseaux de Petri P-Temporels : Contributions à la commande robuste", Thèse de doctorat, Université de SAVOIE, 12 Juillet 2001.

[DUSS 03] "Contribution à l'aide à la décision en situation de crise", URL http ://anales.org/ri/2003/ri-mai-2003/dusserre45-52.pdf, mai 2003.

[CHA 06] Thomas CHATAIN, "Dépliage symboliques de réseaux de Petri de haut niveau et application à la supervision des systèmes répartis", Thèse de doctorat, Université de RENNES 1, 23 novembre 2006.

[FLA] Flavien BALBO, "Modélisation d'une perturbation sur un réseau de transport : le modèle incident", Rapport pdf.

[O'REI 06] Dan Pilone, Neil Pitman, "UML 2 en concentré", O'REILLY, Janvier 2006.

[GLog 06] Philippe DUGERDIL, Cours de Genie Logiciel, CUI, Université de Genève, Dècembre 2006.

[SUP 06] Annie Choquet-GENIET, "Les réseaux de Petri : Un outil de modélisation Cours et exercices corrigés", DUNOD, 9 mars 2006.

[LEF 05] Mario LEFEBVRE, "Processus Stochastiques Appliqués", Presses Internationales Polytechniques, Août 2005.

[GAL 97] Laurent GALLON, "Le modèles Réseaux de Petri Stochastiques : Extensions et Applications", Thèse de Doctorat, Université Paul SABATIER de TOULOUSE, 16 Décembre 1997.

[BOU] Julien BOURGEOI, "Modélisation de réseaux", Cours de l'université de Franche-Comté, Rapport pdf.

[MENU] Jacques MENU, "Systèmes Informatiques", CUI (Centre Universitaire Informatique), Université de Genève.