N°d'ordre : 09/2014- M / INF

UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE HOUARI BOUMEDIENE (USTHB)

FACULTE D'ELECTRONIQUE ET D'INFORMATIQUE

MÉMOIRE

Présenté pour l'obtention du diplôme de :

MAGISTER

Filière : Informatique

Option : Informatique Mobile

Par :

Mr. OUKAS Nourredine

Sujet :

MODÉLISATION ET ÉVALUATION DE PERFORMANCES DES WSNS AVEC CAPTEURS NON-FIABLES

Soutenu publiquement, le 13/04/2014, devant le jury composé de :

Mme M.BOUKALA-IOUALALEN	**Professeur à l'USTHB**	**Présidente**
Mme N. GHARBI	**MCA à l'USTHB**	**Directrice de mémoire**
Mr Y.ZAFOUNE	**MCA à l'USTHB**	**Examinateur**
Mr M.A.RIAHLA	**MCB à l'UMBB**	**Examinateur-Invité**

Table des matières

Table des figures

Introduction générale

Les progrès réalisés lors de ces dernières décennies dans les domaines de la microélectronique, de la micromécanique et des technologies de communication sans fil, ont permis de produire avec un coût raisonnable des composants de quelques millimètres cubes de volume. Ces derniers, appelés capteurs, intègrent une unité de captage chargée de capter des grandeurs physiques (chaleur, humidité, vibrations) et de les transformer en grandeurs numériques, une unité de traitement informatique et de stockage de données et un module de transmission sans fil. Un grand nombre de ces dispositifs (capteurs) sont déployés dans la nature afin de créer un réseau de capteurs à des fins aussi bien de contrôle que de monitoring. Le fort potentiel d'applications des réseaux de capteurs en font un domaine de recherche très actif.

Depuis plusieurs années, les réseaux de capteurs provoquent un intérêt croissant au sein des communautés scientifiques et industrielles. Concrètement, un réseau de capteurs sans fil est composé d'un ensemble de capteurs intelligents miniaturisés ayant une capacité de communiquer entre eux. Ces réseaux trouvent de nombreuses applications d'activités militaires et civiles dans divers domaines, tels que la surveillance de l'environnement, l'habitat intelligent, la télémédecine, le suivi de la population et la gestion des catastrophes. Ces réseaux sans infrastructure fixe, peuvent être déployés de façon rapide, dans des zones sensibles et/ou difficilement accessibles. Leur but est le plus souvent de surveiller une zone et de prendre régulièrement des mesures par certains nœuds capables de relayer l'information à grande échelle. Ainsi, ils offrent aux utilisateurs de plus en plus de services.

En pratique, certains composants des capteurs (batterie par exemple) pourraient être sujets à des pannes aléatoires. Il est donc d'une importance basique de prendre en considération la non-fiabilité de ces composants lors de la modélisation des réseaux de capteurs.

Cependant, la multitude des services offerts laisse poser une problématique, qui est celle de la complexité d'analyse et d'évaluation des performances de ces réseaux. Cette complexité s'accentue avec l'évolution des nouvelles technologies entraînant une difficulté dans leur conception.

En effet, avant la mise en place d'un réseau de capteurs, son déploiement nécessite une phase de test afin de s'assurer du bon fonctionnement de tous ses composants soft ou hard. Pour ce faire, la simulation reste la solution la plus utilisée en pratique pour faire des évaluations sur des réseaux qui peuvent être à large échelle et ce pour réduire le maximum d'erreurs de conception possibles.

Actuellement, une multitude de simulateurs de réseaux sans fil sont déployés dans le domaine informatique, tels que GloMoSim, NS (Network Simulator), NS2, NS3 et OMNET++, par exemple.

Vus les inconvénients de la simulation, la conception et la planification de ces réseaux de capteurs sans fil, nécessitent des modèles mathématiques formels précis permettant leur modélisation et l'évaluation de leurs performances en se basant sur des résultats exacts. Ainsi, nous nous intéressons, dans ce projet, à l'utilisation des modèles formels permettant la modélisation et l'évaluation des performances des réseaux de capteurs sans fil. Pour cela, nous considérons particulièrement les réseaux de Petri stochastiques généralisés.

Si on reprend un peu l'histoire de la modélisation et de l'analyse des systèmes, on constate d'une manière générale, que l'on a eu des modèles strictement qualitatifs (systèmes de transitions, réseaux de pétri et des modèles strictement quantitatifs (files d'attente, chaînes de Markov, ...). Les études menées avec ces différents types de modèles sont des études disjointes, et on doit se poser la question relative à l'étude d'un système, du lien entre ce qui est évalué et ce qui est vérifié.

Les réseaux de Petri stochastiques et stochastiques généralisés (qui sont des extensions temporelles et stochastiques des réseaux de Petri) sont précisément des modèles qui entrent dans ce cadre et dont l'objectif est de permettre à la fois une analyse qualitative et quantitative des systèmes. Ils ont une puissance de formulation mathématique et probabiliste permettant d'évaluer les performances des systèmes distribués et parallèles.

L'utilisation des réseaux de Petri stochastiques généralisés dans le domaine des réseaux de capteurs sans fil serait intéressante pour une modélisation des différents aspects et une évaluation des paramètres de performances pertinents. Pour cela, nous allons modéliser le trafic de communication entre un capteur et ses voisins avec la prise en compte de certains phénomènes tels que : les appels répétés et la limitation du buffer.

En réalité, les capteurs peuvent tomber en panne à cause des défauts de construction industrielle, de l'influence de l'hostilité de l'environnement ou bien de l'expiration de la batterie du capteur. Par conséquent, il est indispensable de prendre en compte la non-fiabilité des capteurs lors de la construction du modèle.

En plus, les capteurs communiquent entre eux à travers un lien sans fil qui consomme de l'énergie. Un capteur appartient à un réseau et utilise des techniques de conservation d'énergie. Parmi les techniques les plus usuelles, nous citons le changement d'état du capteur i.e. que le capteur se met en veille pendant une période de temps ensuite il s'active pendant une autre période de temps. Par conséquent, le passage inter-mode a une influence sur la consommation d'énergie et sur le temps de

réponse .i.e. sur la durée de vie du capteur et la rapidité du réseau. Donc, négliger ces facteurs, implique une mauvaise modélisation et une évaluation non pertinente.

L'objectif de ce projet est de modéliser le trafic de communication entre un capteur et ses voisins à l'aide des réseaux de pétri stochastiques généralisés afin d'évaluer les performances d'un réseau de capteurs en prenant en compte la non-fiabilité, la taille du buffer, le phénomène de rappel et le changement d'état des capteurs.

Pour ce faire, nous commençons le travail par une introduction suivi par quatre sections et nous achevons le manuscrit par une conclusion.

Dans le premier chapitre, nous présentons d'une manière générale les réseaux de capteurs sans fil, leurs caractéristiques et leurs applications. Aussi, nous présentons un état de l'art de notre recherche qui porte sur les travaux utilisant les modèles formels dans les réseaux de capteurs. Ensuite, dans le deuxième chapitre, nous présentons le modèle de « réseaux de Petri stochastiques généralisés ». Dans le troisième chapitre, nous utilisons le modèle de réseaux de Petri stochastiques généralisés pour modéliser le trafic de communication dans un réseau de capteurs sans fil afin de produire des formules mathématiques et probabilistes pour le calcul des indices de performance. Ensuite, nous discutons de la fiabilité des capteurs, les demandes répétitives d'envoyer un message à un nœud voisin, la taille du buffer et nous donnons les formules de paramètres de performance pour chaque modèle. Dans le quatrième chapitre, nous donnons une brève présentation du logiciel ***TimeNet 4.0*** que nous utilisons pour faire l'évaluation numérique des modèles proposés dans le troisième chapitre et présentons des tableaux et des graphes afin de comparer les modèles proposés. Le présent travail s'achève par une conclusion générale et des perspectives de recherche.

Chapitre 1 : Les réseaux de capteurs sans fil

Introduction

Les progrès réalisés durant ces dernières décennies dans les domaines de la microélectronique, de la micromécanique et des technologies de communication sans fil ont permis de produire à un coût raisonnable des composants de quelques millimètres cubes de volume. De ce fait, un nouveau domaine de recherche s'est créé pour offrir des solutions économiquement intéressantes et pratiques pour la surveillance à distance et le traitement des données dans des environnements complexes et distribués ; à savoir les réseaux de capteurs sans fil et leurs applications.

Les réseaux de capteurs ont de nombreuses perspectives d'application dans des domaines très variés : applications militaires, surveillance industrielle ou surveillance des phénomènes naturels, etc. De ce fait, les micro-capteurs sont de véritables systèmes embarqués. Le déploiement de plusieurs d'entre eux, en vue de collecter et transmettre des données environnementales vers un ou plusieurs points de collecte, d'une manière autonome, forme un réseau de capteurs.

Dans le présent chapitre, nous détaillons les principaux concepts liés aux réseaux de capteurs. En premier lieu, nous définissons l'élément de base dans les RCS (Réseau de Capteur Sans fil) qui est le capteur sans fil et nous détaillons son architecture interne. Puis, nous donnons une description des RCS en présentant les caractéristiques, les paramètres entrant dans leur conception et leurs différentes applications possibles.

1.1. Définition et architecture d'un nœud capteur

1.1.1. Définition d'un capteur

D'une façon générale, un capteur est un dispositif qui répond aux stimulus physiques (tels que la chaleur, la lumière, le bruit, la pression, le magnétisme, etc.) et il convertit la quantité ou le paramètre de stimulus physique en des signaux enregistrables (tels que des signaux électriques, des signaux mécaniques, etc.). Ces signaux sont numérisés pour produire des données de détection. Les capteurs représentent une interface entre le monde physique et le monde des dispositifs électriques, tels que les ordinateurs. Par conséquent, ils facilitent aux personnes de comprendre, de surveiller et de commander des machines et des environnements à distance.

Les capteurs diffèrent par leurs types et leurs formes. Ils peuvent mesurer presque toutes sortes de stimulus physiques. Par convention, les capteurs peuvent être classifiés dans différents types selon le

phénomène à mesurer [1]; on recense des capteurs qui mesurent des quantités mécaniques, fluides, électriques, thermiques, chimiques, biologiques, etc.

En plus de la fonctionnalité essentielle d'un capteur qui est la détection et la mesure d'un phénomène précis, le nœud capteur sans fil (appelé aussi couramment "capteur"), quand à lui, inclut également d'autres unités pour traiter et fournir des données de détection.

1.1.2. Architecture générale d'un capteur sans fil

Un capteur sans fil est un dispositif physique, de taille miniature, qui a pour but essentiel d'accomplir trois tâches complémentaires : le relevé d'une grandeur physique, le traitement éventuel de cette information et la communication avec d'autres capteurs afin de transmettre cette information. Pour pouvoir accomplir ces tâches, le capteur est composé principalement de quatre unités de base : l'unité de captage, l'unité de traitement, l'unité de transmission et l'unité de contrôle d'énergie [2]. Selon le domaine d'application, il peut aussi contenir des modules supplémentaires tels qu'un système de localisation (GPS) ou un système générateur d'énergie (cellules photovoltaïques). Quelques micro-capteurs, plus volumineux, sont dotés d'un système mobilisateur chargé de les déplacer en cas de nécessité.

Nous donnons ci-dessous les rôles des unités de base d'un nœud capteur : [9]

- **L'unité de captage :** généralement composée de deux sous-unités : le récepteur et le transducteur (convertissant le signal du récepteur en signal électrique). Le capteur fournit des signaux analogiques, basés sur le phénomène observé, au convertisseur Analogique/Numérique. Ce dernier transforme ces signaux en un signal numérique compréhensible par l'unité de traitement.
- **L'unité de traitement :** elle comprend un processeur généralement associé à une petite unité de stockage. Elle fonctionne à l'aide d'un système d'exploitation spécialement conçu pour les micro-capteurs. Elle exécute les protocoles de communications qui permettent de faire collaborer le nœud avec les autres nœuds du réseau. Elle peut aussi analyser les données captées pour alléger la tâche du nœud collecteur.
- **L'unité de transmission :** elle effectue toutes les émissions et réceptions des données dans un milieu sans-fil. Elle peut être de type optique (comme dans les nœuds *Smart Dust* [3]), ou de type radiofréquence.
    - Les communications de type optique sont robustes vis-à-vis des interférences électriques. Néanmoins, ne pouvant pas établir des liaisons à travers des obstacles,

elles présentent l'inconvénient d'exiger une ligne de vue permanente entre les entités communicantes.

- Les unités de transmission de type radiofréquence comprennent des circuits de modulation, démodulation, filtrage et multiplexage ; ceci implique une augmentation de la complexité et du coût de production du micro-capteur.

Concevoir des unités de transmission de type radiofréquence avec une faible consommation d'énergie est un défi, car pour qu'un nœud ait une portée de communication suffisamment grande, il est nécessaire d'utiliser un signal assez puissant et donc une énergie consommée importante. L'alternative consistant à utiliser de longues antennes n'est pas possible à cause de la taille réduite des micro-capteurs.

- **Unité de contrôle d'énergie :** l'unité de contrôle d'énergie est un système essentiel. Elle doit répartir l'énergie disponible aux autres modules et unités, de manière optimale (par exemple en réduisant les dépenses inutiles et en mettant en veille les composants inactifs). Cette unité peut aussi gérer des systèmes de rechargement d'énergie à partir de l'environnement via des cellules photovoltaïques par exemple.

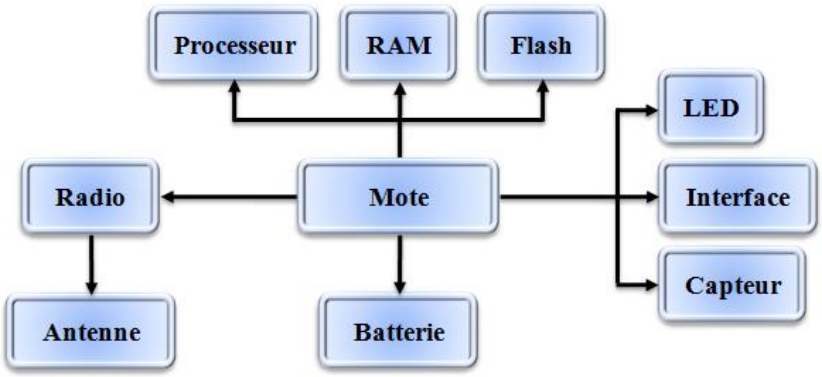


Figure 1.1. Architecture générale d'un nœud capteur sans fil.

Généralement, les capteurs ont la même structure matérielle, qui est représentée dans la Figure 1.1.

- **Mote, Processeur, RAM, Flash :** c'est la carte physique, composée de processeur, RAM et Flash. L'ensemble assure les traitements binaires et le stockage. La taille d'un programme qui s'exécute sur le capteur est limitée par la taille de la RAM.

- **Radio et antenne :** pour pouvoir communiquer avec l'ensemble du réseau, un capteur est équipé d'un radio émetteur/récepteur et d'une antenne.
- **LED, Interface, Capteur :** le but principal d'un capteur est de capter. Il est équipé de plusieurs types de détecteurs des différents phénomènes à observer, de plusieurs LED qui indiquent l'état de certains composants selon leur valeur (on/off) et d'interfaces de communication (Serial, RG45, …) ou d'autres interfaces d'extension.
- **Batterie :** le capteur est équipé d'une batterie qui assure son alimentation autonome. Certains capteurs sont dotés d'un panneau solaire permettant le rechargement continu de cette batterie.

Le nœud capteur sans fil est mis en application par un panneau de capteur qui intègre tous les composants mentionnés ci-dessus et d'autres circuits nécessaires. Actuellement, beaucoup de nœuds capteurs ont été fabriqués dans le cadre de la recherche et du développement des réseaux de capteurs sans fil. La Figure 1.2 présente un nœud capteur sans fil MICAZ fabriqué par Crossbow Corporation [4]. La couche supérieure du nœud est le panneau de capteur MPR2400TM [5], de 58mm×32mm×7mm de taille et 18g de poids. Le microcontrôleur est basé sur l'Atmel ATmega128L et trois types de mémoire, à savoir, mémoire flash de programme 128KB, mémoire de configuration EEPROM 4KB et une mémoire de mesure 512KB.

Figure 1.2. Exemple d'un capteur sans fil MICAZ de Crossbow.

Le panneau de capteur peut supporter plusieurs types de capteurs, y compris des acoustiques, lumineux, la magnéto compteur et le capteur d'accéléromètre. Il comporte aussi un convertisseur analogique-numérique (ADC) de 10 bits. Un nœud MICAZ fonctionne avec une fréquence de 2.4 GHz et un débit qui va jusqu'à 250 Kbps, conformément à la norme IEEE 802.15.4. Le fond de capteur est une caisse de batterie qui peut inclure deux batteries de type AA. La consommation électrique du microcontrôleur en mode actif est de 8mA et en mode veille est moins de 15µA. La consommation de l'émetteur-récepteur est de 19.7mA en mode de réception, 17.4mA en mode transmission et 20µA en mode inactivité (avec le régulateur de tension allumé) et de 1µA en mode veille (avec le régulateur de tension éteint).

1.2. Les réseaux de capteurs sans fil

1.2.1. Description

Les réseaux de capteurs forment une sous-classe des réseaux mobiles ad-hoc. Les nœuds capteurs, déployés en grand nombre, sont capables de récolter et de transmettre des données environnementales d'une manière autonome. Les positions des nœuds ne sont pas obligatoirement prédéterminées. Les capteurs sont dispersés aléatoirement dans une zone géographique, appelée champ de capture, qui définit le terrain d'intérêt pour le phénomène capturé. Cela implique que les protocoles réseau utilisés doivent posséder des capacités d'auto-organisation. Une autre caractéristique des réseaux de capteurs est la coopération des capteurs. Les données capturées sont acheminées grâce à un routage multi-sauts vers un nœud considéré comme "point de collecte", appelé nœud collecteur (**S**tation de **B**ase ou *Sink*). Ce dernier peut être connecté à l'utilisateur du réseau via un satellite ou Internet. L'usager peut ainsi adresser des requêtes aux nœuds du réseau en précisant le type de données requises, puis les données environnementales capturées seront récoltées par le collecteur. Les capteurs sont équipés d'un processeur embarqué. Au lieu d'envoyer des données brutes aux nœuds responsables de la fusion, ils utilisent leur capacité de traitement pour effectuer localement des traitements et transmettre les données nécessaires partiellement traitées [2].

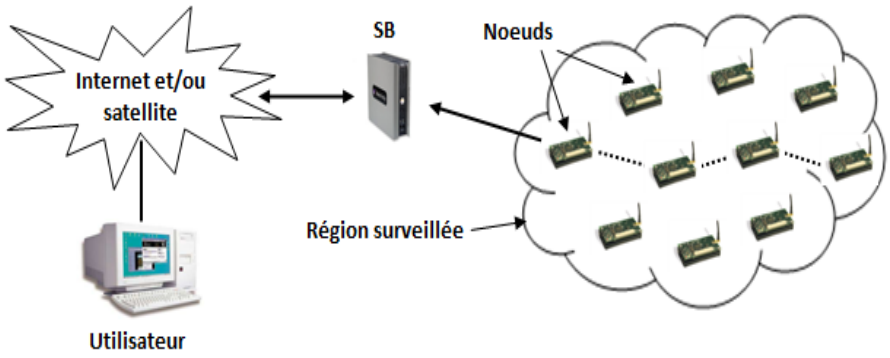


Figure 1.3. Schéma général d'un réseau de capteurs.

Chaque application a ses propres contraintes. Dans tous les domaines, le rôle d'un réseau de capteurs est cependant le même. Les nœuds doivent surveiller certains phénomènes grâce à leurs capteurs, puis envoient les informations capturées à un collecteur. Le collecteur est un nœud particulier doté généralement d'une puissance de calcul supérieure et d'une quantité d'énergie potentiellement infinie. Ce collecteur peut être connecté à Internet ou possède un lien radio de type GSM ou GPRS qui lui permet d'envoyer les informations (données ou alertes) à un centre de contrôle pour l'utilisateur final (Figure

1.3). Il peut y avoir plusieurs collecteurs mobiles ou fixes dans un réseau mais pour des raisons de coût, il y a beaucoup moins de collecteurs que de nœuds ordinaires.

1.2.2. Types de réseaux de capteurs

Les réseaux de capteurs peuvent être classés selon différents critères. Un schéma de classification possible est celui de la mobilité des nœuds. Ainsi, les réseaux de capteurs sans fil peuvent être de deux types selon [6] :

- **Réseaux de capteurs mobiles :** le réseau est constitué d'un ensemble de capteurs mobiles évoluant dans un environnement statique. Le but de tels réseaux est souvent l'exploration de zones inaccessibles ou dangereuses. Nous trouvons souvent des travaux de recherche orientés vers la robotique où les nœuds jouent à la fois le rôle de capteur et d'actionneur. Un actionneur étant un capteur particulier doté de caractéristiques importantes en puissance d'énergie et de calcul. Ce type de nœud a la capacité de prise de décision en fonction des données récoltées.
- **Réseaux de capteurs fixes (stationnaires) :** le réseau est constitué de capteurs fixes servant à la surveillance d'occurrence d'évènements dans une zone géographique. Dans ce cas, le réseau n'effectue que la surveillance. Les données mesurées sont transmises en mode multi-sauts au nœud collecteur. Ce dernier est chargé après réception de l'information, de mettre en œuvre les actions nécessaires. Ce collecteur peut être connecté, de manière filaire à un autre réseau. Ce type de réseaux de capteurs est le plus répandu et le plus traité dans les travaux de recherche.

Selon la nature de la station de base, les réseaux de capteurs peuvent être aussi répartis en deux classes :

- **Réseaux de capteurs avec station de base fixe :** les positions des collecteurs dans le réseau de capteurs sont connues à l'avance par tous les nœuds du réseau.
- **Réseaux de capteurs avec station de base mobile :** les collecteurs peuvent être statiques ou mobiles, ils peuvent se déplacer vers des positions déterminées pour interroger le réseau et réaliser une collecte optimale des données capturées.

1.2.3. Caractéristiques et paramètres des réseaux de capteurs sans fil

La conception des réseaux de capteurs est influencée par de nombreux paramètres qui ont fait l'objet de plusieurs études. Toutefois, aucune de ces études n'a pu intégrer tous les paramètres qui sont à l'origine de la conception des réseaux de capteurs. Ces paramètres sont importants car ils constituent un

guide pour la conception de ce type de réseaux. Ils peuvent être également utilisés pour comparer et analyser les performances du réseau.

Nous définissons ci-après ces paramètres [2] :

- **Tolérance aux fautes :** l'échec des capteurs dû aux différentes raisons telles que l'épuisement d'énergie, dommages physiques ou la présence d'obstacles ne devrait pas affecter la tâche globale du réseau. La tolérance aux fautes est la capacité de maintenir les fonctionnalités du réseau sans aucune interruption due à la défaillance de quelques capteurs. Le niveau de tolérance aux fautes devrait être adapté en fonction de l'hostilité du milieu dans lequel est déployé le réseau.
- **Passage à l'échelle :** le nombre de capteurs dans un réseau dépend de la nature du phénomène observé. Dans certains cas, il peut atteindre l'échelle des milliers voir même des millions de capteurs. Les réseaux de capteurs doivent être en mesure de supporter ce très grand nombre de capteurs. Ainsi, les protocoles et les algorithmes développés doivent fonctionner correctement malgré ces dimensions.
- **Les contraintes matérielles :** comme nous l'avons détaillé plus haut, le capteur est composé de plusieurs unités de base en plus d'autres modules supplémentaires. Tous ces composants consomment de l'énergie. La capacité de la batterie d'alimentation est donc la contrainte la plus forte. D'autres contraintes s'imposent sur la taille et le volume du capteur. Tous ces composants doivent être regroupés dans un volume qui ne dépasse pas quelques centimètres cube, et un poids assez léger. Par ailleurs, les transmissions radios sont plus complexes et consomment plus d'énergie que les liaisons optiques.
- **La topologie du réseau :** la topologie d'un réseau de capteurs est en perpétuel changement à cause de la défaillance des capteurs, de l'épuisement d'énergie des capteurs, des obstacles qui se présentent dans le réseau ou du besoin d'ajouter d'autres capteurs. Ces raisons engendrent une nécessité de gestion des changements et une maintenance de la topologie. On distingue trois phases [6]:
    - ***Le pré-déploiement :*** les nœuds capteurs peuvent être déployés d'une manière **aléatoire** (largués en masse d'un avion par exemple). Ils peuvent aussi être placés un par un, soit par un humain ou un robot dans des positions prédéterminées, on parle alors d'un déploiement déterministe.
    - ***Le post-déploiement :*** après le déploiement, les changements de topologie sont dus aux changements des états des capteurs : la position, l'accessibilité (à cause de brouillage, obstacles mobiles), disponibilité de l'énergie, dysfonctionnement, etc.

- ***Redéploiement des nœuds supplémentaires :*** d'autres nœuds supplémentaires peuvent être redéployés à tout moment pour remédier aux dysfonctionnements des nœuds.

- **Environnement :** un capteur peut être déployé tout près de l'objet à surveiller ou carrément à l'intérieur de l'objet. Ceci peut être l'intérieur d'une grosse machine, le fond d'un océan, un lieu contaminé biologiquement ou chimiquement, dans un champ de bataille au-delà des lignes ennemies, dans une maison ou un immeuble, etc. Ces situations très variées engendrent de très fortes contraintes sur l'environnement de déploiement des capteurs.
- **Support de transmission :** il est clair que dans ces réseaux, la communication se fait via des liens sans fil. Ces liens peuvent être formés par radio, infrarouge, ou un support optique. La majorité des capteurs utilisent la radiofréquence comme support de transmission. La transmission via infrarouge ou support optique est résistante aux interférences provoquées par les appareils électriques, mais elle est écartée car elle exige une vue directe entre l'émetteur et le récepteur.
- **La consommation en énergie :** la seule source d'énergie dans un capteur est la batterie qui fournit une quantité d'énergie très limitée. Généralement, l'énergie emmagasinée n'est pas renouvelable. Les transmissions de données entre nœuds demandent beaucoup d'énergie, surtout, si des nœuds tombent en panne ce qui exige la réorganisation du réseau. C'est pourquoi, des algorithmes et des protocoles ont été spécialement développés avec comme principale contrainte l'économie d'énergie, parfois même, au détriment de la qualité du service fourni par le réseau.

1.2.4. Architectures adoptées pour les réseaux de capteurs

Les architectures dans les réseaux de capteurs dépendent des applications et des techniques utilisées pour faire acheminer l'information des capteurs à la station de base.

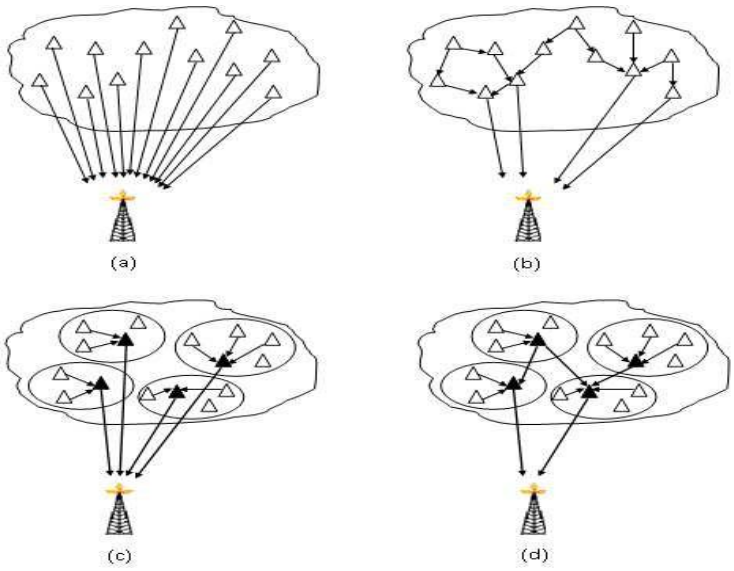


Figure 1.4. Architecture de communication de données dans un réseau de capteurs

Le processus d'acheminement de l'information des capteurs à la station de base peut prendre quatre formes. Dans les architectures plates, les capteurs peuvent communiquer directement avec la station de base en utilisant une forte puissance (Figure 1.4 (a)), ou via un mode multi-sauts avec des puissances très faibles (Figure 1.4 (b)), alors que dans les architectures hiérarchisées, le nœud représentant le groupe (cluster), appelé cluster-head, transmet directement les données à la station de base (Figure 1.4 (c)), ou via un mode multi-saut entre les cluster-heads (Figure 1.4 (d)).

1.3. Domaines d'application des réseaux de capteurs

Un réseau de capteurs peut être composé de différents types de capteurs : sismiques, magnétiques, thermiques, visuels, infrarouges, acoustiques, et beaucoup d'autres types qui sont capables de surveiller une large variété de conditions ambiantes telle que : la température, l'humidité, le mouvement, la pression, la présence de certains objets, etc. Les capteurs sont caractérisés essentiellement par leur coût très réduit et leur souplesse de support de communication utilisé : le sans fil. Ces avantages ont permis aux réseaux de capteurs d'envahir plusieurs domaines d'applications. Ils permettent aussi d'étendre les applications existantes et de faciliter la conception d'autres systèmes tels que le contrôle et l'automatisation des chaînes de montage. Les réseaux de capteurs ont le potentiel de révolutionner la manière même de comprendre et de construire les systèmes physiques complexes. Les réseaux de

capteurs peuvent se révéler très utiles dans de nombreuses applications, lorsqu'il s'agit de collecter et de traiter des informations provenant de l'environnement.

Parmi les domaines d'applications de ces réseaux de capteurs, nous citons : militaire, environnemental, sanitaire, domestique, commercial, etc. Voici quelques exemples détaillés d'applications potentielles dans ces différents domaines [2].

1.3.1. Applications militaires

Les réseaux de capteurs sans fil peuvent être intégrés dans le commandement militaire, le contrôle, les communications, le renseignement, la surveillance et les systèmes de reconnaissance et détection des cibles [2]. Le déploiement rapide, l'auto-organisation et la tolérance aux fautes, caractéristiques des réseaux de capteurs, en font une technique de détection très prometteuse pour les militaires. Bien que les réseaux de capteurs soient basés sur le déploiement dense des capteurs, caractérisés par leur faible coût, la destruction de certains nœuds n'affecte pas une opération militaire autant que la destruction d'un capteur traditionnel, ce qui dote les réseaux de capteurs d'une grande utilité dans le champ de bataille. Une partie des applications militaires des réseaux de capteurs sont : la surveillance des forces amies, des équipements et des munitions, la surveillance du champ de bataille ; la reconnaissance des forces ennemies et le terrain, la détection de cible, l'évaluation des dommages de combat, et la détection des armes nucléaires, biologiques et chimiques.

Surveillance des forces militaires, l'équipement et la munition : les dirigeants et les commandants peuvent constamment surveiller l'état des troupes amies, l'état de la disponibilité de l'équipement et des munitions dans un champ de bataille par l'utilisation de réseaux de capteurs [2]. Tous les soldats, véhicules, matériels et munitions peuvent être attachés avec de petits capteurs qui établissent des rapports de leur statut. Ces rapports sont rassemblés et envoyés aux chefs de troupes. Les données peuvent également être transmises aux niveaux supérieurs de la hiérarchie de commandement, tout en étant regroupées avec les données provenant d'autres unités à chaque niveau.

Surveillance du champ de bataille : les terrains critiques, les parcours, les chemins et les ponts peuvent être rapidement couverts par des réseaux de capteurs, ce qui permet de surveiller de près les activités des forces ennemies [2]. A tout moment, de nouveaux réseaux de capteurs peuvent être déployés pour assurer la surveillance de l'évolution des opérations militaires.

Reconnaissance des forces ennemies et de terrain : les réseaux de capteurs peuvent être déployés dans des zones critiques, ce qui permet d'intercepter certains renseignements précieux détaillés et en temps opportun, sur les forces ennemies.

1.3.2. Applications environnementales

Les réseaux de capteurs sont utilisés dans certaines applications environnementales notamment pour localiser les mouvements d'oiseaux, de petits animaux et d'insectes, surveillance des conditions environnementales qui affectent les cultures et le bétail, détection chimique et biologique, agriculture de précision, biologique, la surveillance de la terre et de l'environnement marin, les sols et l'atmosphère, détection des incendies de forêt, la recherche météorologique ou géophysique, détection d'inondation et la pollution.

Détection d'incendie dans les forêts : les nœuds capteurs sont déployés de manière stratégique, ou aléatoire, et d'une façon dense dans une forêt où ils peuvent détecter l'origine exacte de l'incendie et alerter l'utilisateur final avant que le feu se propage et devient incontrôlable [2].

Détection des inondations : un exemple d'une détection d'inondation est le système ALERT [7] déployé aux États-Unis. Plusieurs types de capteurs sont déployés dans le système ALERT, tels que les capteurs de niveau d'eau et les capteurs météorologiques. Les informations capturées alimentent une base centrale de données.

Agriculture de précision : les réseaux de capteurs offrent la possibilité de contrôler le niveau de pesticides dans l'eau potable, le niveau d'érosion des sols, et le niveau de pollution de l'air en temps réel [2].

1.3.3. Applications médicales

Certaines applications fournissent des interfaces pour les handicapés ; elles intègrent la surveillance des patients, les diagnostics, l'administration de médicaments dans les hôpitaux, la télésurveillance des données physiologiques de l'homme ainsi que le suivi et la surveillance des médecins et des patients dans un hôpital.

Le suivi et la surveillance des médecins et des patients dans un hôpital : un ensemble de capteurs minuscules pourraient être attachés au patient [2]. Chaque nœud capteur a une tâche spécifique (la détection du rythme cardiaque ou la pression artérielle). Les médecins peuvent aussi être équipés de capteurs, afin de pouvoir les localiser au sein de l'hôpital.

L'administration de médicaments dans les hôpitaux : des nœuds peuvent être attachés à des médicaments [2], ce qui pourrait réduire le taux d'une mauvaise prescription des médicaments aux patients. Ceci est possible grâce aux nœuds capteurs attachés aux patients qui permettent d'identifier leurs allergies et les médicaments nécessaires.

1.3.4. Applications domestiques

Domotique : le progrès technologique en matière de construction des capteurs et des actionneurs intelligents a permis leur intégration dans les appareils, tels que les aspirateurs, fours à micro-ondes, réfrigérateurs et magnétoscopes [2]. Ces nœuds capteurs à l'intérieur des appareils domestiques peuvent interagir les uns avec les autres et avec le réseau externe via Internet ou par satellite. Ils permettent aux utilisateurs finaux de gérer des dispositifs d'accueil localement et à distance plus aisément.

L'environnement intelligent : les nœuds capteurs peuvent être intégrés dans les meubles et les appareils, et ils peuvent communiquer les uns avec les autres et avec le serveur de la pièce. Le serveur de la pièce peut également communiquer avec d'autres serveurs de pièces pour connaître les services qu'ils offrent, par exemple, l'impression, la numérisation et la télécopie. Ces serveurs de pièces et les nœuds capteurs peuvent être intégrés avec les dispositifs embarqués existants pour devenir auto-organisés, autorégulés. Un autre exemple d'environnement intelligent est «*The Residential Laboratory*» à l'institut de technologie de Georgia [8].

1.3.5. Autres applications commerciales et industrielles

Les exemples d'application n'en manquent pas dans ce domaine, on peut citer quelques uns : le contrôle des matériaux, la gestion des stocks, le contrôle de qualité des produits, le contrôle de l'environnement dans les immeubles et les bureaux, le contrôle de robots dans les environnements de fabrication automatique, les musées interactifs, le diagnostic des machines, etc.

Détection et surveillance des vols de voitures : les capteurs doivent être déployés pour détecter et identifier les menaces dans une région géographique. Le rapport de ces menaces sera envoyé par Internet aux utilisateurs finaux pour analyse.

Gérer le contrôle d'inventaire : les capteurs peuvent être attachés à chaque élément dans un entrepôt. Les utilisateurs finaux peuvent trouver l'emplacement exact de l'article et de les dénombrer facilement. Ainsi, les gérants peuvent suivre et localiser les stocks à tout moment.

1.4. Protocoles de routage

Le routage consiste à trouver un chemin pour envoyer le message de la source à la destination. Dans le cadre des réseaux de capteurs, le routage doit être efficace en énergie. Pour cela, il faut bien sûr être capable de trouver une route qui ne coûte pas trop d'énergie, une route pas trop longue. Mais, il faut aussi être capable de trouver ou de maintenir les routes sans dépenser trop d'énergie. Les protocoles dans lesquels on maintient à jour les relations des nœuds dans le réseau à l'aide d'envois périodiques de paquets "hello" ont un coût constant non négligeable. Ce coût constant est particulièrement pénalisant puisque l'on a des trafics très sporadiques : maintenir une table de routage, pour avoir des routes très efficaces, n'est pas intéressant si l'on n'utilise que très rarement ces routes.

Les protocoles de routage spécifiques aux réseaux de capteurs doivent tenir compte du type de communications induit par l'application. Outre le fait que la quantité de données échangées est très faible par rapport aux applications de types réseaux ad hoc, notons que le trafic est particulièrement prévisible puisqu'il va des nœuds vers le puits ou du puits vers les nœuds.

Nous ne faisons pas ici un état de l'art exhaustif des protocoles de routage. Nous voulons seulement présenter des protocoles types des réseaux capteurs. Nous les avons choisis parce qu'ils sont représentatifs des protocoles de routage pour réseaux de capteurs.

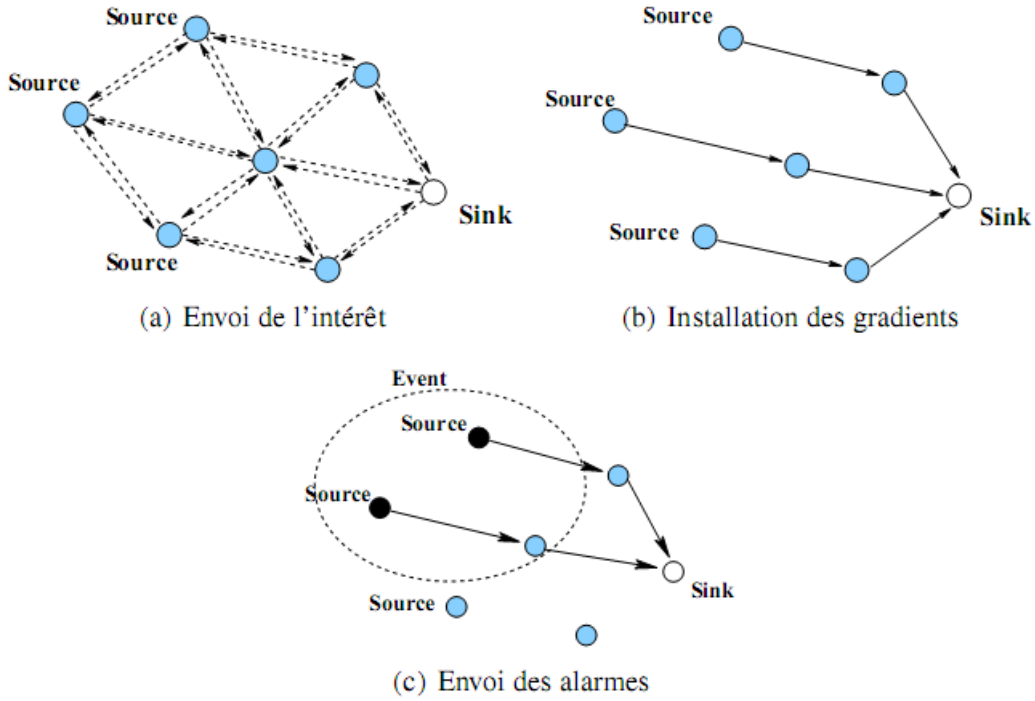


Figure 1.5. Schéma du mécanisme de routage par diffusion dirigée

1.4.1. Inondation

L'inondation ("flooding", en anglais) consiste à envoyer un message à tout le réseau. L'émetteur envoie le message à tous ses voisins. Chaque voisin envoie à son tour le message à tous ses voisins et ainsi de suite. Les nœuds vont donc recevoir le même message plusieurs fois de différents voisins. Pour éviter que le message ne se multiplie dans le réseau, chaque nœud ne le renvoie qu'une seule fois. Pour ce faire, chaque message envoyé en inondation a un identifiant unique. Les nœuds qui réémettent le message notent l'identifiant. S'ils reçoivent à nouveau un message avec cet identifiant, ils ne le renvoient pas [11].

1.4.2. Diffusion dirigée

L'algorithme de diffusion dirigée ("directed diffusion", en anglais) a été proposé en 2000 dans [12]. Depuis, de nombreuses améliorations ont été proposées. Nous présentons rapidement son principe, le lecteur intéressé est encouragé à lire le papier [13].

Le principe de l'algorithme est le suivant : le puits envoie une requête à tout le réseau. Cette requête est envoyée à l'aide du mécanisme de routage précédent, l'inondation. Les nœuds concernés par cette requête répondent au puits en envoyant un message qui emprunte la route inverse. Pour cet algorithme, on suppose que les liens radio sont bidirectionnels. Il s'agit d'un algorithme local, les nœuds n'ont que la connaissance de leur voisinage. Pour joindre le puits, un nœud envoie son message au nœud duquel il a reçu en premier le message du puits. Chaque nœud a seulement besoin de savoir par quel voisin il pourra joindre le puits. Ce voisin ayant également connaissance d'un nœud grâce auquel il pourra joindre le puits, de proche en proche, le message arrivera à destination.

Le puits envoie donc sa requête à tous les nœuds du réseau à l'aide de l'inondation, voir figure 1.5. Les nœuds reçoivent alors le même message de plusieurs de leurs voisins. Ils mémorisent quel nœud leur a envoyé l'intérêt en premier. C'est à ce nœud qu'ils enverront les données (reçues ou mesurées localement) destinées au puits. Avoir des liens radio bidirectionnels est essentiel : on considère que le chemin le plus rapide du nœud au puits est aussi le chemin le plus rapide du puits au nœud. On dit que les nœuds installent des gradients, figure 1.5(b). Sur l'exemple représenté dans la figure 1.5, l'intérêt ne concerne que les nœuds "Source", à gauche du réseau. La requête demande que l'on envoie une alarme si un évènement est détecté. L'évènement est ensuite détecté par trois capteurs (figure 1.5(c)). Mais seuls les deux nœuds concernés par l'intérêt vont envoyer un message. Le message est envoyé en suivant le gradient, c'est-à-dire au voisin par lequel le nœud a reçu la requête du puits. Celui-ci, même s'il n'est pas concerné par la requête, a établi un gradient et sait à quel nœud il doit envoyer le message. Et ainsi de suite, de voisin en voisin, jusqu'au puits.

Il existe aussi des mécanismes dits de renforcement pour consolider certaines routes. Différents

mécanismes sont proposés, on ne les détaillera pas ici.

Le but de cet algorithme est donc de fournir un protocole de routage et d'organisation pour des applications dans lesquelles un puits pilote le réseau en envoyant des requêtes et en récupérant les données. Les requêtes peuvent être ponctuelles, par exemple pour demander le relevé de mesure à l'instant courant. Elles peuvent être périodiques, exemple : "envoyez-moi la température tous les jours". Enfin, elles peuvent dépendre des capteurs : "si la température dépasse un certain seuil, envoyez une alarme". Les auteurs appellent les paquets émis par le puits des intérêts, ce qui montre bien que le puits est intéressé par une certaine information disponible à l'aide de son réseau.

1.5. Facteurs de conception des réseaux de capteurs

La conception et la réalisation des réseaux de capteurs sans fil est influencée par plusieurs paramètres, parmi lesquels nous citons la tolérance aux pannes, la scalabilité, le coût de production, l'environnement d'exploitation, la topologie du réseau, la localisation, les contraintes matérielles, le support de transmission et la consommation d'énergie. Ces facteurs importants servent comme directives pour le développement des algorithmes et protocoles utilisés dans les réseaux de capteurs. Ils sont considérés également comme métriques de comparaison de performances entre les différents travaux dans le domaine.

Dans ce travail, nous allons nous intéresser aux problèmes de pannes et d'énergie. Pour la consommation d'énergie, nous allons étudier l'influence de certains phénomènes (notamment le phénomène de rappel) lors de la communication entre les nœuds du réseau de capteurs.

1.5.1. Consommation d'énergie

1.5.1.1. Formes de dissipation d'énergie

Les nœuds capteurs sont alimentés principalement par des batteries. Ils doivent donc fonctionner avec un bilan énergétique frugal. En outre, ils doivent le plus souvent avoir une durée de vie de l'ordre de plusieurs mois, voir de quelques années, puisque le remplacement des batteries n'est pas une option envisageable pour des réseaux avec des milliers de noeuds.

Afin de concevoir des solutions efficaces en énergie, il est extrêmement important de faire d'abord une analyse des différents facteurs provoquant la dissipation de l'énergie d'un nœud-capteur [14].

Cette dissipation d'énergie se fait de manière générale selon plusieurs modes :

- **Le MCU (l'unité du microcontrôleur):** Généralement les MCUs possèdent divers modes de fonctionnement : « actif », « oisif », et « sommeil », à des fins de gestion de l'énergie. Chaque mode est caractérisé par une quantité différente de consommation d'énergie. Par exemple, le MSP430

consomme 3 mW en mode actif, 98 µW en mode « oisif » et seulement 15 µW dans le mode sommeil. Toutefois, la transition entre les modes de fonctionnement implique un surplus d'énergie et de latence. Ainsi, les niveaux de consommation d'énergie des différents modes, les coûts de transition entre les modes et aussi le temps passé par le MCU dans chaque mode, ont une incidence importante sur la consommation totale d'énergie d'un noeud-capteur.

- **La radio:** la radio opère en quatre modes de fonctionnement : émission, réception, oisif, et sommeil. Une observation importante dans le cas de la plupart des radios, est que le mode « oisif » induit une consommation d'énergie significative, presque égale à la consommation en mode réception [15]. Ainsi, il est plus judicieux d'éteindre complètement la radio plutôt que de passer en mode « oisif » quand on a ni à émettre ni à recevoir de données. Un autre facteur déterminant est que, le passage de la radio d'un mode à un autre engendre une dissipation d'énergie importante due à l'activité des circuits électroniques. Par exemple, quand la radio passe du mode sommeil au mode émission pour envoyer un paquet, une importante quantité d'énergie est consommée pour le démarrage de l'émetteur lui-même [14]. Un autre point important est que les données des constructeurs sous-estiment assez régulièrement ces différentes consommations comme ont pu le montrer les auteurs de [16], en particulier concernant la consommation dans le mode « oisif ».
- **Le détecteur ou le capteur proprement dit:** il y a plusieurs sources de consommation d'énergie par le module de détection, notamment l'échantillonnage et la conversion des signaux physiques en signaux électriques, le conditionnement des signaux et la conversion analogique-numérique. Étant donné la diversité des capteurs, il n'y a pas de valeurs typiques de l'énergie consommée. En revanche, les capteurs passifs (température, sismiques, ...) consomment le plus souvent peu d'énergie par rapport aux autres composants du nœud-capteur. Notons, les capteurs actifs tels que les sonars, les capteurs d'images, etc. peuvent consommer beaucoup d'énergie.

En outre, il existe d'autres formes de dissipation d'énergie telles que les lectures et les écritures mémoire. Un autre aspect non négligeable est le phénomène d'autodécharge de la batterie. En effet, cette dernière se décharge d'elle même et perd de sa capacité au fil du temps. Il est difficile d'apporter ici une étude quantitative et comparative précise de la consommation de chaque composant d'un nœud-capteur en raison du grand nombre de plates-formes commerciales existantes. Cependant, des expérimentations ont montré que c'est la transmission de données qui est la plus consommatrice en énergie [14]. Le coût d'une transmission d'un bit d'information est approximativement le même que le coût nécessaire au calcul d'un millier d'opérations [17]. La consommation du module de détection dépend du type spécifique du noeud-capteur [18].

1.5.1.2. Sources de surconsommation d'énergie

Nous appelons surconsommation d'énergie toute consommation inutile que l'on peut éviter afin de conserver l'énergie d'un noeud-capteur. Les sources de cette surconsommation sont nombreuses, elles peuvent être engendrées lors de la détection lorsque celle-ci est mal gérée (par exemple : une fréquence d'échantillonnage mal contrôlée) [19].

La surconsommation concerne également la partie communication. En effet, cette dernière est sujette à plusieurs phénomènes qui surconsomment de l'énergie surtout au niveau MAC où se déroule le contrôle d'accès au support sans fil. Certains de ces phénomènes sont les causes majeures de la perte d'énergie et ont été recensés dans [20, 21] :

- **Les collisions:** elles sont la première source de perte d'énergie. Quand deux trames sont émises en même temps et se heurtent, elles deviennent inexploitables et doivent être abandonnées. Les retransmettre par la suite, consomme de l'énergie. Tous les protocoles MAC essayent à leur manière d'éviter les collisions. Ces dernières concernent plutôt les protocoles MAC avec contention.
- **L'écoute à vide (oisif listening):** un noeud dans l'état « oisif » est prêt à recevoir un message, mais il n'est pas actuellement en train de recevoir quoi que ce soit. Ceci est coûteux et inutile dans le cas des réseaux à faible charge de trafic. Plusieurs types de radios présentent un coût en énergie significatif pour le mode « oisif ». Eteindre la radio est une solution, mais le coût de la transition entre les modes consomme également de l'énergie, la fréquence de cette transition doit alors rester « raisonnable ».
- **L'écoute abusive (overhearing):** cette situation se présente quand un nœud reçoit des messages qui ne lui sont pas destinés. Le coût de l'écoute abusive peut être un facteur dominant de la perte d'énergie quand la charge de trafic est élevée et la densité des nœuds est grande.
- **L'overmitting:** Cette situation arrive quand un nœud envoie des données et le nœud destinataire n'est pas prêt à les recevoir.
- **L'overhead des paquets de contrôle:** l'envoi, la réception, et l'écoute des paquets de contrôle consomment de l'énergie. Comme les paquets de contrôle ne transportent pas directement des données, ils réduisent également le débit utile effectif.

1.5.2. La tolérance aux pannes

La défaillance ou le blocage des nœuds dans un réseau de capteurs peut être engendré par plusieurs causes, notamment l'épuisement d'énergie, l'endommagement physique, ou les interférences liées à l'environnement. La propriété de tolérance aux pannes est définie par l'habilité du réseau à maintenir ses fonctionnalités sans interruptions provoquées par la panne des capteurs. Elle vise donc à minimiser l'influence de ces pannes sur la tâche globale du réseau [22].

Cette propriété notée R(t) a été modélisée dans [23] par une distribution de *Poisson,* où R(t) donne la probabilité de ne pas avoir une panne pour un nœud capteur pendant l'intervalle de temps [0,t].

$R(t) = \exp(-\lambda_k t)$

Où : λ_k est le taux de panne du nœud capteur k [24] .

Les protocoles conçus pour les réseaux de capteurs doivent atteindre le niveau de tolérance aux pannes requit par le réseau. Cela dépend essentiellement de l'environnement de déploiement du réseau, des caractéristiques des micro-capteurs, etc.

En effet, si le réseau de capteurs est destiné aux environnements avec un faible degré d'interférences, tels que ceux utilisés dans les bâtiments pour surveiller le taux d'humidité et le degré de température, les protocoles utilisés ne doivent pas cibler une grande tolérances aux pannes, car dans ce type de réseau, il n'existe pas une grande interférence avec l'environnement, et ses nœuds ne sont pas exposés au risque d'endommagement.

Par contre, si le réseau est destiné aux applications militaires, telles que la surveillance et le contrôle d'un champ de bataille, le niveau de tolérance aux pannes visé par les protocoles employés doit être très élevé, car les nœuds sont exposés à un grand risque d'endommagement par des actions hostiles, et les informations captées sont très critiques.

Par conséquent, le niveau de tolérance aux pannes requis dépend de l'application du réseau de capteurs conçu, et les schémas de conception doivent prendre en charge ce paramètre.

Dans notre travail, on va prendre en compte les pannes des capteurs pour savoir leurs influences sur les performances du réseau de capteurs [24].

1.6. Les simulateurs de réseaux

Lors de la conception des réseaux de capteurs, l'analyse se fait habituellement par simulateurs. Parmi les simulateurs de réseaux, on trouve les simulateurs de réseaux classiques, qui ont été conçus pour modéliser et simuler : des réseaux filaires, des réseaux sans fil voir des réseaux ad hoc. Pour ces réseaux, la consommation d'énergie n'était pas encore la préoccupation majeure.

Nous présentons dans ce qui suit une synthèse des simulateurs de réseaux classiques et des simulateurs dédiés aux réseaux de capteurs.

1.6. 1. Simulateurs de réseaux généraux

Un des simulateurs de réseaux les plus utilisés est NS (Network Simulator) et ses extensions NS2 et NS3. L'origine de NS est le projet VINT en 2000, Breslau et al, estiment qu'il faut un seul simulateur de réseaux pour la communauté scientifique. Ce souhait donne naissance au projet VINT. L'intérêt d'avoir un simulateur unique est essentiellement la facilité à comparer différentes solutions. NS2 est un simulateur à évènements discrets. Il propose quatre niveaux d'abstraction [25], ce qui permet d'adapter le simulateur aux différents intérêts. En effet, certains souhaiteront des informations bas-niveau, pour étudier par exemple l'effet d'un système multi-antennes alors que d'autres étudient les protocoles de routage et ne souhaitent que des informations au niveau réseau. Un simulateur qui calcule des informations trop précises passera moins bien à l'échelle qu'un simulateur dédié au routage. NS était d'abord destiné aux réseaux filaires ce qui explique certainement la simplicité des modèles de propagation radio de ce simulateur. Malgré les différents niveaux d'abstraction, NS est essentiellement utilisé par les gens qui s'intéressent aux protocoles de routage et/ou aux protocoles d'accès au médium. Le code de NS est ouvert (open source) ce qui permet à chacun d'ajouter sa contribution. Il existe aussi une large bibliothèque de protocoles MAC et routage, qui permet en effet de comparer facilement ses dernières avancées avec l'état de l'art. D'autre part, il génère des traces qui détaillent tous les évènements de la simulation. Avec l'intérêt grandissant de la communauté de recherche en réseaux pour les réseaux de capteurs, NS est naturellement resté un simulateur très utilisé.;

Cependant, selon certains chercheurs trouvent que NS n'est pas le simulateur parfait. Tout d'abord, la modélisation sommaire des propagations radio est un point faible pour des réseaux dans lesquels toutes les communications sont des communications radio. En partie à cause des multiples contributions qui enrichissent NS, ce simulateur est devenu très compliqué. De plus, les composants qui constituent un réseau, ou les nœuds d'un réseau, ne sont pas toujours bien séparés dans NS. Même s'il est logique, pour implémenter des protocoles qui contiennent des optimisations inter-couches ("cross-layer design"), de devoir modifier du code à différents niveaux, dans NS les couches n'étant pas clairement séparées, l'implémentation devient vite illisible. De plus, il est très difficile d'estimer le rapport entre les simulations effectuées par NS et la réalité. En effet, des améliorations sont proposées par de multiples contributeurs, mais il est difficile de dire si elles sont valides par rapport à la réalité. NS est capable de simuler un réseau d'une centaine de nœuds maximum. Enfin, NS ne propose pas de modélisation de la consommation d'énergie. Pour estimer l'efficacité en énergie de leurs protocoles, les chercheurs font des abstractions assez brutales comme par exemple : compter le nombre de paquets émis/reçus, compter le temps pendant lequel la radio est allumée. De même, NS n'étant pas un simulateur dédié aux réseaux de capteurs, il ne propose pas de modèle

d'environnement. Pour faire des simulations avec des communications, on peut choisir d'avoir un flux (TCP/IP, UDP...) ou bien on considère qu'une loi de Poisson modélise bien l'émission de paquets au niveau d'un nœud. Pour les réseaux de capteurs, ces deux approches sont tout à fait discutables.

Les auteurs de NAB (Network in A Box) [18], souhaitent répondre aux manquements de NS en proposant un simulateur qui passe à l'échelle, contenant un outil de visualisation inclus, et une architecture propre et flexible. Ils utilisent OCAML pour programmer leur simulateur. OCAML dispose d'un typage fort qui oblige à écrire les programmes de manière plus propre. Notamment grâce à OCAML, NAB passe mieux à l'échelle que NS. Un outil de visualisation basé sur l'outil graphique d'OCAML est disponible. NAB ne propose ni modèle d'environnement, ni modélisation de l'énergie. NAB n'est plus disponible. Nous présentons maintenant des simulateurs de réseaux de capteurs.

1.6.2. Simulateurs dédiés aux réseaux de capteurs

AVRORA [26] est un simulateur pour réseaux de capteurs. Pour obtenir une simulation fiable, il simule pour chaque nœud toutes les instructions qui s'exécutent sur ce nœud. Certe, cette approche offre une modélisation particulièrement précise mais on ne peut pas espérer qu'elle passe à l'échelle. Même dans le domaine des systèmes sur puce, une modélisation cycle-précis s'avère trop détaillée pour être utilisable en pratique. De plus, Avrora est écrit en Java et chaque nœud est implémenté par un thread (processus léger) Java ce qui impose une difficulté supplémentaire : il faut synchroniser les threads pour s'assurer qu'un nœud ne reçoive pas un paquet avant que son voisin ne l'ait envoyé.

ATEMU [27] est aussi un simulateur pour réseaux de capteurs très précis. Ici, les nœuds sont émulés, c'est-à-dire que l'on exécute le code binaire de chaque nœud. Par contre, les transmissions sans fil sont simulées. Une fois encore, il nous semble qu'une telle approche est trop précise. On pourrait cependant imaginer que l'on émule un nœud et que l'on simule moins finement les autres mais on ne peut émuler tous les nœuds d'un réseau comportant plusieurs centaines de nœuds.

TOSSIM [28] est le simulateur dédié à TinyOS. TinyOS [29, 30] est un système d'exploitation dédié aux réseaux de capteurs. TOSSIM essaye de tirer parti du mode d'exécution de TinyOS pour proposer un simulateur efficace et fiable. Le mode d'exécution de TinyOS est dirigé par les évènements, ce mode d'exécution se calque bien sur un simulateur à évènements discrets. TOSSIM contient un modèle abstrait de chaque composant du matériel d'un nœud. Pour une simulation TOSSIM, on utilise le même code que celui destiné au nœud cible mais cette fois-ci TOSSIM émule le comportement du matériel en utilisant les modèles des composants. Simuler exactement le code qui tournera sur les nœuds permet de tester l'implémentation finale des algorithmes. Cette notion d'abstraction du matériel est tout à fait intéressante, cependant TOSSIM dans sa première version ne

permet pas d'estimer l'énergie consommée.

POWERTOSSIM [31] est l'extension de TOSSIM qui contient un modèle de consommation d'énergie. Pour les valeurs de consommation, les auteurs se sont basés sur le Mica2 (nœud développé à l'université de Berkeley). Les auteurs connaissent les consommations des différents composants de ce nœud suivant leurs états. Il faut donc connaitre l'état de chaque composant d'un nœud pendant la simulation. Grâce au modèle de simulation basé sur TinyOS, on connait immédiatement l'état des composants autres que le microcontrôleur puisque les changements d'états correspondent à des évènements dans TinyOS et donc dans TOSSIM. Plusieurs composants du nœud sont parfois abstraits dans TOSSIM par un seul composant. L'estimation de la consommation du microcontrôleur est plus délicate : il faut instrumenter le code pour être capable de compter le nombre d'exécutions de chaque bloc d'instructions, et il faut faire correspondre chaque bloc d'instructions avec son code en assembleur. Lors de la simulation, on note le nombre de passages, d'exécutions, de chaque bloc d'instructions. Sachant combien d'instructions élémentaires contient chaque bloc de base, on en déduit le nombre d'instructions effectuées par le microcontrôleur et donc sa consommation. Cette approche est intéressante, mais elle ne permet pas de varier la précision du modèle de consommation. Enfin, les simulateurs TOSSIM et PowerTOSSIM ne conviennent que pour des applications écrites en TinyOS.

1.6.3. Simulateurs prenant en compte un modèle d'environnement

Tous ces simulateurs sont dédiés aux réseaux de capteurs et cherchent donc à estimer de façon précise la consommation tout en restant capables de simuler des réseaux d'assez grande taille. Cependant, nous faisons remarquer qu'il faut inclure un modèle d'environnement pour que les simulations aient un trafic réaliste. En effet, dans un réseau de capteurs, le trafic dépend des capteurs et donc de l'environnement. Les simulateurs que nous venons de présenter ne contiennent pas de modèle d'environnement et il est difficile de savoir avec quel trafic sont générées les simulations. Nous faisons maintenant un état de l'art des simulateurs qui prennent en compte un modèle d'environnement.

Sridharan et al [32] proposent de connecter le simulateur d'environnements Matlab [33] au simulateur de réseaux de capteurs TOSSIM. L'expérience qu'ils ont faite est la simulation d'un réseau de capteurs dédié au contrôle de la structure d'un bâtiment. Pour cette application, Matlab fournit un bon modèle de l'environnement ; cependant Matlab convient beaucoup moins à la simulation d'environnements qui ne suivent pas des équations différentielles.

Outre ce lien entre Matlab et TOSSIM, les simulateurs de réseaux qui incluent un modèle d'environnement le font souvent en faisant une analogie entre la propagation des ondes radio et la

propagation du phénomène à observer.

SensorSim [34] est un simulateur à événements discrets pour réseaux de capteurs. Pour simuler l'environnement, les nœuds de ce simulateur contiennent en plus du module radio un module capteur qui dispose d'une couche protocolaire ("Sensor protocol stack") qui reçoit des messages venant d'un canal capteur ("Sensor channel"). La différence principale entre ce module capteur et le module radio est que les nœuds ne peuvent que recevoir sur ce canal, ils ne peuvent pas émettre. Pour effectuer une simulation, il faut déterminer les caractéristiques de propagation du phénomène à observer sur le canal capteur.

J-Sim [35] s'inspire de SensorSim pour la modélisation de l'environnement. J-Sim contient donc un canal capteur. Le phénomène à capter est créé par un nœud particulier appelé "Target node", ce nœud envoie périodiquement des stimuli qui se propagent sur le canal "capteur". Deux modèles de propagation sont implémentés : un modèle de propagation sismique et un modèle de propagation acoustique.

Downard, dans [36], étend NS2 pour simuler des réseaux de capteurs. Ici aussi, l'auteur fait une analogie entre le canal capteur et le canal radio, mais il va plus loin dans cette analogie. Un canal radio est créé pour chaque phénomène. Il y a deux types de nœuds. En plus des nœuds classiques (les capteurs du réseau), qui communiquent sur le canal radio et reçoivent des informations des canaux capteurs, Downard crée des nœuds PHENOM. Les nœuds PHENOM ne "communiquent" que sur un canal capteur. Ces nœuds émettent régulièrement un paquet indiquant leur présence. Les nœuds PHENOM disposent également d'une couche MAC et d'une couche routage. Pour éviter des collisions entre phénomènes qui ne seraient pas réalistes, la couche MAC utilisée est parfaite. Le routage détermine quand et avec quelle fréquence les nœuds PHENOM envoient les messages indiquant leur présence. C'est le protocole de routage qui détermine la propagation du phénomène. Selon nous, utiliser le modèle de propagation des ondes radio pour simuler la propagation de phénomènes quelconques n'est pas forcément très réaliste. Certains phénomènes à détecter, comme une cible par exemple, se déplacent mais ne se propagent pas et dans ce cas l'analogie avec la radio atteint déjà ses limites. Enfin, le modèle de propagation radio est une partie très coûteuse en calculs dans un simulateur de réseaux, dupliquer ce composant paraît inutilement coûteux. Dans le cas de cette dernière approche, on surcharge le simulateur de nouveaux nœuds comportant protocoles MAC et routage uniquement pour le composant environnement de la simulation.

1.7. Les modèles formels et les réseaux de capteurs

Vue les limites de la simulation, l'application des modèles mathématiques formels s'avère indispensable lors de la conception des réseaux de capteurs. Ce type de modélisation vise à avoir des résultats exacts en appliquant des formules analytiques ce qui rend le modèle facilement analysable ou bien en appliquant des méthodes numériques ou algorithmiques pour des modèles plus complexes.

Ces analyses peuvent s'intéresser au cas moyen comme le font les simulations mais elles permettent aussi d'exhiber le pire cas du modèle par rapport à la propriété à valider. Ces modèles permettent ainsi une meilleure compréhension du système.

Une des techniques de modélisation pour l'évaluation de performances consiste à modéliser le système en utilisant des probabilités. Kleinrock et Tobagi [37], [38] ont fait des travaux précurseurs en modélisant les communications sans fil à l'aide de probabilités. Leur but était l'étude de protocoles MAC. Ce type d'étude peut également être utile dans le cas des réseaux de capteurs. Par exemple, dans notre travail, nous modélisons un canal radio entre deux nœuds avec l'hypothèse suivante : chaque message envoyé a une probabilité constante et indépendante du temps d'être transmis correctement. Sous cette hypothèse, nous comparons différent stratégies afin de vérifier leur robustesse et leur efficacité en énergie.

Demirkol et al [39] proposent une modélisation analytique de l'environnement. L'application qui les intéresse est la détection d'intrusion. Pour cela, ils modélisent à l'aide de probabilités, le déplacement de la cible dans un champ de capteurs.

De nombreux composants d'un réseau de capteurs peuvent donc être modélisés afin d'évaluer leurs performances. Cependant, il est difficile voire impossible d'arriver à de tel modèles sans des simplifications drastiques. La question qui se pose alors est, quel lien existe-t-il entre le modèle abstrait sur lequel on a des résultats et la réalité? Les auteurs dans [40] proposent pour répondre à cette question de modéliser un ensemble de systèmes types à l'aide de différentes techniques et outils. Les systèmes considérés ne sont pas des réseaux de capteurs mais des systèmes temps réel distribués dans lesquels des unités de calculs exécutent des tâches concurrentes et communiquent. Le but de l'analyse est d'estimer les temps d'exécutions pire cas (WCET). Ils proposent un ensemble de systèmes types qui servent de bancs d'essais. Ensuite, ils modélisent et analysent ces systèmes à l'aide de différentes techniques chacune implémentant des abstractions propres. En outre, une modélisation à l'aide d'automates temporisés explore de façon exhaustive tous les cas possible du système. Cette analyse, plus coûteuse que les autres, sert de référence. Le résultat est le suivant : les différentes méthodes d'analyses fournissent des pire cas pessimistes que l'on ne peut pas comparer. Autrement dit, il est impossible de savoir à l'avance pour un cas d'étude particulier quelle analyse et

donc quelles abstractions fourniront le résultat le plus fidèle. Ce résultat montre à quel point il est difficile de relier un résultat obtenu par une modélisation abstraite avec la réalité. Les auteurs recommandent aux concepteurs de tels systèmes d'utiliser plusieurs modélisations et abstractions différentes afin d'accroître la confiance en leurs résultats.

Les réseaux de capteurs sont également des systèmes complexes pour lesquels nous pensons qu'il faut être très prudent avec des modélisations mathématiques trop abstraites. Pour être utilisables en pratique, les modélisations mathématiques peuvent s'avérer éloignées de la réalité. Il est en effet difficile d'exprimer le comportement complexe d'un réseau de capteurs à l'aide de modèles mathématiques principalement pour deux raisons. Premièrement, un réseau de capteurs est un système qui comporte de nombreux éléments logiciels et matériels qu'il faut modéliser. De plus pour modéliser un réseau de capteurs, on est amené à modéliser des comportements physiques (communications radio par exemple) pour lesquels les modèles mathématiques existants sont complexes.

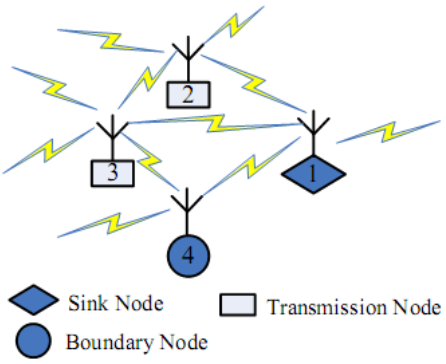
1.7.1. Modélisation par les files d'attente

Hao [64], a utilisé le modèle de files d'attente pour analyser les performances des protocoles MAC dans un réseau de capteurs. L'auteur a proposé un protocole basé sur TDMA (Temporel Detection Medium Acces) avec réutilisation des canaux. Il a considéré que les nœuds du réseau sont non fiables et les capteurs peuvent se trouver dans l'un des deux états : actif ou en veille. Le modèle de file d'attente utilisé par l'auteur pour analyser le réseau de capteur est : M/M/1. Après l'analyse du modèle proposé, l'auteur a dérivé différents paramètres de performances tels que le temps moyen d'envoi d'un paquet et le taux moyen de la mise en veille.

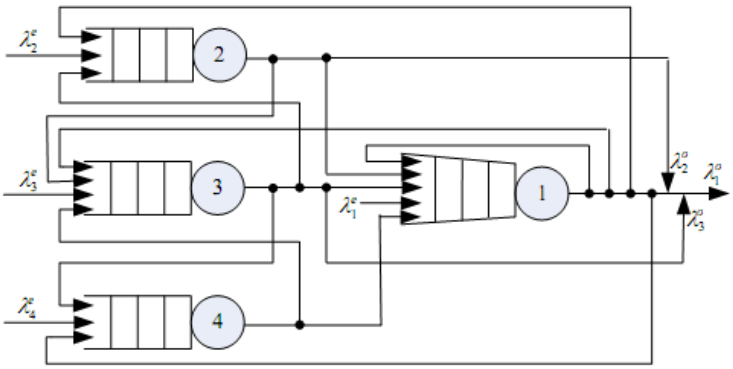
Tie Qiu et al [65], ont utilisé le modèle de file d'attente M/M/1/N pour analyser la capacité du buffer des nœuds dans le but de la détection et du contrôle de congestion dans un réseau de capteurs sans fil. Les auteurs ont considéré différentes stratégies et ont étudié leur influence sur la qualité du service. Le réseau de capteurs considéré par les auteurs est un réseau à capteurs immobiles. Le calcul des résultats est fait à l'aide d'algorithmes itératifs

Tamàs Bérczes et al [66], ont utilisé le modèle de file d'attente avec vacances des serveurs, appels répétés et service à priorité pour modéliser un nœud d'un réseau de capteurs. Deux priorités sont définis: une pour la file d'attente (la stratégie utilisée est: FIFO) et une autre pour les paquets qui joignent l'orbite. La politique de vacance permet essentiellement à un capteur de consommer moins d'énergie. Les auteurs ont utilisé l'outil logiciel *Mosel* pour l'évaluation numérique des formules et l'analyse du modèle proposé. Des résultats sous forme de graphes et de tableaux sont produits par l'outil *Mosel* pour illustrer l'effet de certains paramètres sur le temps moyen de réponse. La figure-

1.6 montre la topologie et le modèle utilisés par les auteurs.



(a) : Topologie du WSN étudié



(b) : Relation entre les nœuds de réception et de l'envoi

Figure 1.6 : Topologie et modélisation des nœuds de transferts du WSN [66]

Qaoqin Li et al [67], ont utilisé le modèle de file d'attente (comme montre la figure 1.8) pour étudier les performances du cluster-head dans un réseau de capteurs basé sur le PMRC (Progressive Multi-hop Rotational Clustered). Les auteurs ont considéré le PMRC imbriqué et non imbriqué lors de la modélisation. Ils ont dérivé le temps moyen de réponse pour le cluster-head et la taille de la file d'attente dans plusieurs couches. Les couches considérées sont définies par la distance par rapport au sink (voir figure 1.7). La figure 1.8 représente le modèle proposé par les auteurs. Plusieurs graphes sont présentés afin de savoir et choisir les bons paramètres pour un bon fonctionnement du réseau de capteurs.

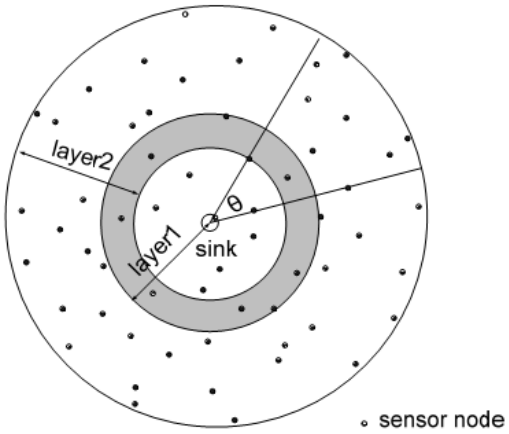


Figure 1.7 : Couches imbriquées dans un WSN basé sur le PMRC [67]

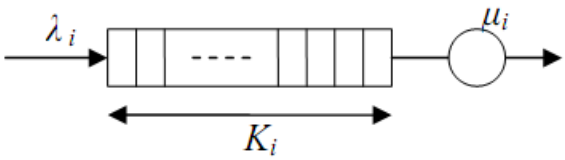


Figure 1.8 : Modèle de file d'attente pour un cluster-head dans une *couche i* [67]

1.7.2. Modélisation par les réseaux de Petri stochastique généralisé

Shi Zhang-song [42], a traité le problème de consommation d'énergie dans les réseaux de capteurs. Il a constaté que le mécanisme du sommeil est une technique efficace d'économie d'énergie. Cette technique est largement étudiée et utilisée. Le plan d'énergie qui décrit l'information sur l'énergie restante dans chaque partie du réseau est utile pour prolonger la durée de vie du réseau. Dans cet article, l'auteur propose un modèle basé sur les réseaux de Petri stochastiques généralisés pour prédire la consommation d'énergie d'un réseau de capteurs. Le mécanisme du sommeil est proposé pour la construction du plan d'énergie. Le modèle fournit une base théorique et une fondation numérique pour assurer une utilisation efficace de l'énergie.

Vüchner et al [41], ont proposé un modèle qui permet de trouver un compromis entre l'efficacité énergétique et la performance des réseaux de capteurs sans fil en montrant les aspects positifs et négatifs des messages bloqués et des messages perdus dans le cas où un nœud capteur est en état de veille ou en panne. Le modèle se base sur les files d'attente avec rappel dans lesquelles le client qui arrive et trouve tous les serveurs occupés, quitte le système temporairement et rejoint une orbite. Une orbite est une zone tampon imaginaire où les clients rappellent pour obtenir le service jusqu'à ce qu'ils soient bien servis. Les auteurs ont supposé et traité le cas où les nœuds capteurs sont non fiables et pour ce dernier, ils ont considéré la politique des pannes indépendantes. Ils ont introduit la notion de rappel dans le processus d'envoi d'un message sur un protocole point à point. Ils ont utilisé les RDPSG pour la représentation

graphique et le traitement analytique. En bénéficiant de la puissance du modèle de RDPSG, les auteurs ont analysé les performances du phénomène d'envoi de messages entre deux nœuds voisins. Ils ont utilisé l'outil ***MOSEL-2*** pour le calcul numérique des paramètres de performances.

1.8. Problématique et hypothèses pour la modélisation du système

Nous considérons un lien radio entre un nœud et ses voisins. Nous cherchons à savoir les paramètres de performances d'une transmission d'un paquet de données d'un nœud à l'autre via ce lien radio. Nous définissons le temps moyen de réponse d'une telle transmission comme étant la somme des temps du rappel et temps d'attente et temps de l'émission du paquet à un capteur voisin. Le but de l'étude est de savoir quels paramètres techniques (par exemple : le temps de veille d'un capteur) faut-il choisir suivant la fiabilité souhaitée et l'état du canal radio.

En fait, nous cherchons à minimiser ici, le temps pendant lequel la radio des nœuds est allumée lors d'une transmission. Ce temps n'est pas directement proportionnel à la consommation d'énergie puisque d'autres éléments matériels des nœuds consomment de l'énergie.

Conclusion

Dans ce chapitre, nous avons présenté le nœud capteur sans fil avec son architecture interne. Nous avons ensuite présenté les réseaux de capteurs et leurs propriétés tout en mettant en évidence leurs applications, ainsi que les défis auxquels ils sont confrontés.

A travers ce chapitre, nous avons montré l'apport des capteurs et en l'occurrence des réseaux de capteurs mais nous avons aussi montré du doigt les contraintes de ces environnements dont la limitation énergétique, la limitation de la mémoire et une déficience en environnements hostiles présentent des obstacles divers ainsi que les difficultés de déploiement. Nous avons aussi présenté quelques exemples d'application des réseaux de capteurs.

En plus, nous avons présenté un état de l'art sur les outils utilisés lors du pré-déploiement d'un réseau de capteurs. Nous avons présenté les outils de simulation et la modélisation pour l'évaluation des performances pour un réseau de capteurs. Nous avons présentés quelques modélisations par les files d'attente et par les réseaux de Petri stochastiques généralisés.

Dans la suite du manuscrit, nous allons présenter le méta-modèle RDPSG dans un chapitre à part. En se basant sur cet outil, nous éclairons la notion du rappel dans le trafic de communication dans un réseau de capteur sans fil dont les nœuds subis à des pannes aléatoires. Nous allons essayer de voir l'influence de certains facteurs techniques sur la performance du réseau en traitant différentes stratégies de pannes.

Chapitre 2 : Réseaux de Petri stochastiques généralisés

Introduction

Les réseaux de Petri permettent une description simple et efficace des systèmes parallèles. Cependant, avec un RDP simple seule l'évaluation qualitative est possible, et aucune modélisation temporelle n'est prise en compte. Or, l'évaluation quantitative nécessite une temporisation du modèle de base. Ainsi, le modèle des réseaux de Petri stochastiques a été introduit. Ce modèle associe à chaque transition un délai de franchissement aléatoire.

2.1. Définition informelle

Un réseau de Petri stochastique est un réseau de Petri dans lequel est associé à chaque transition une variable aléatoire de loi exponentielle représentant le délai de tir ou le délai de franchissement, ou encore le travail qui doit être réalisé pour qu'une opération aboutisse et produise le franchissement de la transition. A toute transition, correspondra alors un taux de franchissement

2.2. Définition formelle [43]

Un réseau de Petri stochastique (**RDPS**) est un couple **S =<R,W>** où :

- **R** est le réseau de Petri marqué défini par le cintuplé : R = < P,T,Pré,Post,Mo> tel que :
  - P = $\{p_1,p_2,\ldots,p_n\}$ est un ensemble fini de places, pour lequel n = |P| ;
  - T = $\{t_1,t_2,\ldots,t_m\}$ est un ensemble de transitions, pour lequel m = |T| ;
  - Pré : P x T → IN est une application d'incidence avant, telle que Pré(p,t) contient la valeur entière associée à l'arc allant de la place p à la transition t ;
  - Post : P x T → IN est une application d'incidence arrière, telle que Post(p,t) contient la valeur entière associée à l'arc allant de la transition t à la place p;
  - M_0 : P → IN est le marquage initial qui donne la valeur initiale du nombre de jetons dans chaque place ;

- **W** = $(\lambda_1, \lambda_2, \ldots \lambda_m)$ est le vecteur des taux de franchissement associés aux transitions. (m=|T|).

Remarque : Le taux de tir ou de franchissement λ_i peut être dépendant du marquage.

Soit Mj un marquage, on écrit alors λ_i(Mj): le taux de tir associé à la transition t_i pour le marquage Mj. Donc, une transition donnée peut avoir des taux de franchissement qui diffèrent d'un marquage à l'autre. Ceci dépend de la sémantique de tir de la transition.

2.3. Notion de marquage [44]

L'évolution d'un réseau de Petri stochastique est définie par la notion de marquage.

- Un marquage M : $P \rightarrow |N$ d'un RDPS est une fonction décrivant un état observable du système.A chaque place est associé le nombre de jetons ou marques qu'elle contient. M est représenté par un vecteur à n éléments où $n=|P|$.
- Un marquage M est accessible depuis le marquage initial Mo si et seulement s'il existe une séquence de franchissements $s = t_1, \ldots, t_n$, après laquelle le marquage M est obtenu. On note Mo[s>M.

Pour un réseau de Petri stochastique, la suite des marquages (Mo,…Mn,…) obtenus à des instants de franchissements successifs (d_0,…d_n,…) constitue l'ensemble de ses états observables.

2.4. Processus de marquage [44,46]

Le marquage d'un RDPS à l'instant t est défini par : **$M(t)=[M(p_1,t),\ldots, M(p_n,t)]$**, où $M(p_i,t)$ est le marquage de la place p_i à l'instant t.

Pour une trajectoire $w = \{(X_o,t_o),\ldots,(X_n,t_n),\ldots\}$, le marquage atteint à l'instant t est défini comme suit :

$M(t,w)=[M(p_1,t,w),\ldots, M(p_n,t,w)]$ = $X_n(w)$, $\forall\ t \in [\ t_n\ ,\ t_{n+1}]$.

L'ensemble des marquages à l'instant t, pour w parcourant l'ensemble des trajectoires possibles, définit un vecteur aléatoire noté N(t). Ce vecteur est composé des variables aléatoires tel que $M(p_i,t)$ représentant le marquage de la place p_i à l'instant t.

La suite des vecteurs aléatoires N(t) constitue un processus aléatoire {M(t),t≥0} appelé : *processus de marquage.*

2.5. Temps moyen de franchissement d'une transition [44,46]

Le temps moyen de franchissement d'une transition t est une variable aléatoire de loi exponentielle, il est noté **Tmft** et est donné par la formule suivante :

$Tmft=1/\lambda_i(M_j)$.

Lorsque les taux de franchissement des transitions sont indépendants du marquage, la formule précédente devient :

$Tmft=1/\lambda_i$

Remarque : Lorsque plusieurs transitions sont franchissables, la transition qui provoque le changement d'état du système est celle qui a la plus petite durée de franchissement dans l'ensemble des transitions franchissables.

2.6. Règle de franchissement et graphe des marquages accessibles

2.6.1. Règle de franchissement [44,46]

Une transition t est franchissable dans un marquage M si et seulement si :

$\forall p \in P, M(p) \geq Pré(p,t)$.

Le marquage M' obtenu après le franchissement de t est calculé par la formule suivante :

$\forall p \in P, M'(p) = M(p) - Pré(p,t) + Post(p,t)$;

2.6.2. Graphe des marquages accessibles [44,46]

- Marquage accessible : Soit (R,Mo) un réseau marqué

Un marquage M est dit accessible à partir du marquage initial Mo, si et seulement si:

$\exists S \in T^*$ tel que :Mo[S>M (S étant une séquence de franchissement)

- Ensemble d'accessibilité :

Soit (R,Mo) un **RdP** marqué. L'ensemble d'accessibilité noté A(R,Mo) est l'ensemble des marquages atteints ou accessibles par une séquence de franchissement depuis Mo.

A(R,Mo)={ M / $\exists$ S $\in$ T* tel que :Mo[S>M}. Cet ensemble dit d'accessibilité décrit l'ensemble des configurations (états) possibles du système modélisé.

- Graphe des marquages accessibles (**GMA**) :

On appelle graphe des marquages accessibles **GMA**(R ,Mo), le graphe ayant A(R ,Mo) pour ensemble de sommets et dont les arcs sont définis comme suit :

Un arc joint M' à M'' ssi $\exists t \in T$: M' [t>M'' .On étiquette chaque arc de ce graphe par la transition correspondante au changement d'état du réseau..

2.7. Temps moyen de séjour dans un marquage [44,46]

Le temps moyen de séjour dans un marquage donné correspond à la durée de franchissement de la transition qui a provoqué le changement d'état.

La valeur moyenne de ce temps de séjour est donnée par la formule ci-dessous :

$$\mathbf{Tms = 1/\sum \lambda_i(M_j)}$$

$i : t_i \in E(M_j)$ où $E(M_j)$ est l'ensemble de toutes les transitions sensibilisées par le marquage Mj.

Lorsque les taux de franchissement des transitions sont indépendants du marquage, la formule précédente devient :

$$\mathbf{Tms = 1/\sum \lambda_I \quad telque \quad i : t_i \in E(M_j)}$$

2.8. Analyse des RDPS [44,46]

L'étude des performances d'un système donné est basée sur deux aspects essentiels. D'une part, nous avons l'aspect qualitatif qui repose sur la vérification des propriétés structurelles et d'autre part, nous avons l'aspect quantitatif qui a pour but le calcul des paramètres de performance du système modélisé.

2.8.1. Analyse qualitative [43, 44,46]

Dans cette partie, il s'agit de vérifier les propriétés qualitatives du système à analyser (la vivacité, la bornitude, états de blocage,…). Il est à noter que ces propriétés sont les mêmes et sont vérifiées de la même façon que pour les RDP simples grâce à l'identité du graphe des marquages accessibles d'un RDPS à celui du RDP sous-jacent.

Mais, la différence réside seulement dans le fait que les arcs du graphe sont valués par les taux correspondant aux transitions.

2.8.1.1. Bornitude d'un réseau

L'évolution d'un réseau est caractérisée par les différents états (marquages accessibles) que ce réseau peut prendre. Si l'ensemble des marquages accessibles possibles qu'admet le **RdP** est fini, il sera dit borné.

2.8.1.2. Place bornée

Une place p d'un réseau marqué <R,Mo> est dite k-bornée ($k \in IN$) si pour tout marquage accessible $M \in A(R,Mo)$, on a $M(p) \leq k$ .Dans le cas contraire, cette place est dite non bornée [46] .

2.8.1.3. Réseau borné

Un réseau marqué <R,Mo> est borné, si pour toute place p de ce réseau, il existe un entier k tel que p soit k-bornée.

2.8.1.4. La quasi-vivacité

Cette propriété désigne la possibilité de franchir au moins une fois toute transition du réseau.

- Quasi-vivacité d'une transition : Soit (R,Mo) un **RdP** marqué. Une transition t de ce réseau est dite quasi-vivante s' il existe un marquage accessible $M \in A(R,Mo)$ tel que M[t >.
- Quasi-vivacité d'un réseau : Un réseau de Petri <R, Mo> est quasi-vivant si :

$$\forall t \in T, \exists M \in A(R,Mo) \text{ tq : } M[t > .$$

Si le réseau n'est pas quasi-vivant, alors il existe une transition t qui n'est jamais franchissable et donc inutile au fonctionnement du système modélisé.

2.8.1.5. La pseudo-vivacité

Cette propriété caractérise le fait qu'à partir de tout marquage accessible, une transition au moins peut être tirée.

Un **RdP** (R,Mo) est pseudo- vivant si : $\forall M \in A(R,Mo)$, $\exists t \in T$ tq M[t >. [46]

2.8.1.6. La vivacité

Cette propriété est étroitement liée à la situation de blocage .Un **RdP** modélisant un système sans blocage doit être vivant. Ce qui implique que pour tous les marquages accessibles à partir de Mo, il est possible de tirer n'importe quelle transition en progressant le long d'une séquence de franchissement.

- Une transition t d'un réseau marqué (R,Mo) est vivante, si pour tout marquage accessible M appartenant à A(R,Mo) ,t est quasi vivante pour le réseau (R,M).
- Un **RdP** marqué (R,Mo) est vivant si toutes ses transitions sont vivantes .

Un réseau R est vivant s'il existe un marquage M tel que (R,M) soit vivant .
Autrement dit: $\forall M \in A(R,Mo), \forall t \in T, \exists S \in T^* tel$ que :M[S.t>**.**

2.8.1.7. Réseau sans blocage

La notion de réseau sans blocage est moins forte que celle du réseau vivant. Elle caractérise le fait qu'un réseau soit dans un état permanant de fonctionnement sans tenir compte des transitions franchies. Elle repose sur l'absence de marquage « puits ».

- Un marquage M d'un réseau (R, Mo) est appelé marquage puits si aucune transition n'est franchissable depuis M.
- Un réseau (R, Mo) est dit sans blocage si tout marquage accessible depuis Mo n'est pas un marquage « puits ».

2.8.2. Analyse quantitative

L'intérêt des RDPS réside particulièrement dans la nature du graphe des marquages qui est équivalent à une chaîne de Markov homogène. De ce fait, les méthodes de calcul des processus Markoviens deviennent applicables au calcul des paramètres de performances. On dit alors que le graphe des marquages d'un RDPS est isomorphe à une chaîne de Markov homogène [50].

L'analyse quantitative consiste à calculer les paramètres de performances. Ceci est possible grâce à l'isomorphisme entre les RDPS et les chaînes de Markov.

Nous pouvons donc évaluer les performances du système modélisé tout en appliquant les étapes suivantes [50] :

- Construire d'abord la matrice des taux de transitions.
- Ensuite résoudre le système matriciel afin de calculer les probabilités d'états en régime permanente.

2.8.2.1. Condition d'ergodicité

L'analyse quantitative à l'état d'équilibre nécessite la vérification de la condition d'ergodicité. Les modèles ergodiques sont les plus intéressants car cette propriété assure l'existence d'un régime stationnaire du système modélisé permettant le calcul d'une solution stationnaire [44,46].

Théorème

Un RDPS borné est ergodique, s'il admet l'état initial comme état d'accueil [45] .

2.8.2.2. Evaluation des performances d'un RDPS

Une fois l'ergodicité du modèle vérifiée, nous pouvons alors évaluer les performances du système modélisé en procédant de la manière suivante [42] :

- Construire la matrice des taux de transition Q, dite *générateur infinitésimal*.
- Calculer le vecteur des probabilités d'états stationnaires.
- Calculer des paramètres quantitatifs à l'aide du vecteur des probabilités stationnaires qui sont les suivants :
  - Fréquence moyenne de franchissement d'une transition.
  - Nombre moyen de marques dans une place.
  - Temps moyen de séjour des marques dans une place.
  - Probabilité d'un évènement A défini à travers une condition.

Ces quantités peuvent représenter respectivement dans le cas d'un exemple de RDPS modélisant un protocole de communication, le débit du canal de transmission, la charge du canal et le délai moyen de transmission.

2.8.2.3. Calcul du générateur infinitésimal [44]

Les éléments de la matrice Q sont calculés à l'aide du graphe des marquages accessibles. Ils correspondent aux taux de transition entre états ou marquages. La matrice est carrée, d'ordre égal au nombre de marquages dans le graphe d'accessibilité. Les taux de transition sont définis comme suit :

$$Q[i,j] = \begin{cases} \lambda_{ij} & \text{si } i \neq j \\ -\sum_{\substack{k=1 \\ k \neq i}}^{n} Q[i,k] & \text{si } i = j \end{cases}$$

Où : n correspond au nombre de marquages, et λ_{ij} désigne le taux de la transition du marquage M_i au marquage M_j.

Les autres éléments de la matrice sont nuls.

2.8.2.4. Calcul du vecteur des probabilités stationnaires [44]

Le vecteur π des probabilités stationnaires(en régime permanent), de dimension égale au nombre de marquages du système modélisé, est calculé en résolvant le système d'équations suivant :

$$\begin{cases} \pi Q = 0 \\ \sum_{1}^{n} \pi_i = 1 \end{cases}$$

2.8.3. Critères de performances [44, 46]

Le vecteur π des probabilités stationnaires décrit ci-dessus va nous aider dans le calcul des paramètres de performances comme suit :

2.8.3.1. Fréquence moyenne de franchissement d'une transition t_i

$$N^*(t_i) = \sum_{Mj \in GMA} \lambda_i(M_j).\pi_j$$

Où : $\lambda_i(M_j)$ est le taux de franchissement de la transition t_i dans le marquage M_j

2.8.3.2. Nombre moyen de marques dans une place p

$$M^*(p) = \sum_{Mj \in GMA} M_j(p).\pi_j$$

Où : $M_j(p)$ est le nombre de marques de la place p pour le marquage M_j.

2.8.3.3. Temps moyen de séjour des marques dans une place p

$$T^*(p) = M^*(p) / (Post(p, .) * N^*)$$

Où : N^* est le vecteur des fréquences moyennes de franchissement des transitions.

Post(p,.) est la ligne de la matrice d'incidence arrière Post correspondant à la place p.

$M^*(p)$ est le nombre moyen de marques dans la place p.

2.8.3.4. Probabilité d'un évènement A défini à travers une condition

$$Prob(A) = \sum \pi_i.$$

La somme est effectuée sur les indices des marquages où la condition est satisfaite.

Remarque

Il existe certains paramètres de performances qui sont dans le cas général difficile à calculer comme le calcul des distributions des délais de franchissement [43].

2.8.3.5. Exemple d'évaluation de performances d'un RDPS

Soit le réseau de Petri stochastique suivant :

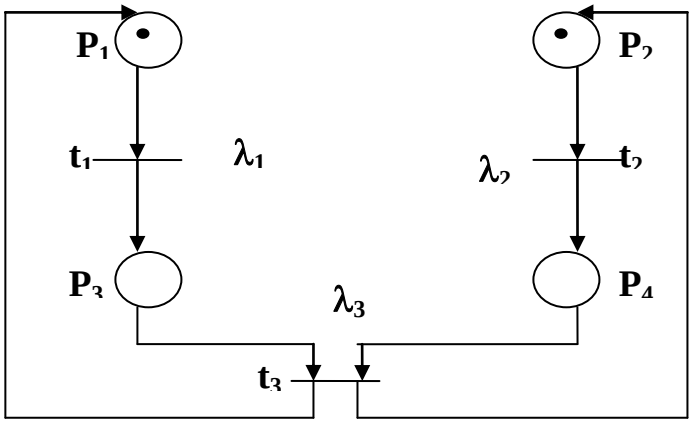


Figure 2.1 : Exemple d'un RDPS

Les transitions t1, t2, t3 ont respectivement comme taux λ1, λ2, λ3

Le graphe des marquages accessibles est constitué des marquages suivants :

$M_0 = [1100]$, $M_1 = [0110]$, $M_2 = [1001]$, $M_3 = [0011]$

Le graphe des marquages accessibles résultant est le suivant :

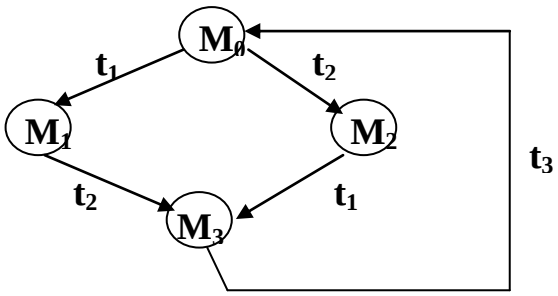


Figure 2.2 : Graphe d'accessibilité du RDPS

Le graphe d'accessibilité est borné, quasi-vivant (toutes les transitions sont franchissables) et est fortement connexe. Donc, d'après les résultats précédents, le réseau est ergodique. La matrice génératrice Q est donnée par :

$$Q = \begin{pmatrix} -(\lambda_1 + \lambda_2) & \lambda_1 & \lambda_2 & 0 \\ 0 & -\lambda_2 & 0 & \lambda_2 \\ 0 & 0 & -\lambda_1 & \lambda_1 \\ \lambda_3 & 0 & 0 & -\lambda_3 \end{pmatrix}$$

En résolvant le système d'équations suivant:

$$\begin{cases} \pi Q = 0 \\ \sum_{i=1}^{n} \pi_i = 1 \end{cases}$$

On trouve le vecteur des probabilités stationnaires suivant : $\pi = (\pi_1, \pi_2, \pi_3, \pi_4)$ où

$\pi_1 = (\lambda_1\lambda_2\lambda_3) /A$

$\pi_2 = (\lambda_1^2\lambda_3) /A$

$\pi_3 = (\lambda_2^2\lambda_3) /A$

$\pi_4 = (\lambda_1\lambda_2 (\lambda_1+\lambda_2)) /A$

Avec $A=\lambda_1\lambda_2\lambda_3 + \lambda_1^2\lambda_3 + \lambda_2^2\lambda_3 + \lambda_1\lambda_2(\lambda_1+\lambda_2)$

En appliquant les formules de calcul des paramètres de performances, on obtient :

Le vecteur des fréquences moyennes de franchissement de transition.

$$N^* = [\lambda_1(\pi_1 + \pi_3), \lambda_2(\pi_1 + \pi_2), \lambda_3\pi_4]$$

Le vecteur des nombres moyens de marques dans chaque place.

$$M^* = [\pi_1 + \pi_3,\ \pi_1 + \pi_2,\ \pi_2 + \pi_4,\ \pi_3 + \pi_4\ ]$$

Le vecteur des temps moyens de séjour des marques dans chaque place.

$$T^* = [\ 1/\lambda_1,\ 1/\lambda_2,\ 1/\lambda_3 + \lambda_1/(\lambda_2(\lambda_1+\lambda_2)),\ 1/\lambda_3 + \lambda_2/(\lambda_1(\lambda_1+\lambda_2))]$$

2.8.4. Les réseaux de Petri à arcs inhibiteurs :

Dans ce modèle qui permet de décrire certaines contraintes telles que le test à zéro par exemple, on a en plus des matrices d'incidence, une nouvelle matrice nommé matrice d'inhibition. Dans ce cas, pour permettre le franchissement d'une transition, le marquage d'une place doit être strictement inférieur au poids de l'arc reliant la place à la transition.

2.8.4.1. Définition [46]

Un RDP à arcs inhibiteurs est défini par le 5-uplet $R_z = < P, T, Pré, Post, Inh >$

1- $P = \{p_1, p_2, \dots, p_n\}$ est un ensemble fini de places, pour lequel $n = |P|$;

2- $T = \{t_1, t_2, \dots, t_m\}$ est un ensemble de transitions, pour lequel $m = |T|$;

3- $Pré: P \times T \rightarrow IN$ est la fonction d'incidence avant, telle que $Pré\ (p,t)$ contient la valeur entière associée à l'arc allant de la place p à la transition t ;

4- $Post: P \times T \rightarrow IN$ est la fonction d'incidence arrière, telle que $Post\ (p,t)$ contienne la valeur entière associée à l'arc allant de la transition t à la place p;

5- $Inh: P \times T \rightarrow (IN/0)$, est la fonction d'inhibition, $Inh(P,t)$ représente le poids de l'arc inhibiteur allant de la place p à la transition t.

Un arc inhibiteur (graphiquement) est un arc dans lequel l'extrémité incidente aux transitions est représentée par un petit cercle.

Exemple :

Voici un exemple comportant un arc inhibiteur :

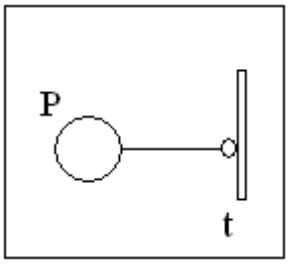


Figure 2.3 : Arc inhibiteur.

2.8.4.2. Franchissement dans un réseau à arcs inhibiteurs [46]

Soit un marquage M d'un RDP à arcs inhibiteurs et t une transition :

- t est franchissable en M si et seulement si :

$$\forall p \in P, M(p) \geq Pré(p,t) \text{ et } M(p) < Inh(p,t).$$

Ce franchissement à partir de M conduit à un nouveau marquage M' défini comme suit :

$$\forall p \in P, M'(p) = M(p) - Pré(p,t) + Post(p,t).$$

2.9. Réseaux de Petri stochastiques généralisés (RDPSG)

Le modèle des RDPSG est apparu afin de résoudre les problèmes liés à la présence de divers types d'événements dans un même modèle, tels que les activités qui ne nécessitent pas de temporisation et d'autres qui sont très rapides (urgentes) ou très lentes. Le modèle des RDPSG prend en considération la nature de ces activités, afin que le modèle soit logiquement correct.

Ainsi, ce modèle est caractérisé par l'introduction d'un nouveau type de transitions, appelées transitions *immédiates***.** Par conséquent, un RDPSG est un RDPS dont les transitions sont de deux types :

3. **Transitions temporisées** : à qui correspondent les variables aléatoires déterminant la durée de franchissement.
4. **Transitions immédiates (instantanées)** : qui se caractérisent par une période de franchissement supérieure à celle des transitions temporisées, et par leur franchissement immédiat, car le taux de franchissement associé est infini, ainsi le délai de tir correspondant à ces transitions est nul.

Définition [43]

Un RDPSG est un triplet <R_0, π, W>, où :

- R_0 est un RDP simple.
- π est la fonction de priorité associant un entier à toute transition.
- W est une fonction qui associe à chaque transition temporisée un taux de franchissement et à chaque transition immédiate un poids.

2.9.1. Evolution d'un RDPSG

Les marquages accessibles dans un RDPS se décomposent en deux catégories :

- **Marquages tangibles** : Dans lesquels, aucune transition immédiate n'est franchissable.

Si tous les marquages sont tangibles, alors le processus stochastique engendré par un RDPSG est identique au RDPS.

- **Marquages évanescents** : Dans lesquels, il y a au moins une transition immédiate parmi les transitions franchissables. Dans ce cas, la priorité est donnée aux transitions immédiates car leur franchissement se fait en un temps nul. D'autre part, si un marquage évanescent sensibilise plusieurs transitions immédiates à la fois, le choix de celle qui sera franchissable ne peut se baser sur des considérations temporelles, puisque la durée de leur franchissement est nulle, mais il se fait par une post-sélection. Cette post-sélection, de nature probabiliste, est basée sur les poids des transitions donnée par l'expression suivante :

$$\mathbf{P\{t_k\} = W_k / \sum W_i} \quad \text{telque :}$$

W_k est le poids de la transition t_k et la somme est faite sur l'ensemble des transitions immédiates sensibilisées par le marquage M.

Le marquage obtenu par le franchissement d'une transition t, est calculé de la même manière qu'un RDPS:

M'(. , t) = M(. , t) + C(. , t) ; tel que **C(. , t) = -Pré(. , t) +Post (. , t) ;**

2.9.2. Représentation graphique d'un RDPSG

Les RDPSG sont constitués de transitions temporisées stochastiques représentées par des rectangles : ▭ et de transitions immédiates représentées par des barres noires fines : ▬

2.9.3. Propriétés des RDPSG [48]

Nous pourrons examiner le RDP sous-jacent pour déterminer les propriétés du RDPSG. Autrement dit, toutes les transitions qu'elles soient temporisées ou immédiates, sont traitées de la même façon lors de l'analyse qualitative.

Ces propriétés constituent donc l'évaluation qualitative du système correspondant.

Exemple2 : soit un RDPSG contenant six places, une transition immédiate (t5) et quatre transitions temporisées (t1, t2, t3, t4) ayant pour taux de franchissement respectifs (λ_1, λ_2, λ_3, λ_4). Les marquages accessibles sont les suivants :

M0 = (110010)

M1 = (011000)

M2 = (100100)

M3 = (110001)

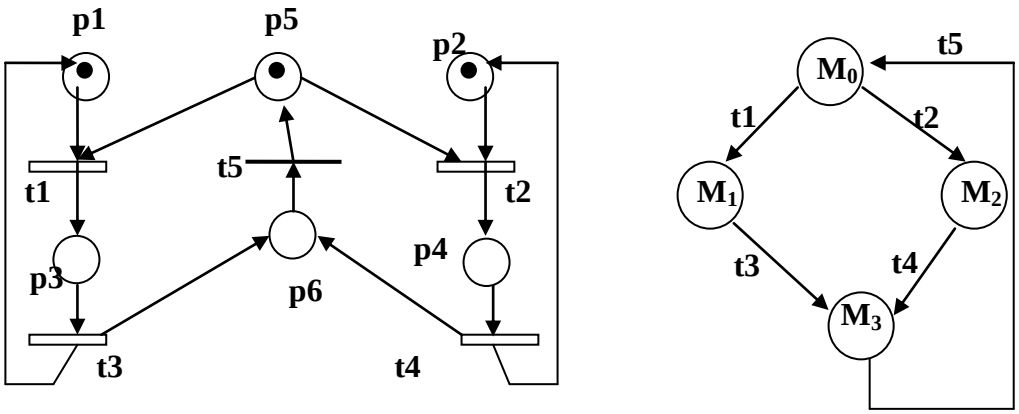


Figure 2.4 : Exemple d'un RDPSG et son graphe d'accessibilité

Fusionner les marquages évanescents avec les marquages tangibles qui les succèdent, nous donne donc le graphe d'accessibilité réduit constitué des marquages tangibles uniquement qui sera utilisé pour l'analyse quantitative.

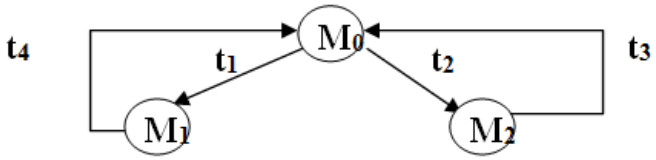


Figure 2.5 : graphe d'accessibilité réduit

Ainsi, le générateur infinitésimal est donné par la matrice suivante :

	M0	M1	M2	M3
P1	1	0	1	1
P2	1	1	0	1
P3	0	1	0	0
P4	0	0	1	0
P5	1	0	0	0
P6	0	0	0	1

- Bornitude : Le réseau est 1- borné, il est dit sauf (sain)
- Pseudo-vivacité : D'après le GMA du RDP précédent, nous avons : M0[t1> ; M0[t2> ; M1[t3> ; M2[t4> ; d'où la condition de pseudo-vivacité est vérifiée.

- Quasi-vivacité : du marquage initial, nous pouvons franchir la séquence t1.t3.t5.t2.t4 où apparaît chaque transition du réseau, d'où le réseau est quasi-vivant.
- Vivacité : de n'importe quel marquage A(R, M0) nous pouvons franchir une séquence où apparaît chaque transition du réseau, le réseau est donc dit vivant.
- Existence d'un état d'accueil : de n'importe quel marquage accessible, tous les marquages sont atteignables, donc il existe un état d'accueil. De plus M0 est un état d'accueil. Donc, le système est réinialisable.
- Réseau sans blocage : Il n'existe pas de marquage puits, donc le réseau est sans blocage.
- L'ergodicité : le RDPSG est borné et réinialisable, donc il est ergodique.

Etant donné que le RDPSG est ergodique nous pouvons alors évaluer les performances du système modélisé à l'état stationnaire.

2.9.5. Evaluation des indices de performances [49]

Un RDPSG reste isomorphe à une chaîne de Markov à temps continu. Pour évaluer les paramètres de performances d'un RDPSG, nous suivons les étapes suivantes :

- Construire le GMA comme dans le cas d'un RDPS.
- Valuer ce graphe par des taux de franchissement exponentiels lorsqu'il s'agit de transitions temporisées et par des taux infinis dans le cas de transitions immédiates.

Fusionner les marquages évanescents avec ceux tangibles qui les succèdent.

On obtient ainsi un nouveau graphe dit réduit qui reste isomorphe à une chaîne de Markov à temps continu : C'est la **chaîne de Markov réduite**. Ceci n'influe pas sur les paramètres quantitatifs à évaluer, car le système modélisé par ce RDPSG passe un temps nul dans les marquages évanescents.

- A partir de cette chaîne réduite ainsi obtenue, on construit la matrice des taux de transitions entre marquages tangibles qui sert de base pour le calcul des paramètres de performances du système à analyser.

Il convient cependant de remarquer que ce graphe réduit ne permet pas d'évaluer qualitativement le système à analyser, car son GMA n'est pas identique à ce graphe réduit. Ainsi, pour une analyse qualitative, il faudrait plutôt examiner le GMA avec marquages tangibles et évanescents

Conclusion

Les modèles des RDP permettent de décrire des systèmes parallèles de différentes manières. Le constat que nous avons fait dans ce chapitre est que, la notion d'un RDPS est basée sur celle d'un RDP simple. Mais seulement dans les RDPS, à chaque transition est associée un délai de franchissement aléatoire. Le facteur temps est alors pris en compte.

Le RDPS nous permet de construire un graphe des marquages accessibles qui est isomorphe à une chaîne de Markov à temps continu. Les paramètres de performances du RDPS peuvent alors être calculés sous condition d'ergodicité de la chaîne de Markov à temps continu. D'où les RDPS permettent la vérification et l'évaluation du système modélisé.

Dans les systèmes modélisés nous avons la présence d'activités rapides et d'autre plus lentes. La négligence de cet aspect peut nous donner des modèles logiquement incorrects.

Le modèle de RDPSG a alors été introduit, comme complément aux RDPS aux quel on y ajoute des transitions immédiates, qui prennent en compte les activités rapides du système, et auxquelles aucune temporisation ne peut être associée.

Dans le chapitre qui suit, nous nous intéresserons à l'utilisation de ce modèle formel dans les réseaux de capteurs sans fil.

Chapitre 3 : Modélisation d'un réseau de capteurs sans fil à l'aide des RDPSG

Introduction

Les réseaux de capteurs sans fil sont des réseaux de communication avec des contraintes difficiles. Ils ont besoin de légèreté, efficacité énergétique et de protocoles d'auto-organisation.

Durant ces deux dernières décennies, un effort considérable a été consacré par beaucoup de chercheurs à l'évaluation des performances des systèmes avec phénomène de rappel, et plus précisément aux files d'attente avec rappel, qui est le modèle conventionnel habituellement appliqué pour l'analyse des performances de ces systèmes. L'intérêt porté à ce thème est principalement expliqué par les développements et les avancés technologiques, notamment dans les domaines de l'informatique et des télécommunications.

En fait, il est d'une importance basique d'étudier la fiabilité des réseaux de capteurs sans fil où le trafic de communication peut être considéré comme un système avec rappel et où les capteurs sont sujets à des pannes et des réparations aléatoires, et ce à cause de la forte influence des pannes sur les performances du réseau. En effet, un réseau de capteurs sans fil nécessite une qualité de service d'un certain niveau même en cas de panne. Ainsi, une importance particulière devrait être accordée à l'étude d'un réseau qui combine la présence simultanée du phénomène de rappel et de la non-fiabilité des capteurs.

D'autre part, les RDPSG constituent un important modèle graphique et mathématique, qui offre une grande puissance descriptive, permettant la modélisation et l'analyse des systèmes qui sont caractérisés par le fait d'être parallèles, distribués et stochastiques. La possibilité d'inclure facilement les aspects de synchronisation, de concurrence et représenter très aisément le blocage ainsi que différents phénomènes stochastiques dans un même modèle, est la principale caractéristique qui les a rendus très populaires.

En effet, depuis ces dernières décennies, les RDPSG présentent un centre d'intérêt pour de nombreuses recherches dans le domaine de la fiabilité et des performances. Ce formalisme de modélisation de haut niveau a été largement appliqué pour décrire le comportement dynamique des systèmes concurrents et asynchrones et aussi pour étudier le comportement qualitatif et quantitatif de ces systèmes parallèles.

Ainsi, nous présentons dans ce chapitre, la modélisation et l'évaluation des performances des réseaux de capteurs sans fils à l'aide des RDPSG. Cette approche nous permet de décrire d'une manière simple et précise différents réseaux avec capteurs fiables ou capteurs non fiables. Dans ce cadre, nous considérons les différentes disciplines de panne définies dans la littérature des modèles avec rappel :

- Les pannes actives qui se produisent lorsque le serveur (le capteur) et en cours de service ;
- Les pannes indépendantes qui se produisent indépendamment du l'état du serveur, oisif ou occupé, mais avec la même probabilité, ce qui constitue une contrainte assez forte ;
- Les pannes dépendantes : Dans ce cas, les taux de panne des serveurs actifs (pannes actives) et les taux de panne des serveurs oisifs (pannes oisives), peuvent être égaux ou différents [50, 54].

D'autre part, cette approche nous offre la possibilité d'effectuer une analyse qualitative aussi bien que quantitative d'un réseau de capteurs sans fil, en se basant sur les méthodes et les outils très avancés développés pour l'évaluation des performances des RDPSG.

Dans la première section de ce chapitre, nous présentons une brève description générale des systèmes avec rappel et un exemple d'un réseau de capteurs sans fil pour illustrer le trafic de communication entre les nœuds capteurs. Dans la section qui suit, nous présentons le modèle de RDPSG décrivant le trafic dans un réseau sans fil avec capteurs fiables. Dans une autre section, nous nous intéressons particulièrement aux réseaux de capteurs où les capteurs sont sujets à des pannes aléatoires (non fiables). Nous proposons dans ce cadre, un modèle pour chaque discipline de panne et nous développons ensuite les formules des indices de performance correspondants.

3.1. Description générale des modèles avec rappel

La plupart des travaux qui portent sur les systèmes avec phénomène de rappel considèrent le modèle des files d'attente avec rappel (FAR) [50]. Dans ce modèle, à l'arrivée d'un client qui trouve les serveurs occupés, il rejoint l'orbite (espace d'attente imaginaire), pour rappeler plus tard pour le service à des intervalles de temps suivant une loi de probabilité. Quand le client est en orbite il peut rappeler pour le service avec la probabilité H_K pour la k-ième fois, ou quitter le système sans être servi avec la probabilité $1 - H_K$. La Figure 3.1 montre la structure générale des FAR:

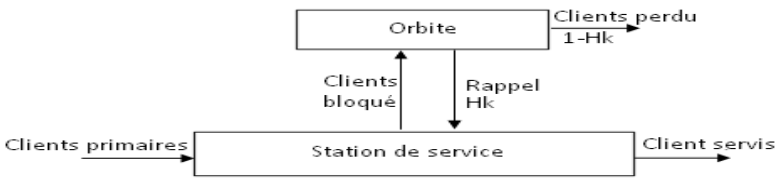


Figure 3.1 : Structure générale d'un modèle avec rappel

3.2. Etude de cas d'un réseau de capteurs

Un exemple de scénario d'un réseau de capteurs sans fil est donné dans la figure 3.2. Les nœuds capteurs (présentés par des cercles) sont déployés dans une région à deux dimensions (x, y). Les nœuds sont étiquetés par leur distance par rapport au nœud collecteur (le nœud *puits* ou bien *Sink* en anglais) mesurée par le nombre de sauts de la communication. Le nœud collecteur (représenté par le cercle noir et plein) est situé aux coordonnées (7, 5).

En raison des contraintes de ressources difficiles et la portée de transmission limitée, chaque nœud i, situé aux cordonnées (x_i, y_i), est seulement capable de communiquer directement avec ses voisins immédiats, c'est à dire avec tous les nœuds j, où $| x_j - x_i | \leq 1$ et $| y_j - y_i | \leq 2$. Par exemple, le nœud situé à (6, 4) est capable d'échanger directement des messages avec les nœuds situés à (6, 6), (5, 5), (5, 3), (6, 2), (7, 3), et aussi . avec le nœud collecteur. Nous supposons que chaque nœud est conscient de sa propre distance au sink mesuré en nombre de sauts.

Le but du réseau de capteurs donné est de surveiller la zone couverte, enregistrer les incidents, et de communiquer avec le nœud collecteur lors de l'apparition d'un incident dans un mode multi-saut. Dans un système réel, les incidents pourraient être, par exemple, la reconnaissance d'une intrusion, les températures ou l'humidité dépassant certains seuils prédéfinis, ou encore la détection d'incendie, de gaz, de vibration, de mouvement, de bruit, etc. Nous supposons que chaque nœud peut détecter plusieurs incidents, mais un nombre fini d'incidents distincts.

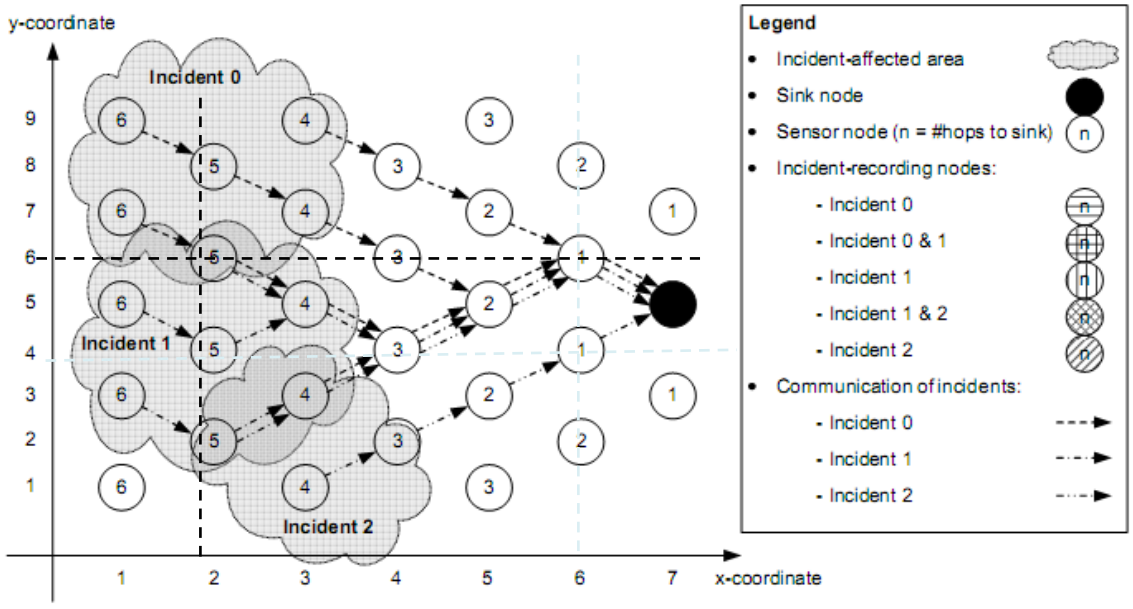


Figure 3.2 : Exemple d'un réseau de capteurs

Dans l'exemple de scénario esquissé dans la Figure 3.2, trois incidents (incidents 0, 1 et 2) peuvent être détectés par le réseau de capteurs. Par exemple, le nœud (2, 6) détecte les incidents 0 et 1, et génère immédiatement un message pour chaque incident, et tente d'envoyer deux messages vers le sink. Le nœud (2, 6) a six voisins dont deux sont plus proches au sink: le nœud (3, 7) et le nœud (3, 5).

En principe, tous les voisins peuvent être au courant de la transmission, ce qui est dû à la propriété de diffusion dans l'air de l'interface réseau sans fil. Ainsi, chaque voisin peut servir comme un nœud de prochain saut. Cependant, les nœuds sont auto-organisés et pour économiser de l'énergie, un nœud peut refuser de recevoir de nouveaux messages, de stocker les messages, ou d'envoyer des messages selon son état, sa distance par rapport au nœud collecteur, etc.

A la réception d'un message, si le message est accepté, le nœud récepteur envoie un accusé de réception et prend soin de l'envoyer plus loin. Par contre, si aucun des voisins du nœud émetteur acceptent le message, ce nœud stocke le message localement, ensuite il fera d'autres tentatives pour le transmettre plus tard. Si dans la période d'attente, le nœud émetteur reçoit en outre des rapports du même incident, il fusionne les messages. Ainsi, chaque incident est enregistré qu'une seule fois au niveau du nœud.

Dès que le message a bien été transféré vers le nœud de prochain saut, ce dernier prend soin de l'acheminer vers le nœud collecteur. Les messages qui atteignent le sink quittent le réseau de capteurs.

3.3. Les réseaux de capteurs fiables

Dans cette section, nous considérons les réseaux de capteurs fiables i.e. que le risque de panne de tous les nœuds est nul. Ce type de réseau peut être vu comme un système d'attente avec rappel, complètement markovien à source d'évènements finie de taille N et à serveurs identiques, parallèles et fiables.

Dans notre modèle:

- La source des messages primaires représente les incidents;
- Le groupe de serveurs représente les voisins;
- La file d'attente représente les messages stockés dans le buffer en attendant leur transmission;
- L'orbite représente les messages bloqués ou retardés temporairement en attendant leur retransmission par l'émetteur;

3.3.1. Modélisation d'un réseau à capteurs fiables

La figure 3.3 représente une modélisation possible du trafic dans un réseau à capteurs fiables, qui peut être vu comme un système multi serveurs avec appels répétés, source finie et buffer à capacité limitée. Cette modélisation est générique, dans le sens où la capacité de la source est représentée par le paramètre entier positif « *N* » qui apparaît comme marquage initial de la place *Msg_libres* et la taille du buffer d'un capteur est limitée et représentée par le paramètre entier positif « *d* ».

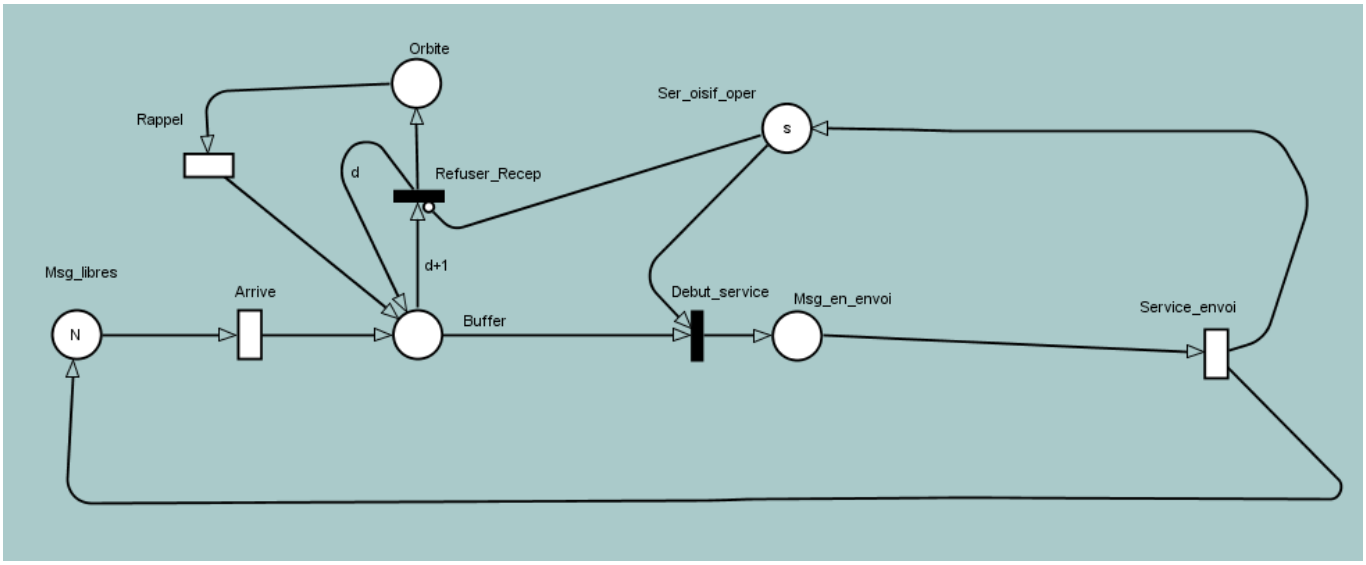


Figure 3.3. Le RDPSG modélisant le trafic dans un réseau à capteurs fiables

Dans ce modèle, nous avons une source d'évènements de taille finie égale à *N* modélisée par la place *Msg_libres,* tel que chaque jeton représente dans la réalité : un message non encore envoyé d'un nœud voisin ou bien un incident non encore détecté et ce type de message est dit message primaire.

Ces messages primaires arrivent à un nœud capteur suivant une distribution exponentielle de taux λ.

Les messages entrent par la suite dans la place *Buffer* à partir de laquelle le message est envoyé si un voisin est disponible (opérationnel et oisif). Cela signifie que le nœud capteur essaie d'envoyer les nouveaux messages dès leur réception ou bien leur détection, à un nœud voisin de premier saut.

Les nœuds voisins d'un capteur donné, correspondent au groupe de serveurs tel que chaque nœud voisin opérationnel et oisif, est représenté par un jeton dans la place *Ser_oisif_oper*. Si un nœud voisin ne peut pas recevoir des messages (son buffer est plein), on dit qu'il est occupé. Si le nœud est en mode économie d'énergie par exemple, on dit que le nœud est inactif (non opérationnel) sinon il est oisif.

Seuls les capteurs voisins qui sont oisifs et opérationnels peuvent accepter des messages. On ne considère ici, que les capteurs actifs et fiables. Si tous les voisins sont occupés, les messages restent en attente localement dans le nœud émetteur (place *Buffer*). Si le nombre de messages dépasse la capacité du buffer, les nouvelles requêtes sont orientées vers l'orbite pour être retransmises par l'émetteur après un délai aléatoire.

Notons que nous considérons le cas de modèle persistant où des tentatives d'envoi sont répétées jusqu'à la transmission effective du message. Nous considérons dans la prochaine section, le cas où les messages sont perdus après un certain nombre de rappels. Ce modèle est dit : Modèle à clients impatients.

Les tableaux suivants résument les caractéristiques de chaque composante du modèle :

Place	***Signification***	***Marquage initial***
Msg_libres	*Les incidents non encore détectés où les messages non encore acceptés*	*N*
Buffer	*Représente le buffer du capteur dont la taille égale à « d »*	0
Msg_en_envoi	*Les messages en cours de transmission*	0
Ser_oper_oisif	*Les capteurs voisins oisifs et opérationnels*	s
Orbite	*Contient les requêtes des messages retardés temporairement en attendant leur retransmission par l'émetteur*	0

Tableau 3.1 : Places du modèle décrivant un WSN avec capteurs fiables

Transition	***Signification***	***type***	***Taux de tir***
Arrive	*Arrivée d'un message où d'une requête*	*Temporisée à serveurs infinis*	λ
Debut_service	*Début d'envoi d'un message*	*Immédiate*	-
Rfuser_Recep	*Mettre le message en orbite (sauvegarde interne) pour une retransmission ultérieur*	*Immédiate*	-
Service_envoi	*L'envoi d'un message*	*Temporisée à serveurs infinis*	μ
Rappel	*Rappel d'un message pour la transmission*	*Temporisée à serveurs infinis*	ν

Tableau 3.2 : Transitions du modèle décrivant un WSN avec capteurs fiables

Le marquage initial du réseau est défini par:
M_0={M(*Msg_libres*),M*(Buffer*),M(*Msg_en_envoi*),M(*Ser_oisif_oper*),M(*Orbite*),}={*N,0,0,s,0*}, ce qui signifie qu'aucun message n'est initialement reçu, que tous les voisins (serveurs) sont disponibles et que l'orbite et le buffer sont vides.

Le franchissement de la transition *Arrive* représente la détection d'un message primaire correspondant à un nouvel incident détecté par le capteur lui-même ou bien un message transmis par un capteur voisin. Ainsi, la place *Buffer* reçoit un jeton, ce qui représente la condition qu'un message est prêt pour la transmission à un capteur voisin. La place *Buffer* a une taille limitée (*d*) car dans la réalité le capteur a une capacité mémoire limitée. Si le buffer du capteur est plein, le capteur refuse de recevoir de nouveaux messages. La sémantique de service de cette transition peut être une *sémantique à serveurs infinis* si le capteur peut recevoir plusieurs messages en parallèle. Nous considérons la sémantique à serveurs infinis, ceci signifie que tous les incidents libres peuvent générer des messages primaires en plus des messages transmis par les capteurs voisins.

À l'arrivée d'un message à la place *Buffer*, si la place *Ser_oisif_oper* contient au moins un jeton qui correspond à un voisin oisif et opérationnel, la transition immédiate *Debut_service* sera tirée instantanément, et le marquage de la place *Msg_en_envoi* sera incrémenté de 1. Ceci représente le fait qu'un message a commencé son envoi et qu'un capteur voisin est passé de l'état oisif à l'état occupé. Par contre, si la place *Ser_oisif_oper* est vide (aucun voisin n'est oisif et opérationnel), le message entre dans la file d'attente jusqu'à qu'il soit servi. Si le buffer est plein (nombre maximum de jetons dans la place *Buffer* est égal à *d*), c'est plutôt la transition immédiate *Refuser_Recep* qui sera tirée et par conséquent le message bloqué rejoint l'orbite. Dans le cas où au capteur dont le buffer est plein et certain voisins sont disponibles, la transition D*ebut_Service* qui sera franchie et non pas la transition *Refuser_Recep,* car la l'arc inhibiteur allant de la place *Ser_oisif_oper* à la transition *Debut_Service* donne à cette dernière une priorité absolue. Une fois en orbite, les requêtes comportent indépendamment les uns des autres, et chacun d'eux rappelle pour le service après un délai de temps distribué exponentiellement avec un taux *v.* Ainsi, la sémantique de la transition *Rappel* doit être à serveurs infinis, i.e. que le taux de franchissement est dépend du marquage de la place *Orbite.* Dès le tir de cette transition temporisée, un jeton est déposé dans la place *Buffer*. Ceci matérialise l'évènement de rappel d'un message de l'orbite. Ainsi, plusieurs processus de l'orbite peuvent rappeler pour le service simultanément. En fait, un taux de rappel constant, dans ce cas, signifie qu'un seul processus peut rappeler à la fois pour le service. Cette politique de rappel constant peut être facilement modélisée avec une transition *Rappel* à sémantique à serveur unique.

D'autre part, à la fin d'une période d'envoi, un jeton sera déposé dans la place *Ser_oisif_oper* pour représenter le voisin qui passe de l'état occupé à l'état oisif, et un autre jeton représentant le message qui

devient libre est déposé dans la place *Msg_libres.* Le franchissement de la transition *Service_envoi* est aussi à sémantique à serveurs infinis si le capteur peut envoyer plusieurs messages en parallèle et tous les capteurs peuvent travailler simultanément.

3.3.2. Modélisation d'un réseau à capteurs fiables avec perte de messages

Dans cette section, nous considérons que les messages peuvent être perdus après un certain nombre de rappels. Ce modèle est dit : Modèle à clients impatients. La perte d'un message peut être due à une panne ou bien à un abandon de la requête d'un capteur émetteur qui a réussi à envoyer son message par un autre chemin.

Le marquage initial du réseau est :

M_0={M(*Msg_libres*),M(*Buffer*),M(*Msg_en_envoi*),M(*Ser_oisif_oper*),M(*Orbite*)}={*N,0,0,s,0*}, ce qui signifie qu'aucun message ou incident n'est détecté, que tous les voisins sont disponibles et que l'orbite et la file d'attente sont vides .

Un message qui trouve le buffer plein, peut entrer dans l'orbite avec une probabilité égale à p_1 (poids de la transition *Refuser_Recep*) ou bien quitter le service avec une probabilité p_2.

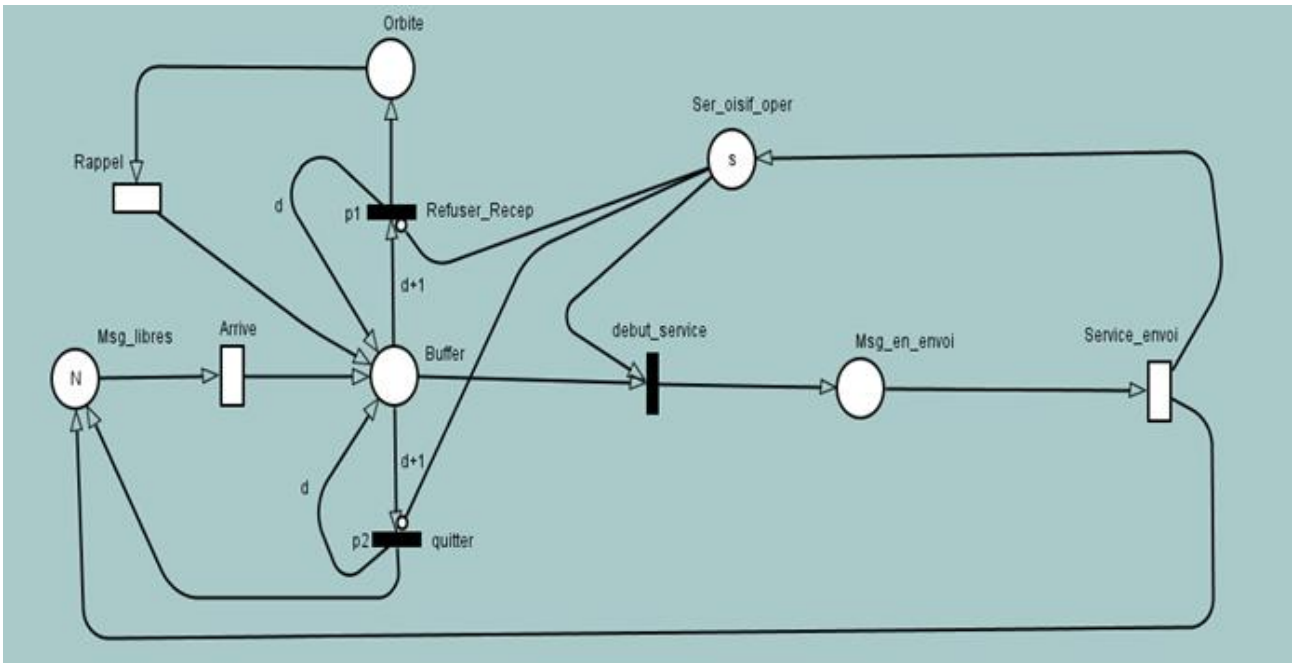


Figure 3.4 : Le RDPSG modélisant le trafic de communication dans un WSN avec possibilité de perte de messages

Le tableau suivant résume les caractéristiques de la transition *quitter* :

Transition	***Signification***	***type***	***Taux de franchissement***
quitter	*Abandon d'un message*	*Immédiate avec un poids égal à p2*	-

Tableau 3.3 : Propriétés de la transition *quitter*

Le modèle de la figure 3.4 peut être présenté d'une autre façon en supprimant les arcs inhibiteurs et en associant un poids « *q* » à la transition *Debut_service* avec : *q >> p1* et *p2.* La figure 3.5 montre le modèle modifié.

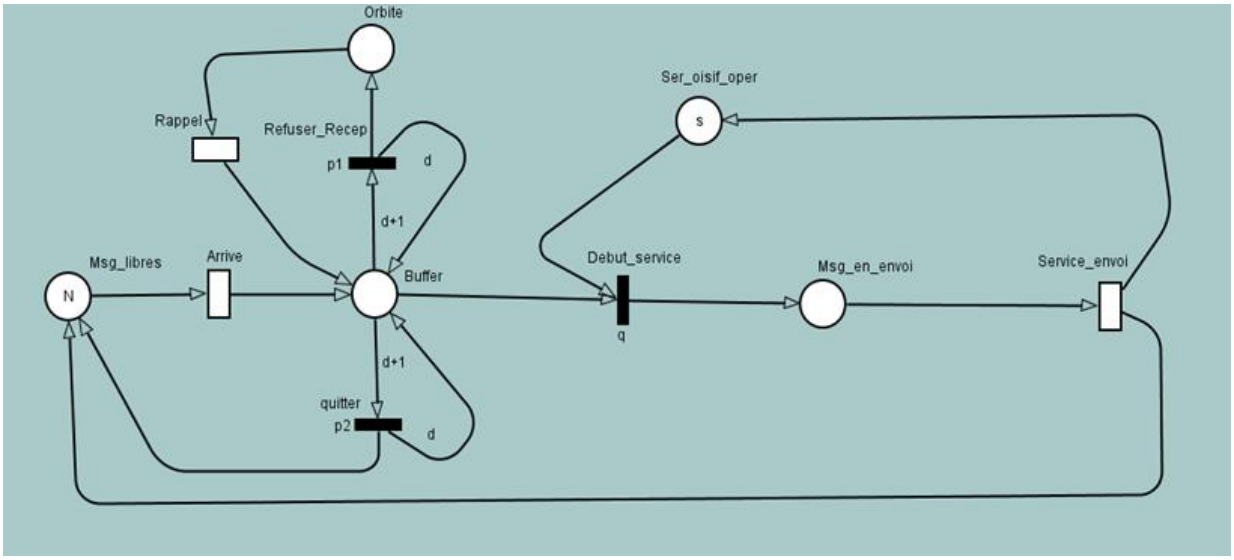


Figure 3.5 : Le RDPSG modélisant le trafic de communication dans un WSN avec possibilité de perte de messages sans arcs inhibiteurs

3.3.3 Analyse d'un réseau à capteurs fiables

Une fois que le modèle de RDPSG est construit, son analyse consiste d'une part à définir ses propriétés qualitatives (comportementales et structurelles) telles que la vivacité, l'absence de blocage, etc, et d'autre part, à calculer ses paramètres de performance. En fait, l'analyse quantitative n'a de sens que si une analyse qualitative a été préalablement menée. Il est en effet inutile de vouloir obtenir les performances d'un système qui se trouve dans un état de blocage par exemple, car les techniques d'évaluation de performances pourront très bien calculer des paramètres sans se rendre compte de l'éventualité d'un tel blocage. Dans ce cas, ces techniques fourniront, donc, des résultats erronés.

D'autre part, l'évaluation des performances à l'état stationnaire nécessite l'ergodicité du modèle, ce qui revient à vérifier certaines propriétés qualitatives. Nous considérons le modèle de RDPSG de la figure 3.5. Le modèle est borné, vivant, sans blocage et le marquage initial est un état d'accueil. Par conséquent ce modèle admet un état stationnaire. Nous notons par : $\pi=(\pi_1, \pi_2,\ldots, \pi_n)$ la distribution des probabilités de marquage à l'état stationnaire où π_i est la probabilité que le processus est à l'état M_i et $M_i(p)$ est le nombre de jetons de la place p dans le marquage M_i. Nous notons par A l'ensemble des marquages tangibles accessibles et par A(t) l'ensemble des marquages tangibles accessibles dans lesquels la transition t est franchissable.

L'évaluation des performances s'intéresse au calcul des paramètres de performance d'un système. Parmi les paramètres les plus importants, que l'on souhaite évaluer pour un réseau de capteurs sans fil, nous citons :

- *le débit ;*
- *le nombre moyen d'une entité donnée ;*
- *le temps d'attente ;*
- *le temps de réponse ;*
- *le taux d'utilisation des ressources ;*

En ayant les probabilités d'état stationnaire π, divers indices de performance intéressants peuvent être calculés. Nous donnons dans ce qui suit, les formules explicites de calcul des paramètres du réseau en nous basant sur le modèle de RDPSG. Nous nous intéressons particulièrement, au calcul des valeurs moyennes de ces paramètres, car ces valeurs ont un intérêt statistique.

- **Le nombre moyen de capteurs voisins occupés (n_s) :**

Il correspond au nombre moyen de jetons dans la place *Msg_en_envoi,* ce qui représente aussi le nombre moyen de messages en cours d'envoi.

$$n_S = \sum_{i:Mi\epsilon A} M_i(Msg\_en\_envoi).\pi i$$

- **Le nombre moyen de capteurs voisins oisifs et opérationnels (n_d) :**

Il correspond au nombre moyen de jetons dans la place *Ser_oisif_oper*

$$n_d = \sum_{i:Mi\epsilon A} M_i(Ser\_oisif\_oper).\pi i$$

- **Le nombre moyen de messages dans le buffer du capteur (n_b) :**

Il correspond au nombre moyen de jetons dans la place *Buffer*

$$n_b = \sum_{i:Mi \epsilon A} M_i(Buffer).\pi i$$

- **Le nombre moyen de messages retardés (n_o) :**

Il correspond au nombre moyen de jetons dans la place *Orbite*

$$n_O = \sum_{i:Mi \epsilon A} M_i(Orbite).\pi i$$

- **Le nombre moyen de messages primaires (n_a) :**

Il correspond au nombre moyen de jetons dans la place *Msg_libres*

$$n_a = \sum_{i:Mi \epsilon A} M_i(Msg\_libres).\pi i$$

- **Le nombre moyen de messages dans le système (n) :**

Il correspond au nombre moyen de messages en envoi, en attente ou en orbite

$$n = n_O + n_b + n_S$$

- **Le taux moyen de génération des messges primaires (λ') :**

Il correspond à la fréquence (débit) de la transition *arrive*

$$\lambda' = \sum_{i:Mi \epsilon A(Arrive)} \boldsymbol{\lambda}.M_i(Msg\_libres).\pi i$$

- **Le taux moyen de génération des messages répétés (ν') :**

Il correspond à la fréquence (débit) de rappel des messages en orbite.

$$\nu' = \sum_{i:Mi \epsilon A(Rappel)} \boldsymbol{\nu}.M_i(Orbite).\pi i$$

- **Le taux moyen de perte d'un message de l'orbite (ν_q') :**

Il correspond à la fréquence (débit) d'abandonner l'orbite.

$$\nu_q' = \sum_{i:Mi \epsilon A(quitter)} \boldsymbol{\nu_q}.M_i(Orbite).\pi i$$

- **Le taux moyen d'envoi desmessages (μ') :**

Il correspond à la fréquence (débit) de la transition *Service_envoi.*

$$\mu' = \sum_{i:Mi \epsilon A(Service\_envoi)} \mu . Mi\,(Msg\_en\_service).\pi i$$

- **L'utilisation de *c* voisins (**Uc**) :**

Ceci correspond à la probabilité que *c* voisins soient occupés

$$Uc = \sum_{i:Mi(Msg\_en\_envoi) \geq c} \pi i$$

- **La disponibilité de *c* voisins (**Ac**) :**

Ceci correspond à la probabilité que *c* voisins sont oisifs et opérationnels

$$Ac = \sum_{i:Mi(Ser\_oisif\_oper) \geq c} \pi i$$

- **Le temps d'attente moyen (W') :**

Le temps d'attente d'un message correspond à la durée de temps entre l'arrivée du message et son début d'envoi. Le temps d'attente moyen W' à l'état stationnaire, peut être facilement obtenu à l'aide de la formule de *Little* :

$$W' = (n_b + n_o)/\lambda'$$

- **Le temps de réponse moyen (R') :**

$$R' = n/\lambda'$$

- **La probabilité de blocage d'un message primaire (**B_p**) :**

$$B_p = \frac{\sum_{i:Mi \epsilon A} \sum_{i=1}^{N-s} i.\lambda.\ Prob\,[Mj(Msg_{libres}) = i\ \&\ Mj(Buffer) = d]}{\lambda'}$$

- **La probabilité de blocage d'un ancien message (de l'orbite),** (**B_r**) :**

$$B_r = \frac{\sum_{i:Mi \epsilon A} \sum_{i=1}^{N-s} i.\nu.\ Prob\,[Mj(Orbite) = i\ \&\ Mj(Buffer) = d]}{\nu'}$$

- **La probabilité de blocage des messages (**B**) :**

$$B = B_p + B_r$$

3.4. Modélisation des réseaux de capteurs sans fil non fiables

En pratique, certains composants des capteurs (batterie par exemple) pourraient être sujets à des pannes aléatoires, il est donc d'une importance basique de prendre en considération la fiabilité des capteurs lors du processus de modélisation. Ces modèles présentés dans la figure 3.3, 3.4 et 3.5 ne prennent pas en considération les pannes des capteurs qui causent des changements dans les indices de performances du système. Une panne peut être une situation d'économie d'énergie ou bien un mauvais fonctionnement dans les composants du capteur lui même.

Dans cette section, nous allons présenter la modélisation ainsi que l'analyse des performances et de fiabilité des réseaux de capteurs sans fil avec différentes disciplines de panne, à l'aide du modèle RDPSG.

3.4.1. Description mathématique

Un nœud voisin, peut être occupé ou oisif et d'autre part, peut être opérationnel ou en panne. Les messages peuvent être dans l'un des quatre états : libre, en service, en attente ou en orbite. Les messages libres génèrent des requêtes (ou appels) primaires suivant un processus aléatoire de taux λ .Ainsi, à l'arrivé d'un message (primaire ou répété), si un des nœuds voisins est opérationnel et oisif, alors ce message peut être transmis immédiatement suivant une loi exponentielle de taux μ, et le capteur récepteur passe à l'état occupé ; autrement, si tous les nœuds voisins sont occupés ou en panne, le message s'enregistre localement et on dit qu'il est en attente. Si le buffer est plein, le capteur refuse la réception et on dit que le message est en orbite à partir duquel il envoie des appels répétés distribués exponentiellement de taux v, jusqu'à qu'il soit accepté (file non pleine) pour être transmis ou bien quitte l'orbite avant dans le cas d'un modèle non persistant. A l'émission d'un message, le capteur devient oisif et le message devient libre. D'autre part, les messages sont adressés aux voisins opérationnels oisifs d'une manière aléatoire et sans aucun ordre de priorité. Autrement dit, le but du capteur est la transmission d'un message donné vers l'un des capteurs voisins sans aucun choix particulier. Cependant, nous pouvons considérer lors d'une analyse d'un protocole particulier que s est le nombre des capteurs voisins choisis selon un critère prédéfini et qui sont capables d'acheminer le message vers le sink.

Un nœud capteur peut tomber en panne avec un taux γ lorsqu'il entrain d'envoyer un message ou bien avec un taux δ lorsqu'il oisif. Dans la littérature, trois politiques de panne ont été considérées pour des modèles avec rappel mono-serveur ou multiserveurs [50,57, 58, 59] que nous avons appliqués aux réseaux de capteurs. Ces politiques sont les suivantes :

- La politique des pannes actives, où $\delta = 0$ *et* $\gamma > 0$;
- La politique de pannes indépendantes, où $\delta = \gamma > 0$;

- La politique de panne dépendantes, où $\delta > 0$ *et* $\gamma > 0$;

Dans la première politique [57, 58], le capteur ne peut tomber en panne qu'en étant actif. Par contre, dans la politique de pannes indépendantes proposée [57], les capteurs peuvent tomber en panne indépendamment de leur état, et la probabilité de panne est la même que le capteur soit oisif ou occupé. Dans les systèmes réels, ces taux peuvent être identiques ou différents. Dans un article [56], Gharbi a introduit une nouvelle politique appelée politique de panne dépendante. Dans ce cas, la probabilité de panne d'un capteur dépend de son état. Ainsi, les taux $\delta > 0$ *et* $\gamma > 0$ peuvent être égaux ou différents. Par conséquent, cette discipline est une généralisation de la discipline de panne indépendante [50].

D'autre part, si un capteur tombe en panne au cours de la transmission, le message entre en orbite et rappellera ultérieurement pour le service, avec une remise à zéro du travail réalisé avant que la panne ne survienne, on parle dans ce cas de panne à service répété jusqu'au premier niveau. Le temps de réparation d'un capteur quelconque est distribué exponentiellement avec un taux donné. Nous supposons que le réparateur suit la discipline FIFO pour réparer les capteurs en panne et après la réparation, le capteur est aussi bon qu'avant de tomber en panne.

En fait, il est important de préciser que quand les temps de franchissement sont distribués exponentiellement et les mesures de performances auxquelles on s'intéresse sont les indices moyens, les différentes disciplines de service des files d'attente (*FIFO*, *LIFO* ou *aléatoire*) donnent les mêmes résultats [50]. Par conséquent, l'ordre aléatoire qui est la politique par défaut des RDP peut être appliqué et l'analyse donnera les résultats espérés.

Par ailleurs, si tous les capteurs voisins sont en panne, deux cas différents peuvent être envisagés :

- Cas de sources bloqués, où toutes les arrivées primaires, les appels répétés et les opérations de transmission sont arrêtés.
- Cas de sources non bloquées, où seulement la transmission est interrompue et toutes les opérations de réception continuent, autrement dit, les nouveaux appels et les appels répétés peuvent être générés indépendamment de l'état des capteurs.

3.4.2. Modélisation des WSN avec pannes actives et sources non-bloquées

La figure 3.6 décrit le modèle RDPSG correspondant aux réseaux de capteurs où un nœud capteur a plusieurs voisins (multi serveurs) avec pannes actives et sources non-bloquées.

- La place *Ser_oisif_oper* représente les voisins du nœud qui sont oisifs et opérationnels ;
- La place *Ser_en_panne* contient les capteurs voisins en panne ;

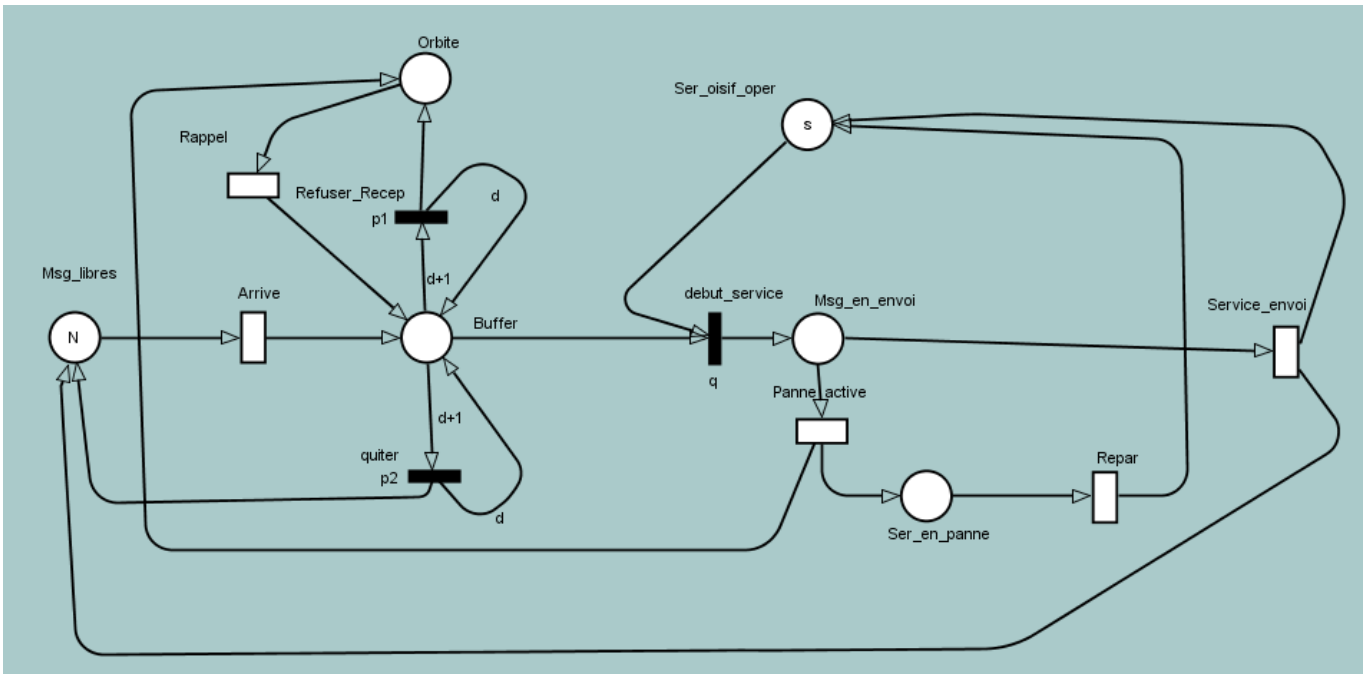


Figure 3.6 : RDPSG modélisant le trafic dans un WSN avec pannes actives et sources non-bloquées

Le marquage initial du réseau est :

M_0= M{ (*Msg_libres*), M(*Buffer*), M(*Msg_en_envoi*), M(*Ser_oisif_oper*),M(*Orbite*), M(*Ser_en_panne*)} ={*N,0,0,s,0,0*}, ce qui signifie que tous les messages sont initialement libres, que tous les *s* voisins sont oisifs et opérationnels et que le buffer et l'orbite sont vides.

Le message en envoi (représenté par un jeton dans la place *Msg_en_envoi*), restera jusqu'à ce que l'une des deux transitions *Service_envoi* ou *Panne_active* soit tirée. Comme le tir est déterminé par deux distributions exponentielles indépendantes, nous ne pouvons pas dire à un instant particulier, laquelle des deux transitions sera tirée en premier. Ceci coïncide avec la réalité de la situation, car quand le capteur commence à recevoir un message, on ne sait pas s'il tombera en panne avant la fin ou pas.

Si le capteur tombe en panne avant qu'il ne finisse sa tâche, ce qui correspond au franchissement de la transition *Panne_active* avant *Service_envoi*, le message qui se trouvait dans la place *Msg_en_envoi* qui représente le message en cours d'envoi, sera consommé et deux jetons seront produits, dont l'un représente le message interrompu après une remise à zéro du travail déjà réalisé par le capteur voisin avant la panne. Le second jeton représente le capteur en panne qui rejoint la place *Ser_en_panne* où il sera réparé ou remplacé. D'autre part, la transition *Panne_active* modélise les pannes des capteurs opérationnels.

Par ailleurs, le franchissement de la transition *Repar* représente la fin du temps de réparation et le fait qu'un capteur réparé retourne à l'état opérationnel oisif i.e. à la place *Ser_oisif_oper*.

Le tableau 3.5 résume les propriétés de la transition *Panne_active* et *Repar.*

Transition	***Signification***	***type***	***Taux de franchissement***
Panne_active	*Panne d'un capteur occupé*	*Temporisée à serveurs infinis*	γ
Repar	*Réparation des capteurs*	*Temporisée à serveurs infinis*	τ

Tableau 3.4 : Propriétés des transitions en rapport avec la panne.

3.4.3. Modélisation des WSN avec pannes dépendantes et sources non-bloquées

Dans le modèle précédent, les capteurs peuvent tomber en panne en étant occupés seulement, ce qui est représenté par la transition *Panne_active*. A présent, nous considérons la politique des pannes dépendantes [56], où un capteur peut tomber en panne durant l'intervalle (t, t+dt) avec une probabilité : $\delta\ dt+o(dt)$ s'il est oisif, et une probabilité : $\gamma\ dt+o(dt)$ s'il est occupé. Les taux de panne $\delta>0$ et $\gamma>0$ peuvent être égaux ou différents. Ceci signifie que la politique des pannes indépendantes où $\delta= \gamma >0$ est considérée comme un cas particulier.

Donc, pour mettre en œuvre la politique des pannes dépendantes, il suffit d'introduire deux transitions séparées *Panne_active* pour les pannes actives et *Panne_oisif* pour les pannes oisives. Le modèle correspondant à un réseau de capteurs avec ce type de pannes est donné dans la Figure 3.5

Dans ce modèle, la place *Msg_en_envoi* contient les messages en cours d'envoi (ou bien les voisins occupés). Durant une période de service, un capteur peut tomber en panne. Ceci est représenté par

le tir de la transition *panne_active* . Dans ce cas, le message interrompu entre en orbite (place *Orbite*) et le capteur en panne rejoint la place *Ser_en_panne* où il sera réparé.

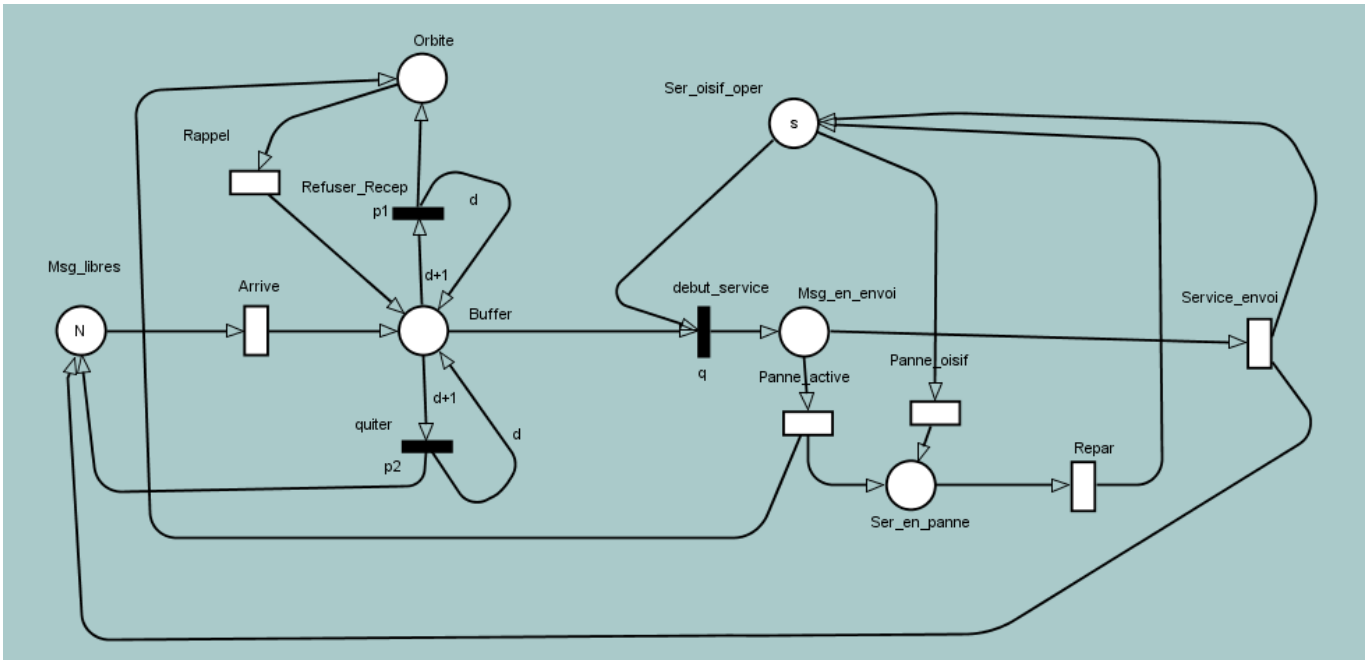


Figure 3.7: RDPSG modélisant le trafic dans un WSN avec pannes dépendantes et sources non-bloquées

Sous la discipline de pannes dépendantes, un capteur peut aussi tomber en panne en étant oisif (dans la place *Ser_oisif_oper*), ce qui correspond au franchissement de la transition *panne_oisif*. La sémantique du franchissement de cette transition est aussi à serveurs infinis, car plusieurs capteurs voisins oisifs peuvent tomber en panne en même temps.

Les transitions *Panne_active* et *Panne_oisif* représentant les pannes actives et les pannes oisives respectivement, peuvent avoir les mêmes taux ou des taux différents (politique des pannes indépendantes et dépendantes respectivement).

Le Tableau 3.6 résume les caractéristiques de la transition *Panne_oisif :*

Transition	***Signification***	***type***	***Taux de franchissement***
Panne_oisif	*Panne d'un capteur oisif*	*Temporisée à serveurs infinis*	δ

Tableau 3.5 : Caractéristiques de la transition *Panne_oisif*

3.4.4. Modélisation des WSNs avec sources bloquées

Dans les figures 3.6 et 3.7, nous avons considéré un réseau de capteurs avec sources non-bloquées. Ceci signifie que si tous les voisins d'un capteur donné sont en panne, seule l'émission est interrompue par contre la réception continue normalement. En fait, il est parfois préférable de bloquer la réception des messages sur un capteur dont tous les voisins qui sont censés acheminer les messages, sont en panne, car leur réception nécessitera leur stockage si possible en attendant la réception des capteurs, ce qui peut retarder les messages. Pour cela, nous considérons un réseau de capteurs avec sources bloqués, où toute réception de message est rejetée et le capteur lui-même est considéré non disponible.

Pour modéliser cette politique, il suffit d'ajouter aux modèles 3.6 et 3.7, un arc inhibiteur de multiplicité s, allant de la place *Ser_en_panne* à la transition *Arrive* et un autre à la transition *Rappel*. Ainsi, les appels primaires et les appels répétés peuvent arriver au système seulement quand le marquage de la place *Ser_en_panne* est inférieur à « *s* », i.e. quand le nombre de voisins non opérationnels est inférieur à « *s* ». Autrement, si tous les capteurs voisins sont en panne, aucune des deux transitions : *Arrive* pour les nouveaux messages et *Rappel* pour les messages en orbite n'est franchissable, et par conséquent, toutes les arrivées sont interrompues.

Les figures 3.8 et 3.9 suivantes, représentent la modélisation par RDPSG d'un réseau de capteurs avec pannes actives et pannes dépendantes respectivement où la source est bloquée.

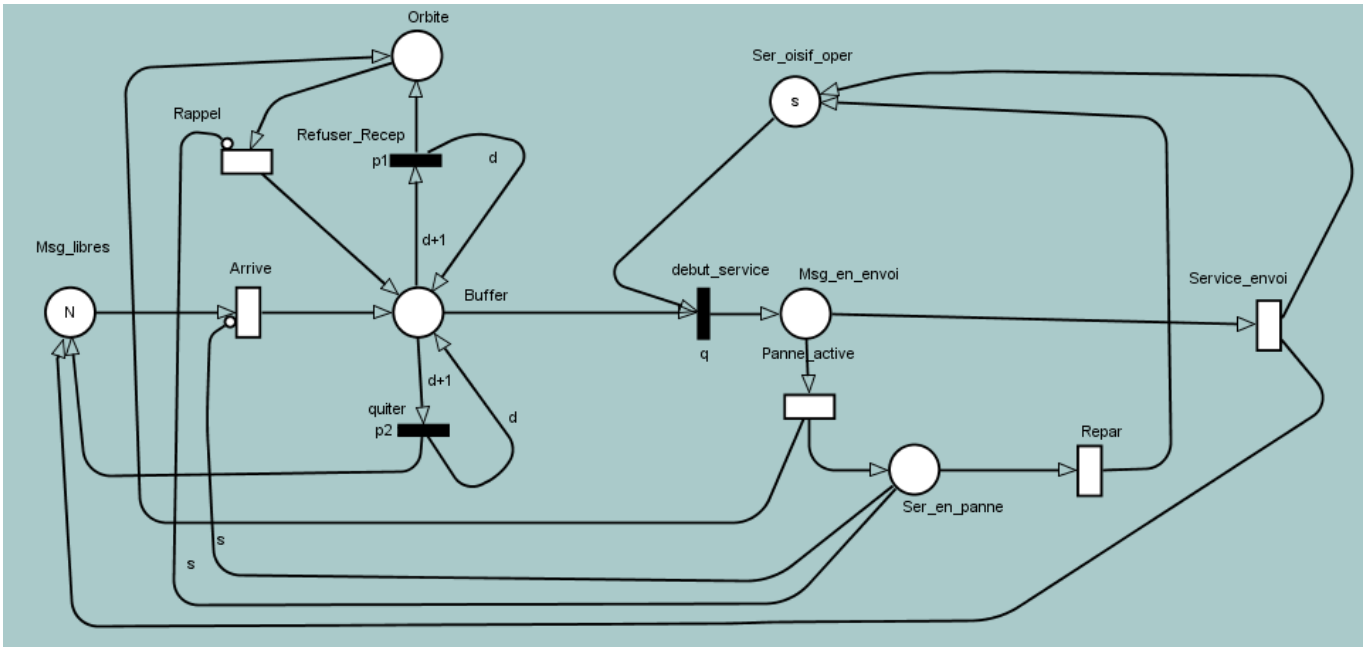


Figure 3.8 : RDPSG modélisant le trafic dans un WSN avec pannes actives et sources bloquées

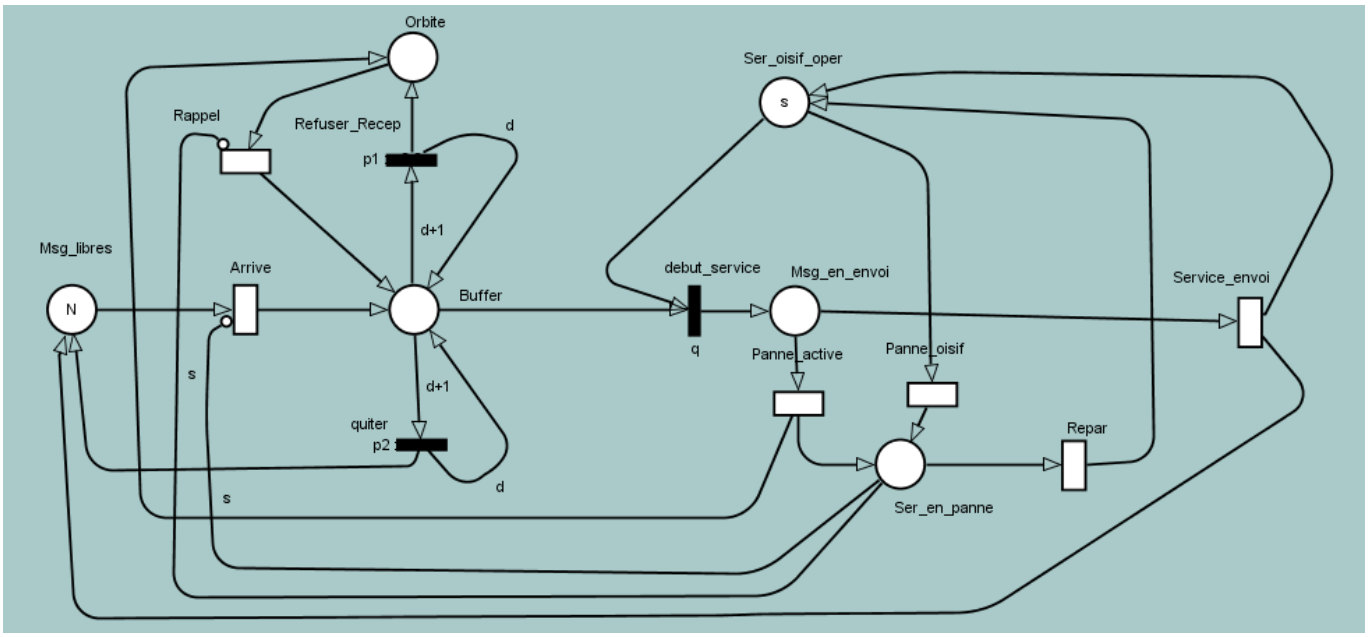


Figure 3.9: RDPSG modélisant le trafic dans un WSN avec pannes dépendantes et sources bloquées

3.4.5. Modélisation des WSN avec pannes dépendantes, sources non-bloquées et mise en veille des capteurs

La technologie actuelle des réseaux de capteurs suppose que les nœuds de capteurs sont alimentés par des batteries. Economiser l'énergie est donc une préoccupation omniprésente dans toutes les conceptions de protocoles MAC et routage pour les réseaux de capteurs sans fil. Mettre en veille prolongée la partie transmission/réception radio est sans doute le moyen le plus efficace car avec les composants actuels, elle est la partie qui consomme le plus d'énergie [62]. Suivant cette approche, une famille de protocoles MAC à faible cycle actif (duty-cycle) a été conçue. Deux états sont associés à la partie radio d'un nœud : active et endormie (ou en veille). Un cycle, d'une période fixe ou variable, est composé d'une période active durant laquelle un nœud peut transmettre et recevoir, et une période de sommeil où la partie radio est éteinte. Le cycle actif (duty-cycle) est défini comme la proportion de la période active sur la durée totale d'un cycle (période active + période de sommeil). Par rapport à un réseau dont les nœuds sont actifs en permanence, il est clair qu'un réseau fonctionnant avec un faible, voire très faible cycle actif (low duty-cycle et ultra-low duty-cycle) permet d'économiser l'énergie et par conséquent prolonger la durée de vie du réseau en la multipliant approximativement par l'inverse de sa valeur du cycle actif [63].

Dans cette optique, les protocoles d'ordonnancement d'activité permettent aux capteurs de passer du mode actif, dans lequel ils participent à la vie du réseau, au mode veille. Les décisions de changement

d'état peuvent être prises par une entité centrale avec une connaissance globale du réseau, ou par les nœuds eux-mêmes qui se basent alors uniquement sur des informations de voisinage.

Un capteur oisif peut tomber en panne à cause d'un défaut technique ou bien d'une expiration complète de l'énergie. En effet, un capteur en veille n'est pas en panne car il peut passer à l'état actif. Donc, il est nécessaire de différencier une panne d'un état de veille car la durée d'une panne peut être différente de celle de l'état de veille i.e. le passage de l'état de veille à l'état actif est plus rapide qu'une réparation d'un capteur en panne. En plus, la transition inter états actif et veille est prédéfinie généralement par le protocole de communication.

Du point de vue modélisation, nous allons introduire dans le modèle de la figure 3.7 deux nouvelles transitions temporisées, une pour la mise en veille et une autre pour la mise en état actif. Nous allons introduire aussi une place contenant des jetons correspondent. La figure 3.8 illustre le modèle de RDPSG correspondant.

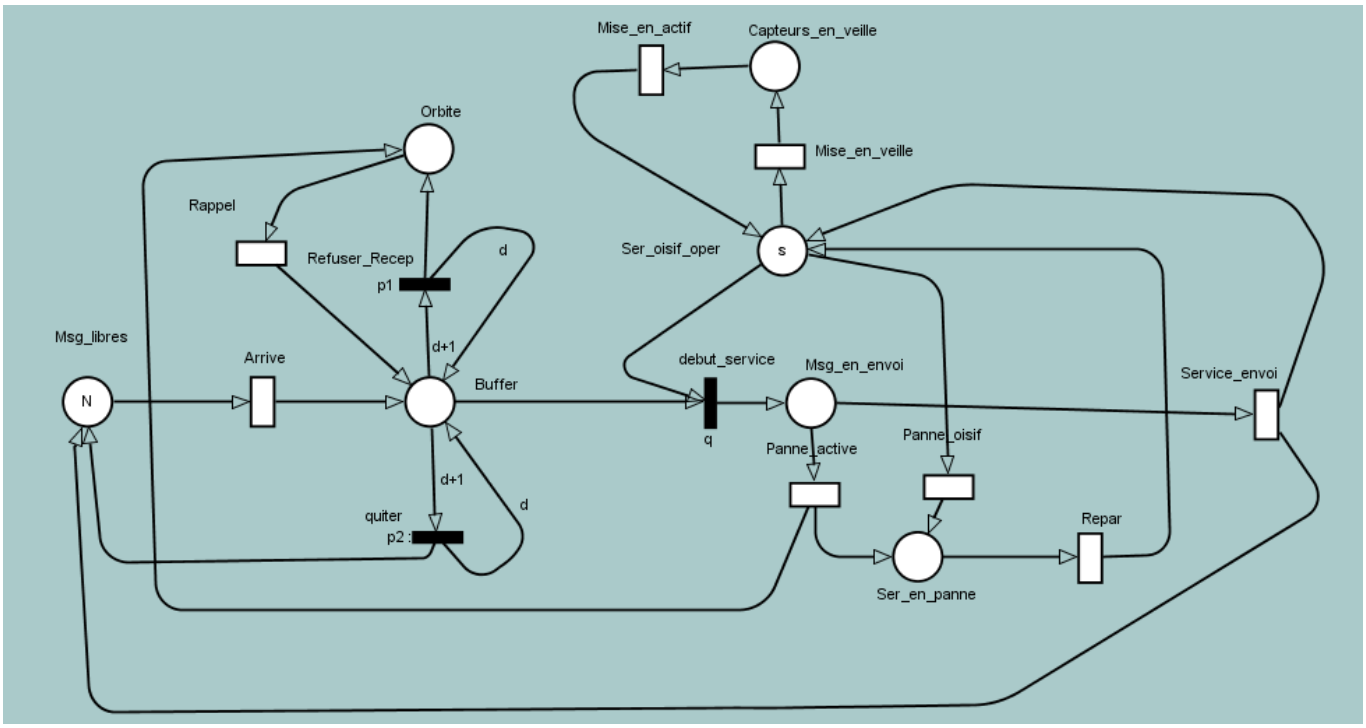


Figure 3.10: RDPSG modélisant le trafic dans un WSN avec mise veille des capteurs

Dans ce modèle, la place *Capteurs_en_veille* représente les capteurs en état de veille. Selon une stratégie du protocole utilisé, le passage de l'état actif à l'état de veille suit une loi exponentielle avec un taux δ_v qui correspond au taux de la transition *Mise_en_veille*. D'autre part, une période de temps de

moyenne $1/\tau_v$ tel que τ_v représente le taux de franchissement de la transition *Mise_en_actif* , un capteur en état de veille s'active et cela est représenté par le tir de la transition *Mise_en_actif*.

Le tableau 3.6 résume les caractéristiques des transitions *Mise_en_veille et Mise_en_actif.*

Transition	***Signification***	***type***	***Taux de franchissement***
Mise_en_veille	*Le franchissement de cette transition représente le passage d'un capteur de l'état actif à l'état de veille.*	*Temporisée à serveurs infinis*	δ_v
Mise_en_actif	*Le franchissement de cette transition représente le passage d'un capteur de l'état de veille à l'état actif.*	*Temporisée à serveurs infinis*	τ_v

Tableau 3.6 : Propriétés des transitions *Mise_en_veille et Mise_en_actif*

Le tableau 3.7 résume les propriétés de la place *Capteurs_en_veille.*

Place	***Signification***	***Marquage initial***
Capteurs_en_veille	*Chaque jeton dans cette place représente un capteur en état de veille.*	0

Tableau 3.7 : Propriétés de la place *Capteurs_en_veille*

Nous notons ici que dans le cas de sources bloquées, seule la transition de réparation des capteurs en panne et la transition d'activation des capteurs en état de veille se franchissent. Pour ce faire, nous ajoutons un arc inhibiteur de multiplicité s entre la place *Ser_en_panne* et la transition *Arrive* et un autre arc entre la place *Ser_en_panne* et la transition *Rappel.* Cependant, on néglige le cas où il y a aucun capteur voisin oisif et opérationnel et on a des capteurs voisins libres mais en veille car le temps du veille n'est pas long par rapport au temps d'une réparation.

3.4.6. Paramètres de performance et de fiabilité

Les RDPSG proposés pour la modélisation du trafic dans un réseau de capteurs sans fil avec les différentes politiques de panne sont bornés et le réinitialisales, ce qui garantit leur ergodicité. Par conséquent, ces modèles admettent une distribution stationnaire π.

En ayant ces probabilités d'état stationnaire, plusieurs paramètres de performance et indices de fiabilité intéressants peuvent être calculés en appliquant les formules explicites définies précédemment.

Pour éviter toute répétition, nous donnons dans ce qui suit uniquement les indices de performance et de fiabilité ayant rapport avec les phénomènes de panne et de réparation. Les paramètres dans la section concernant le cas fiable restent valables pour le cas non fiable.

- **Le nombre moyen de capteurs voisins en état de veille (n_v) :**

Il correspond au nombre moyen de jetons dans la place *Capteurs_en_veille*

$$n_v = \sum_{i:Mi \epsilon A} Mi\,(Capteurs\_en\_veille).\pi i$$

- **Le nombre moyen de capteurs voisins en panne (n_p) :**

Il correspond au nombre moyen de jetons dans la place *Ser_en_panne*

$$n_p = \sum_{i:Mi \epsilon A} Mi\,(Ser\_en\_panne).\pi i = S - (n_d + n_s + n_v)$$

- **La fréquence de panne des capteurs voisins occupés (γ') :**

Elle correspond à la fréquence (débit) de la transition *Panne_active*

$$\gamma' = \sum_{i:Mi \epsilon A(Panne\_active)} \lambda . Mi\,(Msg\_en\_envoi).\pi i$$

- **La fréquence de panne des capteurs voisins oisifs (δ') :**

Elle correspond au débit de la transition *Panne_oisif. Notons que* ce paramètre n'a de sens qu'en cas de politique de pannes dépendantes ou indépendantes

$$\delta' = \sum_{i:Mi \epsilon A(Panne\_oisif)} \lambda . Mi\,(Ser\_oisif\_oper).\pi i$$

- **La fréquence de mise en veille des capteurs voisins oisifs (δ_v') :**

Elle correspond au débit de la transition *Mise_en_veille*

$$\delta_v' = \sum_{i:Mi \epsilon A(Mise\_en\_veille)} \lambda . Mi\,(Ser\_oisif\_oper).\pi i$$

- **Le taux moyen de réparation des capteurs (τ') :**

Elle correspond à la fréquence de la transition *Repar*

$$\tau' = \sum_{i:Mi \epsilon A(Repar)} \lambda . Mi\,(Ser\_en\_panne).\pi i$$

- **Le taux moyen d'activation des capteurs en état de veille (τ_v') :**

Elle correspond à la fréquence de la transition *Mise_en_actif*

$$\tau_V' = \sum_{i:Mi \epsilon A(Mise\_en\_actif)} \lambda . Mi\,(Capteurs\_en\_veille).\pi i$$

- **L'utilisation de *c capteurs* voisins (**Uc**) :**

Ceci correspond à la probabilité que *c* voisins soient occupés où $0 \leq < c \leq s$ inclus *;*

$$Uc = \sum_{i:Mi(Msg\_en\_envoi) \geq c} \pi i$$

- **La disponibilité de *c capteurs* voisins (**Ac**) :**

Ceci correspond à la probabilité que *c* voisins soient opérationnels et oisifs où $0 \leq < c \leq s$ *;*

$$Ac = \sum_{i:Mi(Ser\_oisif\_oper) \geq c} \pi i$$

- **La probabilité de panne de *c* voisins (**Fc**) :**

Ceci correspond à la probabilité que *c* voisins soient en panne où $0 \leq < c \leq s$ *;*

$$Fc = \sum_{i:Mi(Ser\_en\_panne) \geq c} \pi i$$

- **La probabilité de Mise en état de veille de *c capteurs* voisins (**Fv**) :**

Ceci correspond à la probabilité que *c* capteurs soient en état de veille où $0 \leq < c \leq s$ *;*

$$Fv = \sum_{i:Mi(Capteurs\_en\_veille) \geq c} \pi i$$

- **L'utilisation du réparateur (**Ur**) :**

Ceci correspond à la probabilité de panne d'un capteur voisin au moins ;

$$Ur = \sum_{i:Mi(Ser\_en\_panne) \geq 1} \pi i$$

Conclusion

L'objet de ce chapitre a été la présentation d'une approche de modélisation et d'évaluation des performances et de la fiabilité d'un réseau de capteurs sans fil à l'aide du modèle des RDPSG. Un tel réseau peut être vu comme un système avec rappel, mémoire à capacité limitée, source finie et serveurs fiables. ou non fiables, à l'aide du modèle de RDPSG

Les RDPSG constituent un modèle graphique de haut niveau, permettant la description des systèmes complexes caractérisés par la synchronisation, le parallélisme et la temporisation aléatoire.

A travers cette étude, nous pouvons voir que la puissance d'expression de ce modèle, nous a permis de modéliser aisément un réseau de capteurs avec possibilité de rappel tout en considérant la non-fiabilité des capteurs, la possibilité de perte de messages et la taille limitée du buffer. D'autre part, différentes disciplines de panne ainsi que la mise en veille qui permet un gain considérable de l'énergie, ont été modélisées, ce qui permet de faire une analyse plus détaillée.

Dans le chapitre suivant, nous nous intéressons à l'étude expérimentale des modèles présentés dans ce chapitre en utilisant l'outil *TimeNet*.

Chapitre 4 : Evaluation de performances des WSNs

Introduction

Nous présentons dans le présent chapitre, quelques expérimentations relatives aux modèles proposés, ce qui nous permettra au passage d'insister à nouveau sur l'importance du bon choix des paramètres adéquats pour avoir les meilleures performances possibles.

Par ailleurs, les composants d'un réseau de capteurs (nœuds de capteur) sont sujets à des pannes aléatoires, ce qui a une forte influence sur les paramètres de performance au même titre que le phénomène de rappel. Il est important de prendre ceci en compte lors de la construction du modèle et par la suite de son analyse.

En plus, la taille du buffer d'un capteur et la probabilité de perte des messages influent aussi sur les paramètres de performance. D'autre part, la durée de l'état de veille d'un capteur influe sur la quantité d'énergie consommée par un capteur. Donc, il est indispensable de connaitre le taux de mise en veille à choisir.

Par conséquent, notre étude va porter essentiellement sur l'influence du phénomène de rappel, la taille du buffer ainsi que la non-fiabilité des capteurs selon plusieurs politiques de panne sur les mesures de performances des réseaux.

Le modèle de RDPSG est très attrayant, car il offre une approche d'évaluation des performances basées sur une description formelle. Ceci permet l'utilisation du même langage pour la spécification, la validation, l'évaluation des performances et l'implémentation des systèmes.

Dans le cadre de cette étude expérimentale, les résultats numériques ont été calculés en utilisant l'outil software *TimeNet* version *4.0* , qui est un outil très performant, dédié à l'évaluation des performances. *TimeNet* offre à l'utilisateur un environnement graphique simple lui permettant de définir tous les éléments du modèle, ainsi que tous les paramètres de performance et de fiabilité désirés.

Ainsi, en se basant sur cet outil performant, nous obtenons des résultats qui sont présentés sous forme de tableaux et de graphes qui sont commentés et à partir desquels des conclusions seront tirées.

4.1. Description du logiciel *TimeNet 4.0*

TimeNet 4.0 est un progiciel et environnement graphique interactif qui permet de modéliser modéliser différents systèmes avec les méta-modèles de réseaux de Petri stochastiques généralisés et les réseaux de Petri stochastiques colorés (RDPSCs). Dans ce qui suit, nous allons présenter des fonctionnalités essentielles en se basant sur la documentation fournie avec le logiciel *TimeNet 4.0* [61].

4.1.1. Historique du logiciel *TimeNet 4.0*

TimeNET a été développé par l'équipe de *systèmes temps réel* et *groupe de Robotics* de l'université de Berlin en Allemagne. Le projet a été motivé par le besoin d'un logiciel puissant pour l'évaluation efficiente des réseaux de Petri stochastiques.

La première version du *TimeNet* était une révision majeure du l'outil *DSPNexpress* à l'université de Berlin. Elle contenait toutes les composantes d'analyse existantes maintenant mais pour la classe eDSPNs (extended Deterministic Stochastics Petri Nets : RDPS déterministes) seulement.

4.1.2. Fonctionnalités du *TimeNet 4.0*

Différentes solutions algorithmiques peuvent être utilisées en fonction de la classe de RDPS. Le logiciel *TimeNet* comporte une interface graphique spéciale pour la classe des RDPSC. Cette interface fournit différentes fonctionnalités liées à cette classe du réseau de Petri. ***TimeNet*** peut calculer la solution du modèle dans le régime stationnaire et fournit une technique d'analyse approximative pour la classe eDSPNs qui comporte les RDPSG et les RDPSD.

L'analyse, l'approximation et l'évaluation peuvent être effectuées pour une même classe du modèle. En plus de nouvelles fonctionnalités, les enrichissements du *TimeNet 3.0* sont inclus dans *TimeNet 4.0* parmi lesquels nous citons :

- Un algorithme permettant l'analyse transitoire pour la classe RDPSD.
- Un composant pour le régime stationnaire et l'analyse transitoire pour les RDPSG.
- Un composant spécial désigné aux systèmes de fabrication.
- Une interface graphique d'utilisateur complet qui intègre différentes classes de RDP et permet d'utiliser facilement différents algorithmes d'analyse.

4.1.3. Interface graphique du *TimeNet* 4.0

La nouvelle version du *TimeNet* comporte une interface graphique interactive développée à l'aide du langage *JAVA* et fournit un support du système d'exploitation *Windows*. Le *TimeNet* supporte

également maintenant la classe des RDPSC (RDPS Colorés). L'interface graphique générique du *TimeNet 4.0* permet la visualisation des résultats de l'évaluation et de l'analyse sous forme de graphes.

La fenêtre de l'interface graphique du *TimeNet 4.0* est divisée en quatre parties comme le montre la figure 4.1 :

- Une barre de menu en haut.
- Une zone de dessin à gauche.
- Une zone des attributs à droite.
- Une barre d'outils spécifique à la classe du RDP en bas.

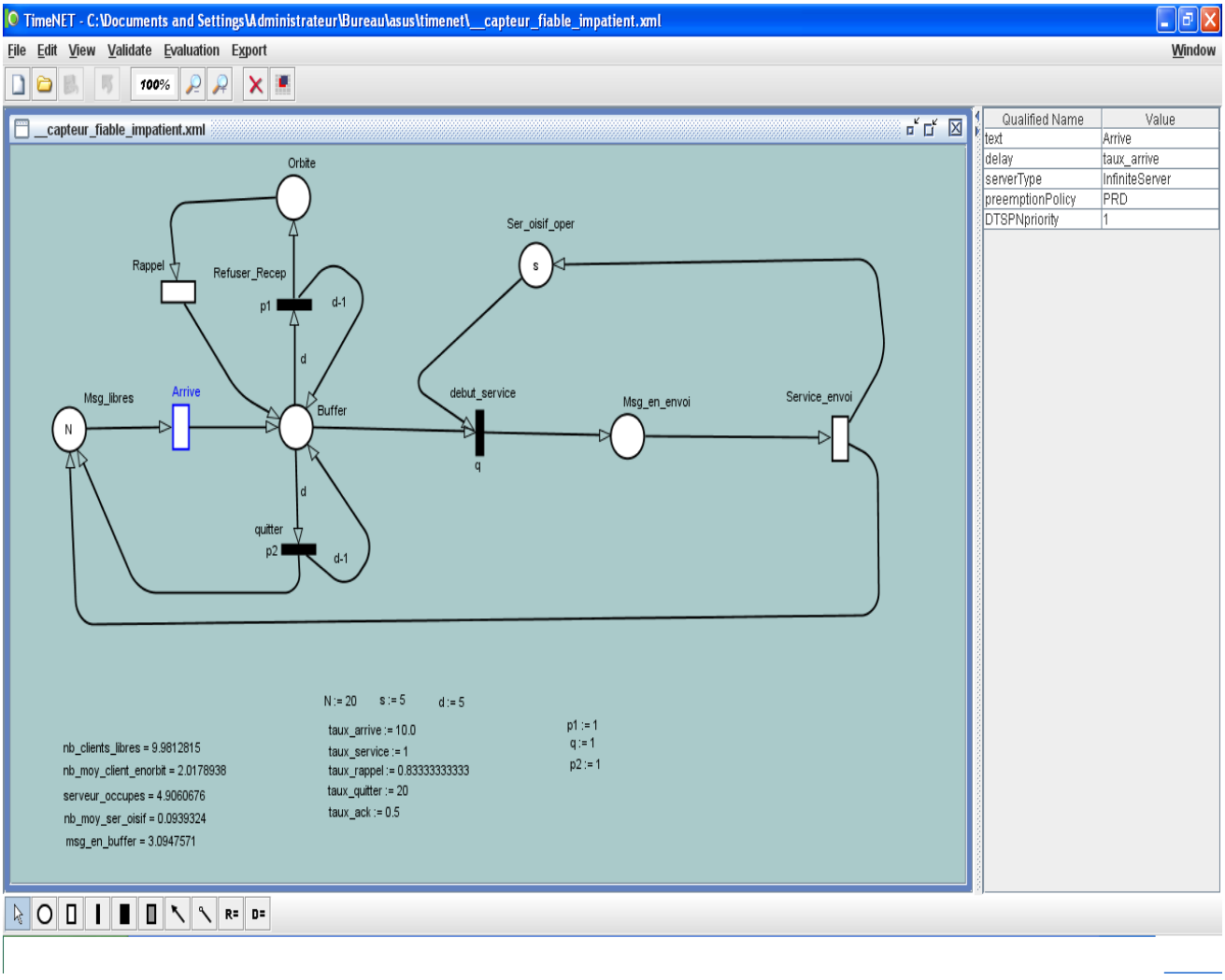


Figure 4.1. L'interface graphique du *TimeNet 4.0* sous Windows

4.1.4. Mesure de performances

Pour mesurer un paramètre de performance, il suffit de placer un objet « R =» à la surface du dessin. Une mesure de performance définit ce qui va être calculé durant l'analyse. La valeur d'une mesure peut être par exemple le nombre moyen de jetons dans une place.

Une mesure a les attributs suivants :

- Le nom du paramètre mesuré
- Une expression qui doit être évaluée
- Le résultat qui remplace l'expression après une évaluation réussite du modèle

Une mesure est une expression qui peut contenir des nombres, des taux de franchissement, des opérateurs algébriques et des mesures de base suivantes :

- **P** { <*condition_logique*>} : correspond à la probabilité de <*condition_logique*>.
- **P** { <*condition_logique1*> **IF** <*condition_logique*2>} : calcule la probabilité de <*condition_logique1*> sachant que <*condition_logique2*>.
- **E** { <fonction_marque (p)>} : renvoie le nombre de jetons dans une place en exécutant la fonction : <fonction_marque (p)>.
- **E {** <*fonction_marque (p)*> **IF** <*condition_logique*>***}* :** renvoie le nombre de jetons dans une place en exécutant la fonction : <fonction_marque (p)> et en satisfaisant la condition : <*condition_logique*>.

Exemple de mesures :

MEASURE *debit*

*E{#Msg_libres}*taux_arrive;*

MEASURE *no*

E{#Orbite};

***MEASURE** ns*

E{#Msg_en_envoi};

***MEASURE** nd*

E{#Ser_oisif_oper};

4.1.4. Analyse qualitative du *TimeNet 4.0*

Le *TimeNet* nous permet d'effectuer aussi bien une analyse qualitative du modèle saisi qu'une analyse quantitative.

Sous le menu 'Validate' du l'outil TimeNet, nous trouvons des fonctions basées sur la structure du modèle de RDPSG. Nous trouvons les fonctions suivantes :

- **Estimate Statespace** : calcule une estimation de la taille de l'espace d'états. La figure 4.2 montre un exemple d'utilisation de cette fonction.
- **Trapps** : calcule l'ensemble minimum de Trapps (i.e. ensemble de places qui pourrait toujours être marquée). La figure 4.3 montre un exemple d'utilisation de cette fonction.

- **Siphons** : calcule l'ensemble minimum de places absorbantes (i.e. ensemble de places qui ne pourrait être jamais marquées). La figure 4.4 montre un exemple d'utilisation de cette fonction.
- **Check Structure** : permet d'obtenir les invariants du modèle et les conflits des transitions immédiates. La figure 4.5 montre un exemple d'utilisation de cette fonction.

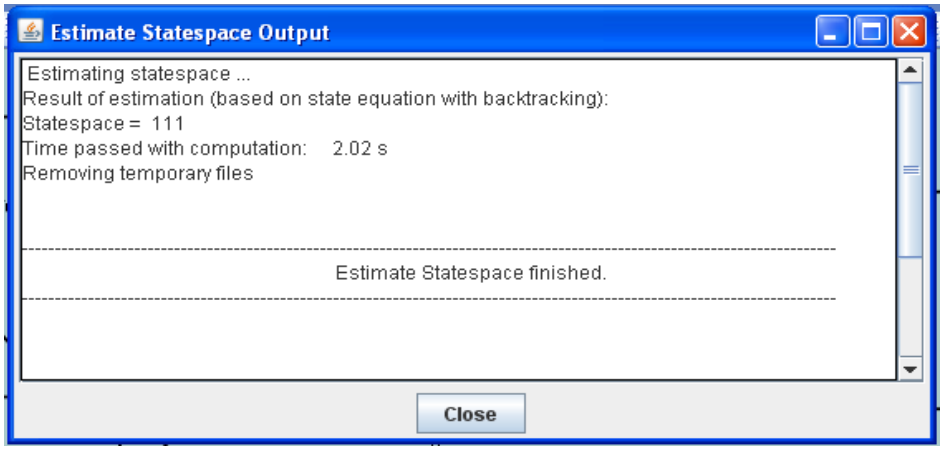


Figure 4.2. Résultat de l'estimation de la taille de l'espace d'états

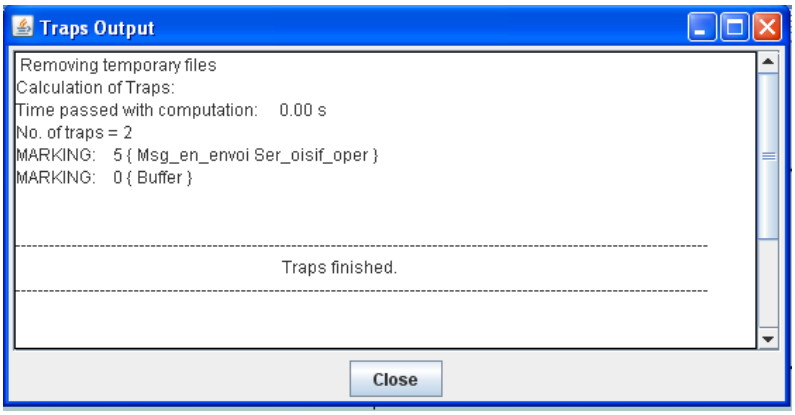


Figure 4.3. Résultat de l'exécution de la fonction *Traps*

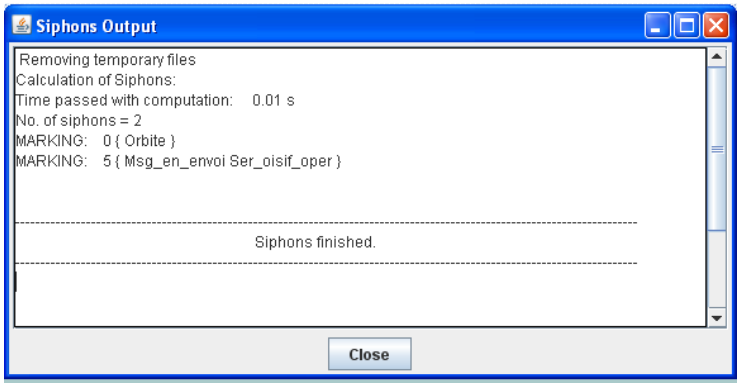


Figure 4.4. Résultat de l'exécution de la fonction *Siphons*

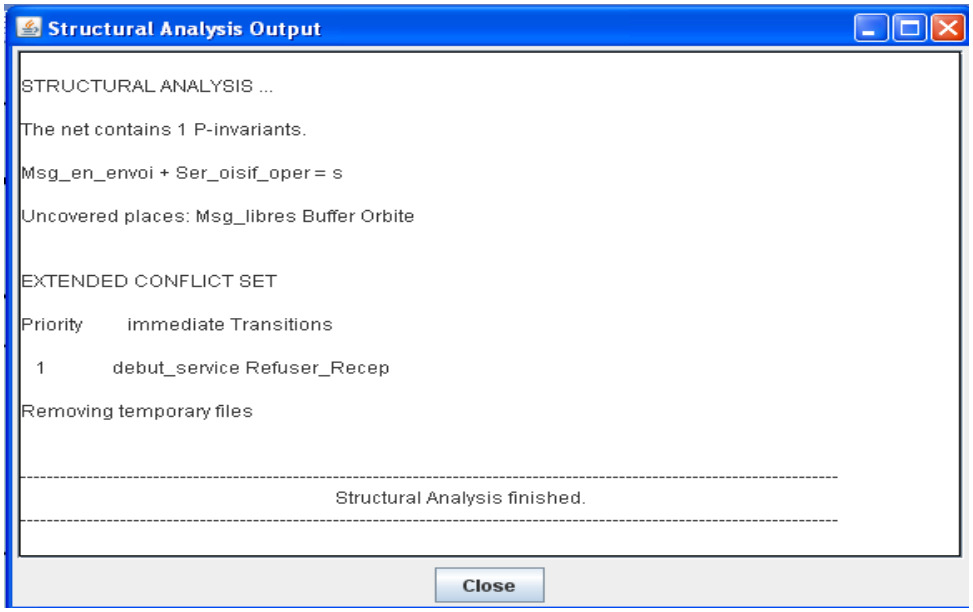


Figure 4.5. Résultat de l'exécution de la fonction *Check Structure*

4.1.6. Les méthodes d'analyse du *TimeNet 4.0*

Le menu ***Evaluation*** de l'interface graphique nous permet d'accéder aux fonctions d'évaluation de performance du modèle. Plusieurs méthodes d'évaluation sont disponibles (Figure 4.6), nous citons :

- L'analyse stationnaire.
- La simulation stationnaire.
- L'analyse transitoire.
- La simulation transitoire.

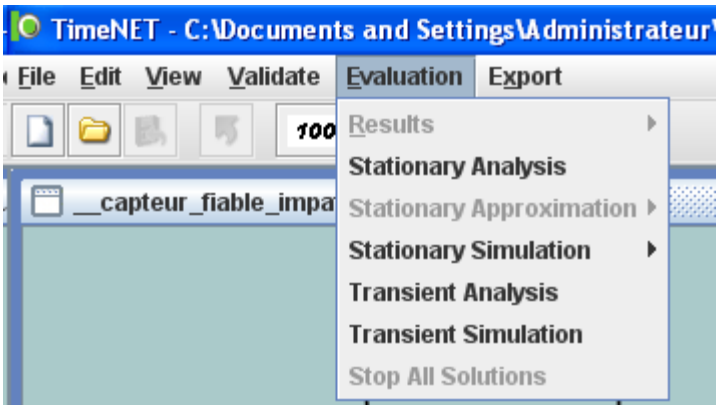


Figure 4.6. Menu d'évaluation du *TimeNet*

Nous présentons ici l'analyse en régime stationnaire

4.1.6. 1. Analyse stationnaire

L'analyse en régime stationnaire permet de calculer la solution d'un modèle. La figure 4.7 montre la fenêtre du *TimeNet* spécifique à cette analyse.

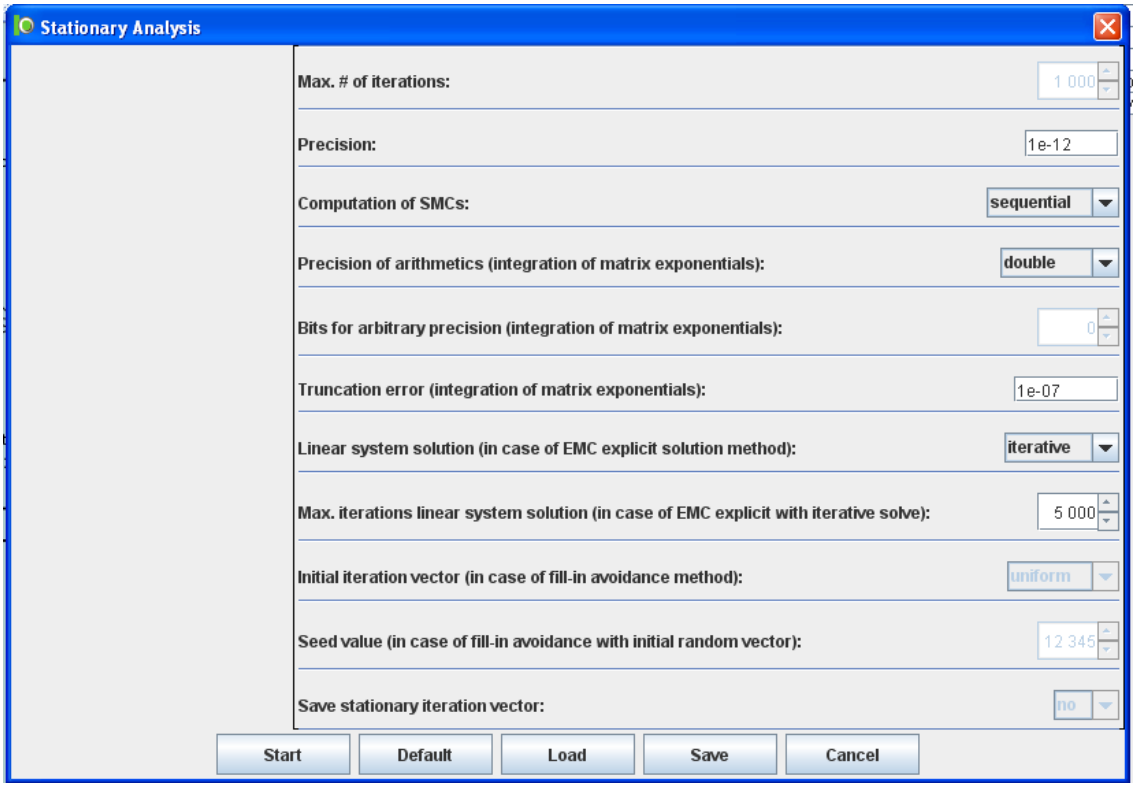


Figure 4.7. Fenêtre permet une analyse stationnaire

L'analyse stationnaire du *TimeNet* permet d'obtenir un ensemble de données qui peuvent être utilisées pour construire des graphes. La figure 4.8 représente les options de l'exécution expérimentale.

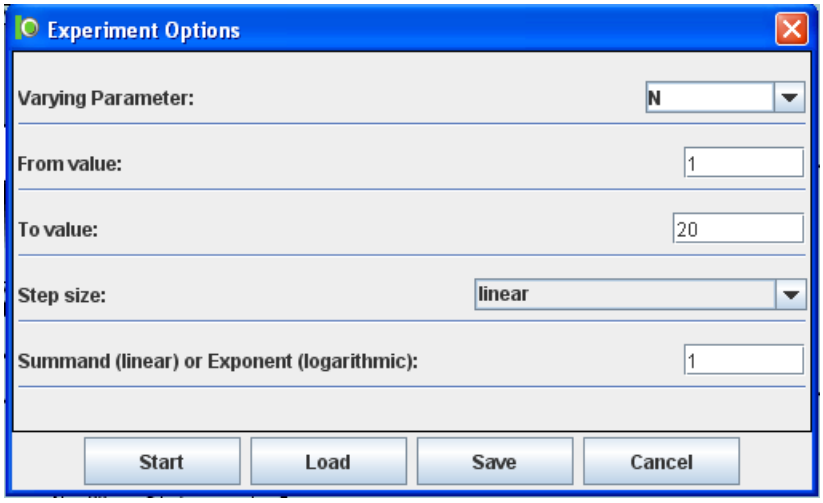


Figure 4.8. Fenêtre permet une exécution expérimentale

4.1.6. 2. Moniteur de résultats

Le moniteur de résultats du *TimeNet* permet de visualiser les résultats d'une analyse stationnaire expérimentale sous forme de courbes ou bien d'histogrammes (voir figure 4.9).

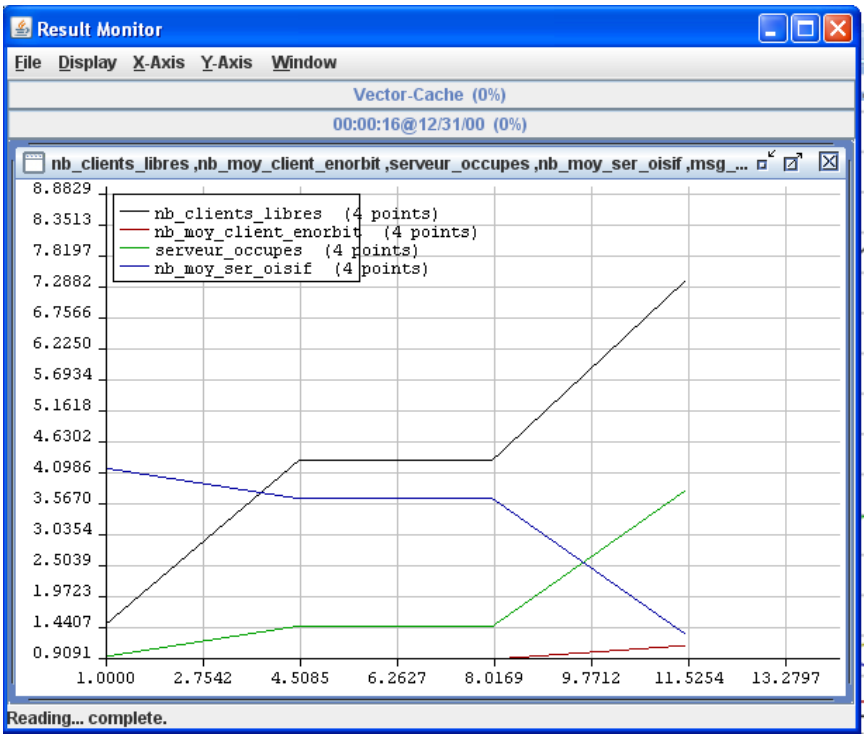


Figure 4.9. Moniteur de résultats de l'analyse stationnaire

4.1.7. Résultats de l'évaluation

Après une évaluation du modèle, les définitions des paramètres de performance (mesures) dans le modèle seront mises à jour et leurs attributs de résultat prennent les résultats de l'évaluation. Une évaluation de performance crée aussi des fichiers textuels contenant les résultats de l'évaluation. A titre d'exemple :

- *EXPRESULTS* contient un tableau des résultats d'une analyse expérimentale.
- *RESULTS* contient les résultats détaillés pour des paramètres de performance donnés. Ce fichier contient aussi les débits des transitions temporisées du modèle.
- *TN* contient le modèle en format *.TN*.
- *traps* contient les *traps* du modèle.
- INV contient les places invariantes du modèle.
- Etc.

4.2. Validation des modèles

Dans un premier temps, les résultats obtenus dans le cas fiable i.e. modèle de la figure 3.5 sont validés par un programme *Pascal* donné dans le livre de *Falin* et *Templeton* [60] et par un modèle analysé à l'aide de l'outil *GreatSPN* [50]. Pour cela, on suppose que le taux de panne dans un modèle non fiable soit très faible et que le taux de réparation soit très élevé. Dans ce cas, les valeurs des indices de performance devraient être proches des valeurs correspondantes au modèle fiable. Ainsi, pour la validation des résultats, nous supposons que la capacité du buffer '*d'* est égale à 1(car les modèles étudiés précédement sont sans buffer) et la probabilité de perte des messages est nulle (P2=0).

Le tableau 4.1 représente les résultats obtenus.

	Fiable	*Non fiable*	
		Panne active	*Pannes dépendantes*
Nombre de capteurs voisins	*4*	*4*	*4*
Nombre de messages libres	*20*	*20*	*20*
Taux d'arriveé	*0,1*	*0,1*	*0,1*
Taux de service	*1*	*1*	*1*
Taux de rappel	*1,2*	*1,2*	*1,2*
Taux de panne active	*/*	*1,00E-05*	*1,00E-05*
Taux de panne oisif	*/*	*/*	*1,00E-06*
Taux de réparation des capteurs	*/*	*1,00E+06*	*1,00E+06*
Nombre moyen de messages en orbite	*0,191771526175520*	*0,191786951598490*	*0,191786951598760*
Nombre moyen de voisins occupés	*1,800748043074950*	*1,800746640763770*	*1,800746640763750*
Taux moyen de génération des messages primaires	*1,800748043074950*	*1,800746640763770*	*1,800746640763740*
Temps d'attente moyen	*0,10649547942757*	*0,10650412848592*	*0,10650412848607*
Temps de réponse moyen	*1,10649547942757*	*1,10650412848592*	*1,10650412848608*

Tableau 4.1 : validations des modèles

4.3. Etude expérimentale

Le but de ces expérimentations est d'étudier l'influence du phénomène de rappel, la taille du buffer ainsi que la non-fiabilité des capteurs sur les mesures de performance d'un réseau de capteurs sans fil.

Nous présentons dans ce qui suit, les résultats numériques de l'étude des performances et de la fiabilité d'un réseau de capteurs sans fil, avec capteurs identiques et non fiables. En premier lieu, nous illustrons l'effet des paramètres du trafic de communication entre capteurs et les disciplines de panne, sur le temps de réponse moyen. Comme nous étudions un réseau de capteurs où un capteur a plusieurs voisins (des capteurs sont non fiables et réparables), nous nous intéressons particulièrement à l'influence du taux de rappel, de la taille du buffer, du taux de panne des capteurs voisins occupés, du temps de réparation et du nombre de capteurs voisins sur le temps de réponse moyen. Pour cela, nous considérons les quatre modèles suivants :

- Modèle 1: Réseau de capteurs avec rappel, buffer de taille limitée, messages impatients, pannes actives et sources non-bloquées (modèle de la figure 3.6).
- Modèle 2 : Réseau de capteurs avec rappel, buffer de taille limitée, messages impatients, pannes dépendantes et sources non-bloquées (modèle de la figure 3.7).
- Modèle 3 : Réseau de capteurs avec rappel, buffer de taille limitée, messages impatients, pannes actives et sources bloquées (modèle de la figure 3.8).
- Modèle 4 : Réseau de capteurs avec rappel, buffer de taille limitée, messages impatients, pannes dépendantes et sources bloquées (modèle de la figure 3.9).
- Modèle 5 : Réseau de capteurs avec rappel, buffer de taille limitée, messages impatients, pannes dépendantes et mise en veille des capteurs (modèle de la figure 3.10).

Nous nous concentrons ensuite sur quatre descripteurs de performance et de fiabilité importants qui sont : le nombre moyen de messages en orbite, le nombre moyen de messages dans le buffer, la probabilité de blocage et le nombre moyen de capteurs voisins en panne.

Le Tableau 4.2 représente les paramètres d'entré pour les études suivantes.

Dans les tableaux 4.3-4.7, nous comparons les temps de réponse moyens pour les différentes disciplines de panne. Nous vérifions respectivement, l'effet du taux de rappel, du taux de panne des voisins occupés, du temps de réparation, du nombre de voisins d'un nœud capteur et de la taille du buffer sur le temps de réponse moyen.

	N	*s*	*d*	*1/λ*	*1/μ*	*1/ν*	*1/γ*	*1/δ*	*1/τ*	*1/δν*	*1/τν*
Tableau 4.3	*10*	*5*	*3*	*1 min*	*20 ms*	***var***	*50 ms*	*100 ms*	*5OO ms*	*20 ms*	*200 ms*
Tableau 4.4	*10*	*5*	*3*	*1 min*	*20 ms*	*5 ms*	***var***	*100 ms*	*5OO ms*	*20 ms*	*200 ms*
Tableau 4.5	*10*	*5*	*3*	*1 min*	*20 ms*	*5 ms*	*50 ms*	*100 ms*	***var***	*20 ms*	*200 ms*
Tableau 4.6	*10*	***var***	*3*	*1 min*	*20 ms*	*5 ms*	*50 ms*	*100 ms*	*5OO ms*	*20 ms*	*200 ms*
Tableau 4.7	*10*	*5*	***var***	*1 min*	*20 ms*	*5 ms*	*50 ms*	*100 ms*	*5OO ms*	*20 ms*	*200 ms*

Tableau 4.2. Paramètres d'entrée

Certains paramètres proposés dans le tableau 4.2 sont pris à partir de [41] pour donner une illustration proche da la réalité.

4.3.1. L'effet du taux de rappel sur le temps de réponse

ν	*Modèle 1*	*Modèle 2*	*Modèle 3*	*Modèle 4*	*Modèle 5*
1	601.2910794	310.8512962	601.6280245	589.1494945	571.2400384
0.1	242.7212985	169.669032	242.8177545	621.5665323	409.1804552
0.01	207.1913023	155.2677384	207.1351045	497.5454605	499.9097214
0.001	203.6497545	156.2901246	203.5749887	484.6082015	566.3134346
0.0001	203.2957642	156.1248479	203.2191055	479.7895067	623.7330437
0.00001	203.2603669	156.1082933	203.1835186	475.1680848	623.7165775
0.000001	203.2568272	156.1066376	203.1799599	475.1604321	623.71493

Tableau 4.3. L'effet du taux de rappel sur le temps de réponse

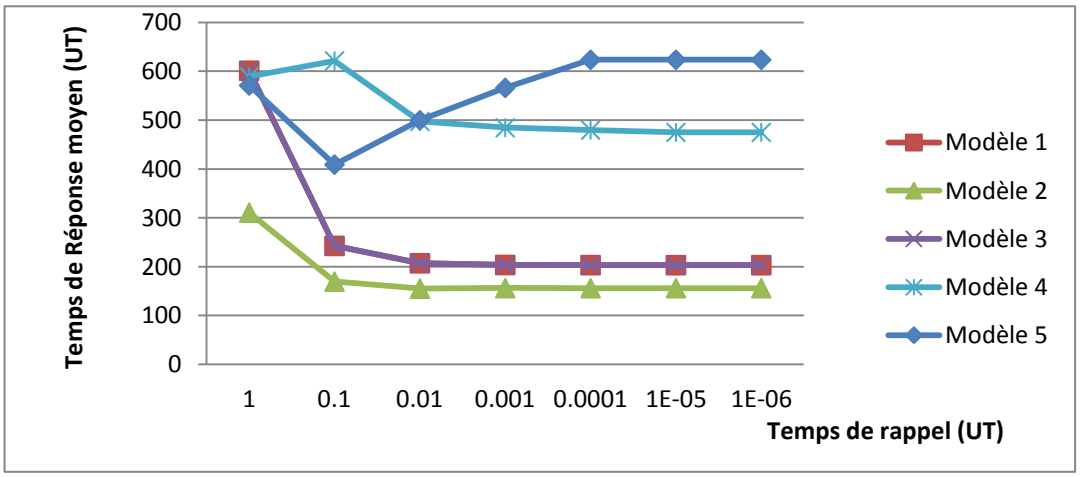


Figure 4.10. L'effet du taux de rappel sur le temps de réponse [1]

[1] Le temps de réponse se calcule selon une **U**nité de **T**emps (UT).

Dans le tableau 4.3 et la figure 4.10, on voit bien que le taux de rappel a une influence significative sur le temps de réponse moyen quand ce taux est petit. Cependant, quand la durée entre deux rappels successifs est faible, le temps de réponse est plus important. Il est plus intéressant d'appliquer la discipline des pannes dépendantes et sources non-bloquées.

4.3.2. L'effet du taux de panne active sur le temps de réponse

γ	*Modèle 1*	*Modèle 2*	*Modèle 3*	*Modèle 4*	*Modèle 5*
0.0001	200.634349	294.594456	200.634349	153.332002	1830.18059
0.001	201.534295	269.713545	201.534295	154.621189	1494.95942
0.01	210.672356	148.634426	210.674703	154.433539	1713.18433
0.1	151.216549	623.625588	167.295241	384.496039	2528.19393
1	189.270512	297.843641	192.838567	263.309886	4085.85899

Tableau 4.4. L'effet du taux de panne active sur le temps de réponse moyen

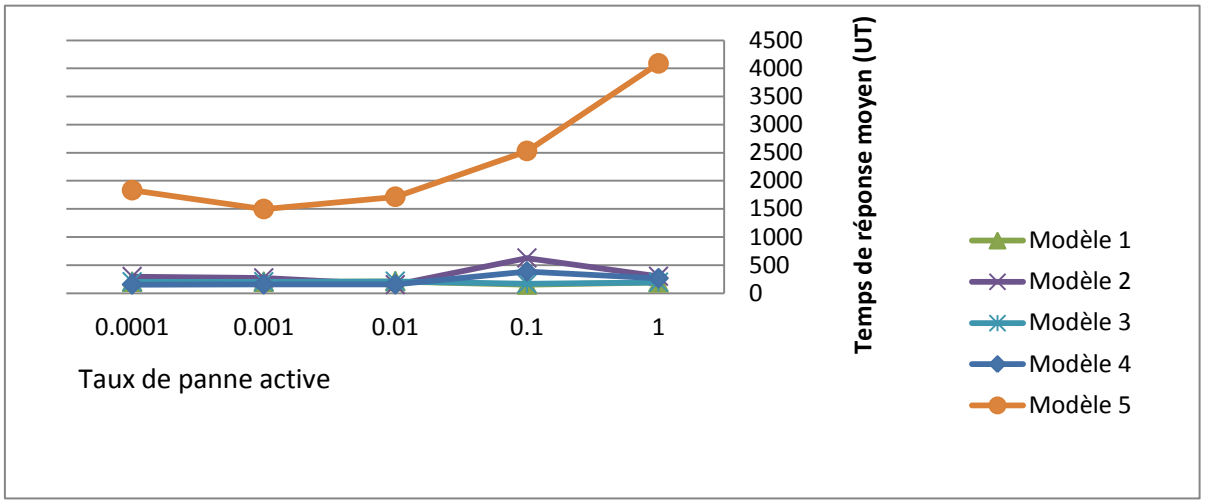


Figure 4.11. L'effet du taux de panne active sur le temps de réponse moyen

Le tableau 4.4 et la figure 4.11, montrent que le temps de réponse moyen croît quand la durée entre deux pannes successives est faible (le taux de panne des capteurs voisins occupés croît). Nous pouvons constater que parmi les cinq modèles, le modèle avec pannes dépendantes et sources bloquées (modèle 4) donne les meilleures performances quand le taux de panne active est plus grand que la valeur *0.1*. Ainsi, pour le cas où les capteurs d'un réseau sans fil sont anciens par exemple, dont les capteurs tombent fréquemment en panne, il est plus intéressant d'appliquer la discipline des pannes actives avec sources bloquées.

4.3.3. L'effet du temps de réparation sur le temps de réponse

1/τ	*Modèle 1*	*Modèle 2*	*Modèle 3*	*Modèle 4*	*Modèle 5*
1.00E-05	220.534778	220.534778	220.534778	400.564492	785.837328
1.00E-04	220.534778	220.534778	220.534778	400.564492	791.899921
0.001	220.534778	220.534778	220.534778	400.564492	800.136117
0.01	220.534778	220.534778	220.534778	400.564492	847.553931
0.1	220.534778	220.534778	220.534778	400.564492	1534.07188
1	220.534778	220.534778	220.534778	400.564494	1614.28789
10	220.534782	220.536139	220.53479	401.359793	1205.99647
100	220.652139	241.183903	220.674369	814.713725	3945.71284
1000	227.225299	829.123455	226.997624	207.738283	1122.4798
10000	688.455692	5423.90917	552.798986	70.8046146	1073.15589
100000	52528.876	75121.2308	25312.1805	773.578738	1715.07977

Tableau 4.5. L'effet du temps de réparation sur le temps de réponse moyen

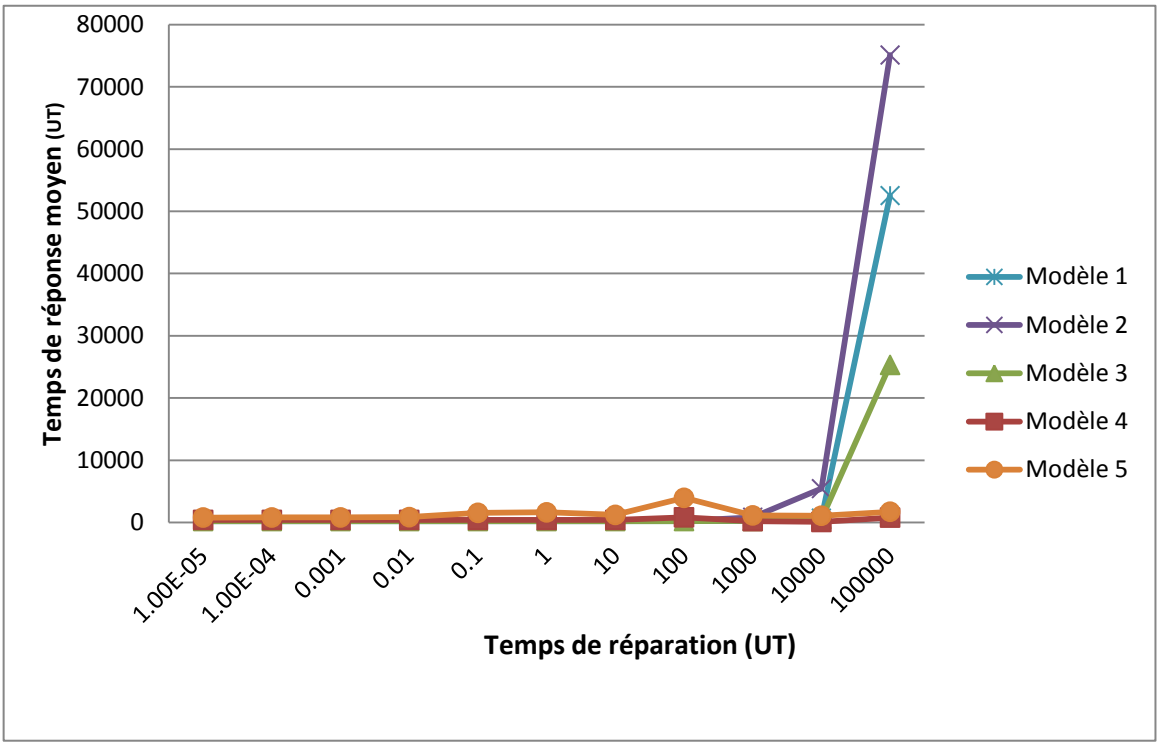


Figure 4.12. L'effet du temps de réparation sur le temps de réponse moyen

Dans le tableau 4.5 et la figure 4.12, on constate que le temps de réponse moyen est minime et presque le même pour les quatre modèles quand la durée de réparation est réduite (de 10^{-5} *à* 10^{4}), puis quand le temps de réparation croît de plus en plus, les modèles 3 et 4 sont plus efficaces que les

modèle 4 et 5 donnent les meilleures performances. Le modèle avec pannes actives et sources bloquées (modèle 3) donne des performances intermédiaires.

A partir des tableaux 4.3 et 4.5 et les figures 4.10 et 4.12, nous voyons aussi que les temps de réponse moyens sont meilleurs (plus faibles) dans le modèle avec pannes dépendantes et sources bloquées (modèle 4) par rapport aux autres modèles, et ce quand les temps de rappel et de réparation sont importants. Cependant, quand ces temps diminuent, la différence entre les cinq modèles n'est plus significative.

4.3.4. L'effet du nombre de capteurs voisins sur le temps de réponse

S	*Modèle 1*	*Modèle 2*	*Modèle 3*	*Modèle 4*	*Modèle 5*
1	2386.23502	6430.31479	2154.60259	3404.33406	43847.5295
2	2386.23502	2928.19006	2154.60259	3100.94317	40792.4419
3	276.941195	1790.61369	275.62973	3100.94317	39773.003
4	231.948899	625.887577	275.62973	574.464254	1132.86128
5	222.960799	159.273365	222.979483	554.95507	926.370134
6	221.067359	306.682107	221.077562	117.855176	550.247633
7	220.655045	236.604534	220.65857	117.855176	997.10469
8	220.562785	177.262596	220.65857	183.137664	997.10469

Tableau 4.6. L'effet du nombre de capteurs voisins sur le temps de réponse moyen

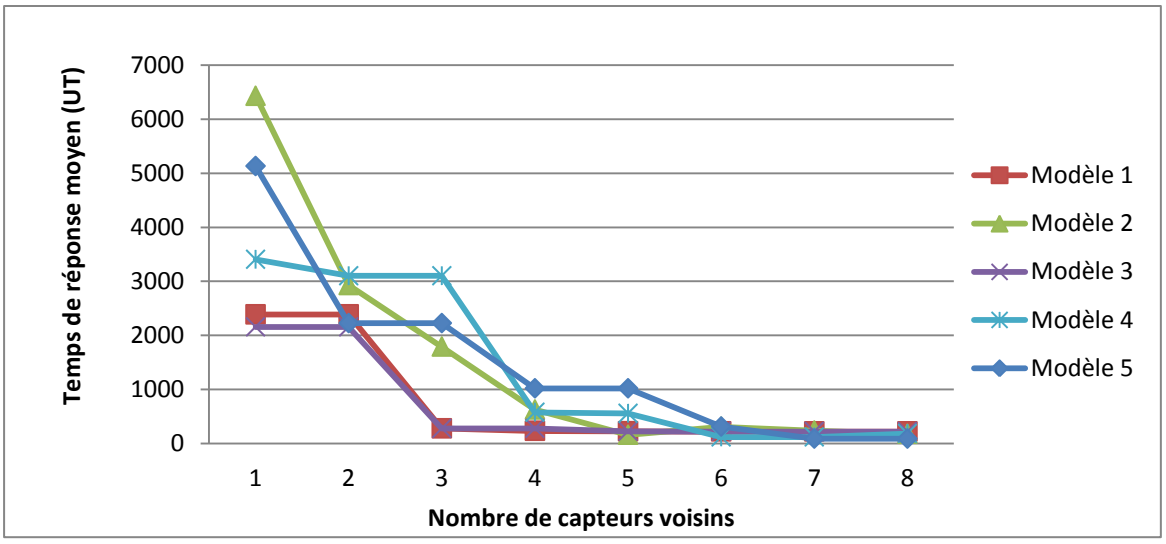


Figure 4.13. L'effet du nombre de capteurs voisins sur le temps de réponse moyen

Dans le tableau 4.6 et la figure 4.13, nous remarquons que des petites modifications dans le nombre des voisins d'un capteur, particulièrement de 1 à 4 voisins, produit une très importante baisse

dans le temps de réponse moyen de 2000 unités de temps à 2 (environ de -60%) pour les modèles 1, 2, 3 et 4. Cependant, après une certaine valeur (s=3), la décroissance n'est pas considérable. Les modèles avec pannes actives (modèle 1 et modèle 3) donnent des meilleures performances par rapport aux autres modèles (modèle 2, modèle 4 et modèle 5). Ce coïncident avec la logique qui dit que le temps de réponse moyen est meilleur quand les capteurs voisins ne tombent en panne seulement si sont en fonction (nombre important de capteurs libres et opérationnels).

5.3.5. L'effet de la taille du buffer sur le temps de réponse

d	*Modèle 1*	*Modèle 2*	*Modèle 3*	*Modèle 4*	*Modèle 5*
1	200.2350396847	127.898010658	201.44517682890	416.2647632521	44.76099250
2	222.7893995264	153.651885470	222.97790687400	416.2647632521	44.76099250
3	222.9607985465	159.273365392	222.97948281630	554.9550695352	613.18915187
4	222.9658816007	159.329011918	222.97949225840	548.8376333056	622.71034196
5	222.9659871763	162.123416812	222.97949229330	545.9307158232	565.43523020
6	222.9659889291	162.123420228	222.97949229340	545.6779083499	643.44244285
7	222.9659889527	164.709113842	222.97949229340	547.5118024305	665.29444924
8	222.9659889529	168.768219393	222.97949229340	553.1362868173	665.29442910
9	222.9659889529	172.748220906	222.97949229340	563.3683403957	648.22499786
10	222.9659889529	172.748221011	222.97949229340	569.1194512991	654.04551639
20	222.9659889529	172.748220967	222.97949229340	569.1194512991	654.04551639
30	222.9659889529	172.748220967	222.97949229340	569.1194512991	654.04551639

Tableau 4.7. L'effet de la taille du buffer sur le temps de réponse moyen

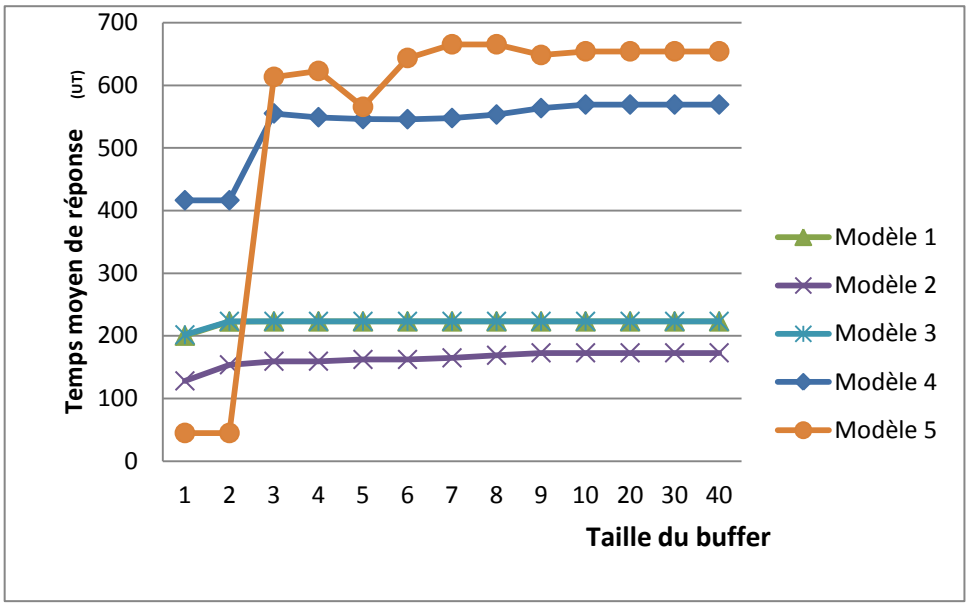


Figure 4.14. L'effet de la taille du buffer sur le temps moyen de réponse

Dans le tableau 4.7 et la figure 4.14, nous remarquons que le temps de réponse augmente faiblement quand la taille du buffer augmente. Le temps de réponse se stabilise sur une valeur quand la taille du buffer atteint une certaine valeur (d=3). Les modèles 1, 2 et 3 donnent des meilleures performances par rapport aux modèle 4 et 5. Dans le modèle 5, le temps de réponse augmente énormément (plus de 600 unités de temps). Ceci est logique car dans le modèle 5 d'autres facteurs influent sur le réseau et particulièrement les temps de mise en veille et oisif (dans notre exemple on a le temps de mise en veille et le temps de mise en oisif sont égal à 10).

Finalement, les résultats numériques dans les tableaux 4.3-4.6 et les figures 4.10-4.13 coïncident avec la logique qui dit que le temps de réponse moyen est meilleur quand le système est neuf (taux de panne faible), la durée de réparation est minime, et quand le taux de rappel et l'intensité du réseau (nombre important de capteurs sur une petite surface) sont importants. Nous remarquons aussi que le modèle 3 est plus efficace que les autres modèles.

Nous donnons dans ce qui suit, des graphes représentant d'autres mesures de performances importants, en fonction de ces facteurs : taux de rappel, taux de panne et taux de la mise en veille.

Le tableau 4.8 et la figure 4.15, représentent le temps de réponse moyen en fonction du rapport *r =taux de veille/ taux de l'activation* (δ_v / τ_v) en utilisant le modèle 5.

Le tableau 4.9 et la figure 4.16, représentent le nombre moyen de messages bloqués en fonction du taux de rappel en utilisant le modèle 2.

Le tableau 4.10 et la figure 4.17, représentent le nombre moyen de capteurs en panne en fonction du taux de panne active en utilisant le modèle 3.

4.3.6. L'effet du rapport *taux de veille / taux de l'activation* sur le temps de réponse

δ_v / τ_v	*Temps de réponse*
0.002	931.73391018
0.02	931.73391018
0.20	931.53105502
2.00	929.69238489
20.00	911.83335787
200.00	775.92974765
2000.00	279.26731211
20000.00	1434.62644461
200000.00	14270.03837885

Tableau 4.8. L'effet du rapport δ_v / τ_v sur le temps de réponse moyen

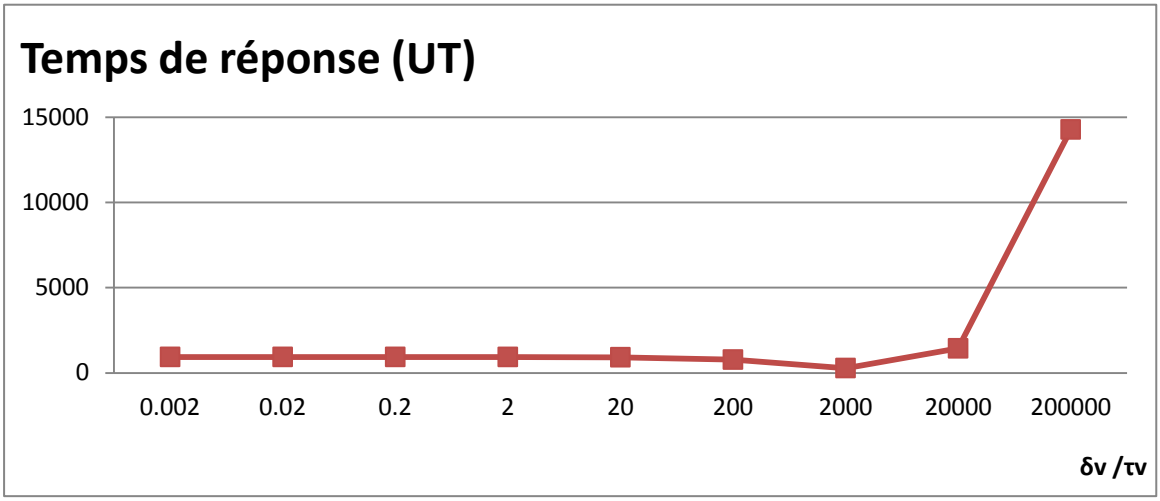


Figure 4.15. L'effet du rapport δ_v / τ_v sur le temps de réponse moyen

Dans le tableau 4.8 et la figure 4.15, nous remarquons que le temps de réponse est stable quand le rapport δ_v / τ_v est inférieur à *20 000*. Le temps de réponse augmente rapidement quand le rapport δ_v / τ_v dépasse la valeur *20 000*.

4.3.7. L'effet du taux de rappel sur le nombre moyen de messages bloqués

v	S=1	S=3	S=5	S=10
1E+05	7E-10	7E-10	2E-10	4.00000000E-10
1E+03	6.79E-08	6.67E-08	2E-10	4.00000000E-10
1E+01	6.7903E-06	6.6743E-06	2.2882E-06	3.90060000E-06
1E-01	6.7903E-06	6.6743E-06	0.00020859	3.29741700E-04
1E-03	0.06656674	0.00183446	0.00183446	1.76908770E-03
1E-05	3.98739714	0.00155501	0.00155501	1.87791940E-03

Tableau 4.9. Le nombre des messages bloqués en fonction du taux de rappel

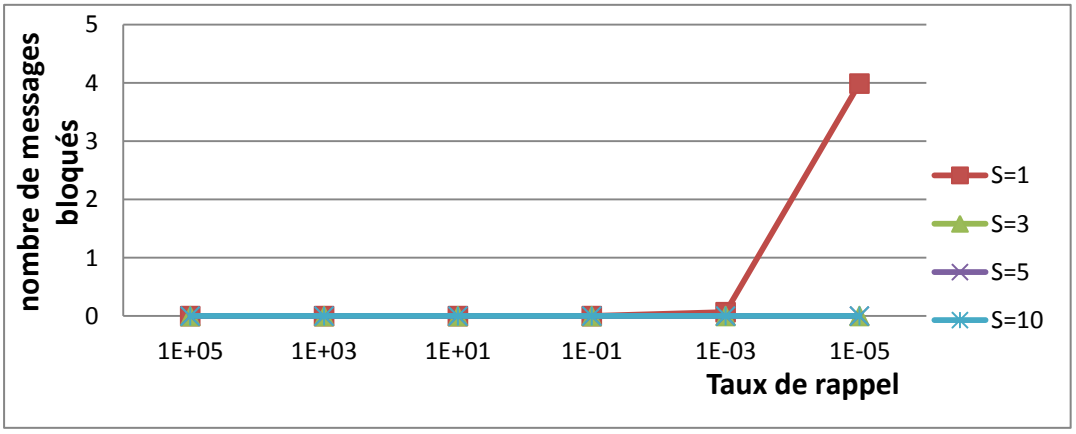


Figure 4.16. Nombre moyen de messages bloqués en fonction du taux de rappel

4.3.8. L'effet du nombre moyen de serveurs en panne en fonction du taux de panne active

Taux de panne	*γ=0,002*	*γ=0,01*	*γ=0,1*	*γ=1*	*γ=10*	*γ=100*
0.00001	1.67112E-05	3.3422E-06	3.342E-07	3.34E-08	3.3E-09	3E-10
0.0001	0.000167112	0.000334219	3.3422E-06	3.342E-07	3.34E-8	3.34E-08
0.001	0.001671097	0.000334219	0.000334181	3.34181E-05	3.342E-07	3.34E-08
0.01	0.022847981	0.003341789	0.000334181	3.34181E-05	3.33945E-05	3.342E-07
0.1	0.022847981	0.003727016	0.003339413	0.003332213	3.33945E-05	3.33221E-05
1	0.012308866	0.003727016	0.001344416	0.003332213	0.003768963	3.33221E-05
10	0.012308866	1.67112E-05	0.001344416	0.038853484	0.003768963	3.3E-09

Tableau 4.10. Nombre moyen de serveurs en panne en fonction du taux de panne active

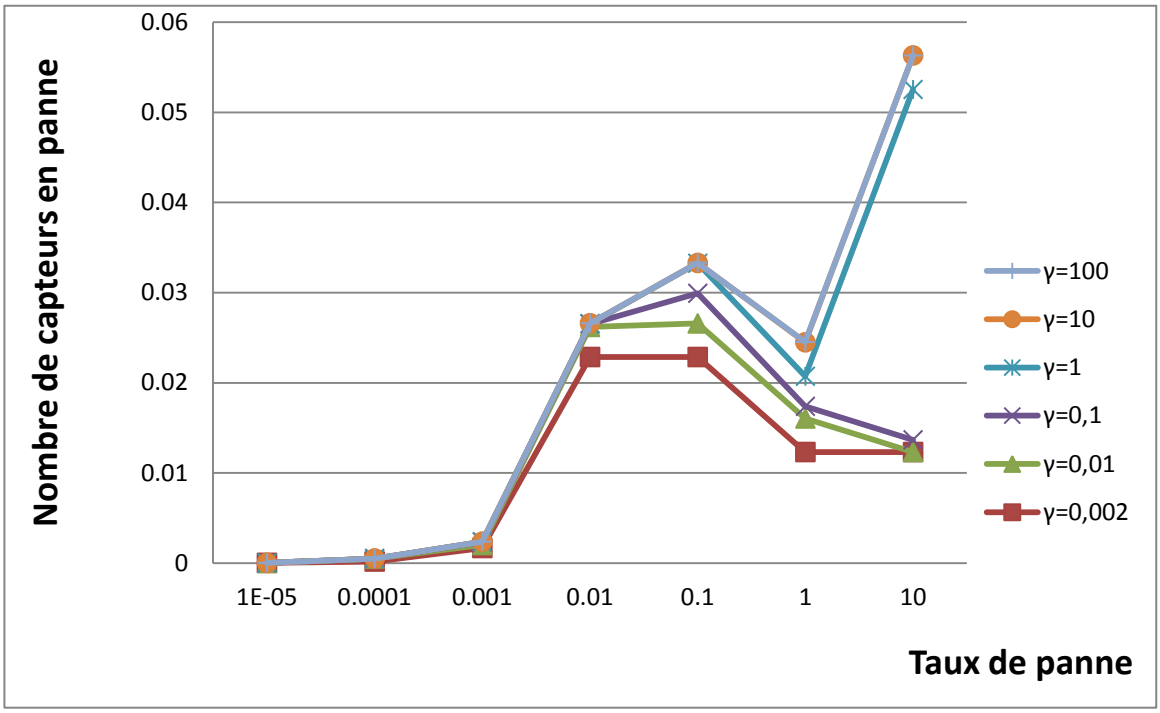


Figure 4.17. Nombre moyen de capteurs voisins en panne en fonction du taux de panne active

La figure 4.17 illustre le comportement du nombre moyen de capteurs voisins en panne en fonction du taux de panne active. Les courbes montrent également l'influence du taux de réparation sur ce nombre moyen de capteurs voisins en panne. Dans cette figure, nous pouvons voir que le nombre de capteurs voisins non opérationnels croît quand le taux de panne croît, et décroît en incrémentant le taux de réparation. Ainsi, dans le cas où le taux de réparation est égal au taux de rappel (v=0.2), on remarque que le nombre moyen de capteurs en panne décroît rapidement ensuite il croît quand les

pannes sont plus fréquentes croît. Donc, les résultats idéaux sont obtenus quand le taux de panne est faible et le taux de réparation est élevé (réparation rapide), ce qui est intuitivement logique.

4.3.9. Utilisation du réparateur en fonction du taux de panne

γ	s=1	s=3	s=4	s=7	s=12
0,02	0,0291	0,0554	0,0569	0,0574	0,0575
0,05	0,0867	0,1722	0,1786	0,1813	0,1814
0,16	0,2313	0,4932	0,5287	0,5617	0,57
0,50	0,487	0,9123	0,9622	0,993	0,9993
1,58	0,6954	0,8965	0,9151	0,8896	0,961

Tableau 4.11: Utilisation du réparateur en fonction du taux de panne

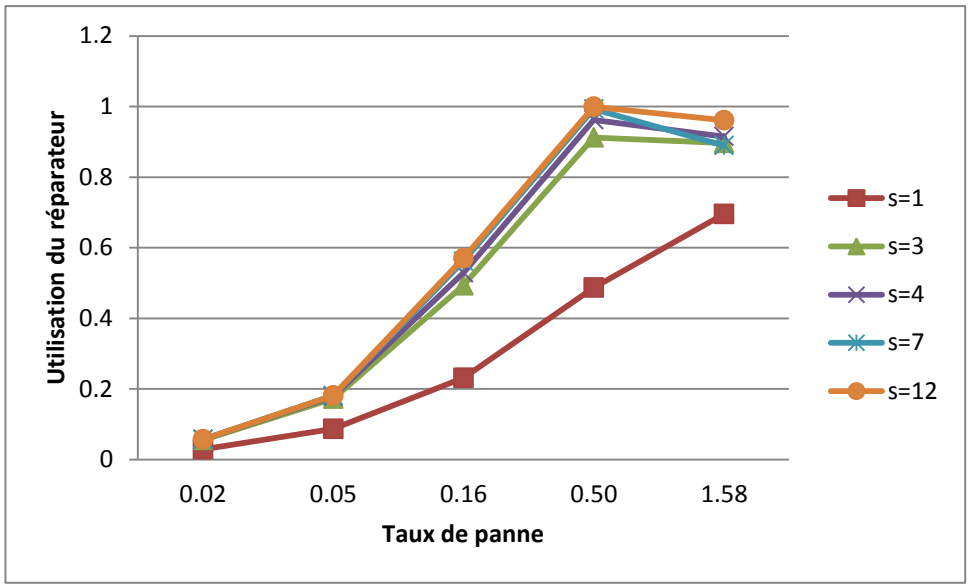


Figure 4.18 : Utilisation du réparateur en fonction du taux de panne

Dans la figure 4.16, le nombre moyen de messages bloqués (en orbite) est représenté en fonction du taux de rappel ν. Nous avons présenté quatre courbes qui correspondent au nombre de capteurs voisins s=1, 3, 5 et 10. A partir de cette figure, on constate que le nombre moyen de messages bloqués croit doucement quand le taux de rappel croit et quant le nombre de capteurs voisins décroit.

La figure 4.16 montre la probabilité de blocage en fonction du taux de rappel. Là aussi, nous voyons bien qu'une incrémentation du nombre de voisins d'un capteur permet de réduire d'une manière significative la probabilité de blocage.

La figure 4.17 montre l'effet du taux de panne des capteurs voisins en traitement sur l'utilisation du réparateur avec le modèle 1, Pour comprendre aussi l'influence du nombre de voisins, nous avons 5 courbes correspondent aux valeurs de s 1, 3, 4, 7 et 12. Ainsi, nous avons pu constater que l'utilisation du réparateur croît avec le taux de panne et le nombre de capteurs voisins.

Finalement, nous pouvons conclure que le taux de rappel, le taux de panne, le taux de réparation et l'intensité du réseau, sont des facteurs essentiels qui affectent la performance et la fiabilité d'un réseau de capteurs.

Conclusion

L'analyse d'un réseau de capteurs et plus précisément, le trafic de communication inter capteurs par le modèle de RDPSG permet de prendre en compte plusieurs paramètres qui influent sur la performance d'un tel réseau. Tels que le phénomène de rappel, la non fiabilité des capteurs et la limitation de la taille du buffer est un des avantages de cette analyse.

Diverses expérimentations ont été accomplies en utilisant l'outil *TimeNet version 4.0*, pour montrer l'effet des différents paramètres et des différentes disciplines de panne, sur les performances d'un réseau de capteurs ans fil avec la prise en considération de la non-fiabilité.

En effet, sur la base d'une série d'expérimentations et de résultats numériques exacts obtenus, nous avons pu étudier les possibilités qui permettent l'optimisation et l'amélioration des performances d'un réseau de capteurs sans fil. D'ailleurs, les résultats obtenus nous ont confirmé qu'une conception réfléchie pour choisir précisément les bons paramètres du trafic de communication entre capteurs et les disciplines de panne adéquates, sans négliger l'aspect économique, qui sont très importants pour atteindre les meilleures performances possibles.

Conclusion générale

Dans ce travail, nous avons étudié la modélisation et l'évaluation des performances des réseaux de capteurs sans fil. De tels réseaux ont un intérêt certain dans les applications de surveillance de l'environnement. Pour atteindre cet objectif, nous nous sommes appuyés sur le méta-modèle de Réseaux de Petri Stochastiques Généralisés pour modéliser le trafic de communication dans un WSN.

Nous avons commencé par présenter d'une manière générale les réseaux de capteurs sans fil, leurs caractéristiques et leurs applications. Nous avons présenté aussi les travaux liés à notre recherche. Ensuite, on a présenté le modèle de «réseaux de Petri stochastiques généralisés» qu'on a utilisé pour modéliser et analyser le trafic de communication dans les réseaux de capteurs sans fil.

L'analyse d'un réseau de capteur et plus précisément, le trafic de communication inter capteurs par le modèle de RDPSG permet de prendre en compte plusieurs paramètres qui influent sur la performance d'un tel réseau. Ainsi, la considération du phénomène de rappel, du non fiabilité des capteurs et la limitation du buffer est un des avantages de cette analyse.

En plus, nous avons étudié la fiabilité des capteurs, les demandes répétitives de réception, la taille du buffer et nous avons montré leurs influences sur les paramètres de performances. Nous avons utilisé l'outil logiciel ***TimeNet 4.0*** pour faire l'évaluation et l'application numérique des modèles proposés dans ce projet. Les résultats sont présentés sous forme des tableaux et des graphes.

Dans le but de réduire le temps de réponse d'envoi d'un message dans un RCSF, qui a une influence directe sur la consommation d'énergie, nous avons étudié l'effet de certains paramètres de performances importants tels que :

- L'effet de taux du rappel sur le temps de réponse
- L'effet du taux de panne active sur le temps de réponse
- L'effet du temps de réparation sur le temps de réponse
- L'effet du nombre de capteurs voisins sur le temps de réponse moyen
- L'effet de la taille du buffer sur le temps de réponse moyen
- L'effet du rapport taux veille /oisif ; sur le temps de réponse moyen
- L'effet du taux de rappel sur le nombre moyen de messages bloqués
- L'effet du taux de panne active sur le nombre moyen de serveurs en panne

Ainsi, nous avons conclu que le taux de rappel, le taux de panne, le taux de réparation et l'intensité du réseau, sont des facteurs essentiels qui affectent la performance et la fiabilité d'un réseau de

capteurs. Ainsi, les résultats obtenus après une analyse d'un tel modèle permettent aux concepteurs des réseaux de capteurs de bien choisir le matériel utilisé et leurs contraintes technique pour atteindre le but désiré et pour répondre aux exigences voulus par un tel réseau.

En fin, comme perspectives, il serait intéressant d'orienter des études qui consistent à la modélisation des différents problèmes posés dans le domaine des réseaux de capteurs en nous basant sur le modèle mathématique des RDPSG et autres extensions des RDP. Il serait plus intéressant d'inclure d'autres paramètres pour mesurer les paramètres d'un WSN tel que le choix de l'algorithme de routage de données. D'autres études plus poussée peuvent être menées comme l'existence de nœuds égoïstes dans un WSN, un phénomène qui a une très grande influence sur les performances d'un tel réseau.

D'autre part, vu le problème d'explosion combinatoire que pourrait engendrer ces modèles quand le nombre de messages à transmettre et le nombre de nœuds voisins sont importants, il sera intéressant et bénéfique de proposer des méthodes d'analyse appropriée pour remédier à ce problème.

En plus, il serait intéressant de considérer les types des messages qui circulent sur le réseau et leurres influences sur les performances du réseau et développer une nouvelle version de l'outil TimeNet répondant à certains types d'analyse.

Bibliographie

[1] B. Wang, “Coverage Control In Sensor Networks”, Edition Springer, ISBN 978-1-84996-058-8, 2010.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, “A Survey on Sensor networks”, IEEE Communications Magazine, pages 102-114, August 2002.

[3] J.M.Kahn, R.H.Katz, and K.S.J.Pister. “Next century challenges: Mobile networking for Smart dust”, in Proc Mobicom, pages 483-492, 1999.

[4] Xbow: Crossbow technology inc. http://www.xbow.com/, dernière visite 22/04/2010.

[5] Crossbow: Micaz data sheet http://www.xbow.com/Products/Product_pdf_files/ Wireless_pdf/MICAz_Datasheet.pdf, dernière visite 28/04/2012.

[6] M. Youssef, N. El-Sheimy, “Wireless Sensor Network: Research vs. Reality Design and Deployment Issues”, Communication Networks and Services Research (CNSR), Mai 2007.

[7] http://www.alertsystems.org, dernière visite 28/04/2012.

[8] I.A. Essa, “Ubiquitous sensing for smart and aware environments”, IEEE Personal Communications, pages 47–49, October 2000.

[9] Gérard CHALHOUB, “Les réseaux de capteurs sans fil”, Clermont Universit´e, IUT de Clermont-Ferrand, Dpt R&T, Complexe scientifique des C´ezeaux, 63177 Aubi`ere cedex, France

[10] J. R. Artalejo and A. Gomez-Corral. Retrial Queueing Systems: A Computational Approach. Springer Verlag, 2008.

[11] Ludovic SAMPER, Modélisations et Analyses de Réseaux de Capteurs, Thèse de Doctorat, Institut National Polytechnique De Grenoble, Avril 2008.

[12] Chalermek Intanagonwiwat, Ramesh Govindan, and Deborah Estrin. Directed diffusion : a scalable and robust communication paradigm for sensor networks. In MOBICOM, pages 56–67, 2000.

[13] Chalermek Intanagonwiwat, Ramesh Govindan, Deborah Estrin, John Heidemann, and Fabio Silva. Directed diffusion for wireless sensor networking. IEEE/ACM Transaction on Networking (TON), 11(1) :2–16, 2003.

[14] Vijay Raghunathan, Curt Schurgers, Sung Park, and Mani B. Srivastava. Energy-aware wireless microsensor networks. IEEE Signal Processing Magazine, 19(2) :40-50, March 2002. 2, 13, 14, 22

[15] Ya Xu, John Heidemann, and Deborah Estrin. Geography-informed energy conservation for ad hoc routing. In Proceedings of the 7th annual international conference on Mobile Computing and networking (MobiCom'01), pages 70-84, New York, NY, USA, 2001. ACM. 13

[16] Hurni Philipp and Torsten Braun. Calibrating wireless sensor network simulation models with real-world experiments. In Proceedings of the 8th International IFIP-TC6 Networking Conference (NETWORKING'09), volume 5550, pages 1-13, Singapore, May 5-6 2009. Springer. 14, 36

[17] Gregory J. Pottie andW. J. Kaiser. Wireless integrated network sensors. Communications of the ACM, 43(5) :51-58, 2000. 14

[18] Rahim KACIMI, Techniques de conservation d'énergie pour les réseaux de capteurs sans fil, Thèse du doctorat de l'université e de Toulouse, Septembre 2009.

[19] Cesare Alippi, Giuseppe Anastasi, Cristian Galperti, Francesca Mancini, and Manuel Roveri. Adaptive sampling for energy conservation in wireless sensor networks for snow monitoring applications. In Proceedings of the 4th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS'07), pages 1-6, Pisa, Italy, October 2007. 14, 29.

[20] Wei Ye, John Heidemann, and Deborah Estrin. Medium access control with coordinated adaptive sleeping for wireless sensor networks. IEEE/ACM Transactions on Networking, 12(3) :493-506, 2004. 14, 16, 26.

[21] Mahmood Ali, Annette Böhm, and Magnus Jonsson. Wireless sensor networks for surveillance applications - A comparative survey of MAC protocols. In Proceedings of the 4th International Conference on Wireless and Mobile Communications (ICWMC '08), pages 399-403, Washington, DC, USA, 2008. IEEE Computer Society. 14.

[22] I.F.Akyildiz,W.Su ,Y.Sankarasubramaniam, E.Cayirci, Wireless sensor networks: a survey, Computer Networks: The International Journal of Computer and Telecommunications Networking, v.38 n.4, pp.393-422, 2002.

[23] G.Hoblos, M.Staroswiecki, A.Aitouche, Optimal design of fault tolerant sensor networks, IEEE International Conference on Control Applications, Anchorage, AK, pp.467 –472, September 2000.

[24] Lyes KHELLADI & Nadjib BADACHE, Les réseaux de capteurs : état de l'art,N° LSI-TR0304 , USTHB Février 2004.

[25] Lee Breslau, Deborah Estrin, Kevin R. Fall, Sally Floyd, John S. Heidemann, Ahmed Helmy, Polly Huang, Steven McCanne, Kannan Varadhan, Ya Xu, and Haobo Yu. Advances in network simulation. IEEE Computer, 33(5) :59–67, 2000.

[26] Ben L Titzer, Daniel K Lee, and Jens Palsberg. Avrora : Scalable Sensor Network Simulation with Precise Timing. Proceedings of IPSN, 2005.

[27] Jonathan Polley, Dionysys Blazakis, Jonathan McGee, Dan Rusk, and John S. Baras. ATEMU : A Fine-grained Sensor Network Simulator. Secon, 2004.

[28] Philip Levis, Nelson Lee,MattWelsh, and David Culler. Tossim : accurate and scalable simulation of entire tinyos applications. In SenSys '03 : Proceedings of the 1st international conference on Embedded networked sensor systems, pages 126–137, New York, NY, USA, 2003. ACM Press.

[29] Philip Levis, Sam Madden, David Gay, Joseph Polastre, Robert Szewczyk, Kamin Whitehouse, Alec Woo, David Gay, Jason Hill, Matt Welsh, Eric Brewer, and David Culler. Tinyos : An

operating system for sensor networks. In W. Weber, J. Rabaey, and E. Aarts, editors, Ambient Intelligence. Springer-Verlag, 2005. © Springer-Verlag.

[30] TinyOS Home Page: http://www.tinyos.net.

[31] Victor Shnayder, Mark Hempstead, Bor rong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the power consumption of large-scale sensor network applications. In SynSys, pages 188–200, 2004.

[32] Avinash Sridharan, Marco Zuniga, and Bhaskar Krishnamachari. Integrating environment simulators with network simulators. Technical report, University of Southern California, 2004.

[33] W.Heinzelman, A.Chandrakasan, H.Balakrishnan, Energy-efficient communication protocol for wireless microsensor networks. International conference on parallel processing, pages 156-163, 2001.

[34] Sung Park, Andreas Savvides, and Mani B. Srivastava. Sensorsim : a simulation framework for sensor networks. In MSWIM ’00 : Proceedings of the 3rd ACM international workshop on Modeling, analysis and simulation of wireless and mobile systems, pages 104–111, New York, NY, USA, 2000. ACM Press.

[35] Ahmed Sobeih, Wei-Peng Chen, Jennifer C. Hou, Lu-Chuan Kung, Ning Li, Hyuk Lim, HungYing Tyan, and Honghai Zhang. J-sim : A simulation environment for wireless sensor networks. In Annual Simulation Symposium, pages 175–187, 2005.

[36] Ian Downard. Simulating sensor networks in ns-2, 2005.

[37] Leonard Kleinrock and Fouad A. Tobagi. Packet switching in radio channels : Part i–carrier sense multiple-access modes and their throughput-delay characteristics. In IEEE Transactions on Communications, volume 23, pages 1400–1416, december 1975.

[38] Fouad A. Tobagi and Leonard Kleinrock. Packet switching in radio channels : Part ii–the hidden terminal problem in carrier sense multiple-access and the busy-tone solution. In IEEE Transactions on Communications, volume 23, pages 1417–1433, december 1975.

[39] Ilker Demirkol, Fatih Alag¨ oz, Hakan Delic¸, and Cem Ersoy. Wireless sensor networks for intrusion detection : Packet traffic modeling. IEEE Communications Letters, 10(1) :22–24, January 2006.

[40] Simon Perathoner, Ernesto Wandeler, Lothar Thiele, Arne Hamann, Simon Schliecker, Rafik Henia, Razvan Racu, Rolf Ernst, and Michael Gonz´ alez Harbour. Influence of different system abstractions on the performance analysis of distributed real-time systems. In ACM Conference on Embedded Software (EMSOFT), Salzburg, Austria, October 2007. ACM Press.

[41] Patrick Wûchner, Janos Sztrik, and Hermann de Meer, Modeling Wireless Sensor Networks Using Finite-Source Retrial Queueswith Unreliable Orbit, Scientific Cooperation (HAS&DFG, 436 UNG 113/197/0-1), 2011.

[42] Shi Zhang-song, Computational and Information Sciences (ICCIS), 2010 International Conference, Coll. of Electron. Eng., Naval Univ. of Eng., Wuhan, China, December 2010.

[43] M.Ajmone Marsan. „"Stochastic Petri Nets ,An elementary introduction". Université de Milan .Italie.

[44] G.Florin et S.Natkin, „"Les réseaux de Petri stochastiques"", TSI,Vol N°1,pp 144 – 160 ,Paris 1985.

[45] C.Dutheillet Lamonthesie, „"Symétrie dans les réseaux colorés,Définition,Analyse et Application à l"évaluation des performances"", Thèse de Doctorat,Université Paris VI ,Mars 1992.

[46] M. Diaz, les réseaux de petri- Modèles fondamentaux, Hermès Science Publications, Paris 2001.

[47] J.Sztrik and J.Roszik," performance analysis of finite source retrial ques with non-reliable heterogeneous servers", Journal of Mathematical sciences 146, 6033-6038, 2007

[48] G.W.BRAMS „Réseaux de Petri: Théorie et analyse", Edition MASSON [49] A.BOBBIO: A.PULIAFTITO, M.TELEK AND K.S TRIVEDI, „Recent developments in stochastic Petri nets", journal of circuits, systems and computers vol.8 n°1, 119-158, 1998

[50] N.GHARBI, Evaluation des Performances et de la Fiabilité des Systèmes Multi-classes avec rappel à l'aide des Réseaux de Petri Stochastiques Colorés, thèse du Doctorat, USTHB 2007.

[51] J.W. Cohen, Basic problems of telephone traffic theory and the influence of repeated calls, Phillips Telecommunication Review 18 (1957) 49-100.

[52] C.E.M. Pearce, Extended continued fractions, recurrence relations and two-dimentional Markov process, Advances in Applied Probability 21 (1989) 357-375

[53] J.R. ARTALEJEO, Numerical Calculation of the Stationary Distribution of the Main Multiserver Retrial Queue, Annals of Operations research 116, 41-56, 2002

[54] N.GHARBI and M. loualalen, Performance analysis of retrial systems with servers breakdowns and repairs, Applied Mathematics and compuration 174, 1151-1168, 2006

[55] L. Zerguini, Performance Evaluation of finte-population Retrial Queueing Systems Using Generalized Stochastic Petri Nets, AMS subject classification: 86M20, 86U20

[56] N.GHARBI and M. loualalen, Performance analysis of retrial queueing using Generalized Stochstic Petri Net, In Theory and Practice of Timed Systems (TPTS'02), Grenoble, France 2002, Electronic Notes in Theorical Computer Sciences, 65(6), 2002

[57] J.Sztrik and J.Roszik," performance analysis of finite source retrial ques with non reliable heterogeneous servers", Journal of Mathematical sciences 146, 6033-6038, 2007

[58] J. Wang, J Cao and Q. Li, Reliability analysis of the retrial queue with server breakdowns and repairs. Queueing Systems, vol. 38, pp. 363-380, 2001

[59] N.GHARBI, Modeling and Performance Evaluation of Small Cell Wireless Networks with Base Station Channels Breakdowns, IARIA 978-1-61208-203-5, 2012

[60] G.I. Falin and J.G.C. Templeton. Retrial Queues. Chapman end Hall, London, 1997

[61] Zimmermann, Armin Zimmermann and Michael Knoke. TimeNET 4.0 User Manual, Real-Time Systems and Robotics Group Faculty of EE&CS Technical Report 2007-13, Technische University, Berlin, ISSN: 1436-9915, August 2007.

[62] Prayati A, Antonopoulos C, Stoyanova T, Koulamas C et Papadopoulos G. (2010), A modeling approach on the TelosB WSN platform power consumption, Journal of Systems and Software, vol. 83, No. 8, pp. 1355–1363, 2010.

[63] Ye-Qiong SONG, Protocoles auto-adaptatifs énergie-trafic pour les réseaux de capteurs sans fil, Université de Lorraine – LORIA.

[64] Hao Gu, Performance Analysis of Wireless Sensors Networks, Thèse du Master, université de Concordia, Canada, Avril 2006.

[65] Tie Qiu, Lin Feng, Feng Xial, Guowei Wu1 et Yu Zhou1, A Packet Buffer Evaluation Method Exploiting Queueing Theory for Wireless Sensor Networks, , Dalian University of Technology, China, ComSIS Vol. 8, No. 4, Special Issue, October 2011.

[66] Tamàs Bérczes, Jànos Sztrik, Péter Orosz, Tool supported modeling of sensor communication networks by using finite-source priority retrial queues, Faculty of Informatics, University of Debrecen, Hungary, Carpathian Journal of Electronic and Computer Engineering 5 (2012) 13-18, ISSN 1844 – 9689.

[67] Qiaoqin Li , Mei Yang , Hongyan Wang , Yingtao Jiang et Jiazhi Zeng , A Finite Queue Model Analysis of PMRC-based Wireless Sensor Networks, College of Computer Science and Technology, University of Electronic Science and Technology et Chengdu, Sichuan, P. R. China Department of Electrical and Computer Engineering, University of Nevada, Las Vegas, NV, 89154, 2011.

Résumé

Les réseaux de capteurs provoquent un intérêt croissant au sein des communautés scientifiques et industrielles, dont le but est de surveiller une zone et de prendre régulièrement des mesures par certains nœuds capables de relayer l'information à grande échelle. Ces réseaux assurent de nombreuses applications d'activités militaires et civiles dans divers domaines. Ces réseaux, sans infrastructure fixe, peuvent être déployés de façon rapide, et ce dans des zones sensibles et/ou difficilement accessibles. Cependant, la multitude des services offerts pose une problématique, celle de la complexité d'analyse et d'évaluation des performances de ces réseaux. En effet, avant la mise en place d'un réseau de capteurs, son déploiement nécessite des modèles mathématiques formels précis, permettant leur modélisation et l'évaluation de leurs performances en se basant sur des résultats exacts. Ainsi, nous considérons dans ce projet les réseaux de Petri stochastiques généralisés. D'autre part, certains composants des capteurs pourraient être sujets à des pannes aléatoires. Il est donc important de prendre en considération la non-fiabilité de ces composants lors de la modélisation des réseaux de capteurs. Nous proposons dans ce projet une modélisation du trafic de communication entre un capteur et ses voisins à l'aide des réseaux de pétri stochastiques généralisés afin d'évaluer les performances d'un réseau de capteurs en prenant en compte la non-fiabilité, la taille du buffer, le phénomène de rappel et le changement d'état des capteurs. En effet, l'application des réseaux de Petri stochastiques généralisés dans le domaine des réseaux de capteurs sans fil serait intéressante pour une modélisation des différents aspects et une évaluation des paramètres de performances pertinents.

Mots clés : Réseaux de capteurs sans fil, Evaluation de performances, Modélisation, Réseaux de Petri stochastiques généralisés, Disciplines de panne, Phénomène de rappel, Etat capteur.

Abstract

Sensor networks cause a growing interest in the scientific and industrial communities, whose purpose is to monitor an area and take regular measurements of some nodes can relay information to large scale. These networks provide many applications of military and civilian activities in various fields. These networks without fixed infrastructure can be deployed quickly, and in sensitive and / or inaccessible areas. However, the multitude of services poses a problem, that the complexity of analyzing and evaluating the performance of these networks. Indeed, before the establishment of a network of sensors, their deployment requires precise formal mathematical models, allowing them modeling and performance evaluation based on accurate results. Thus, we consider in this project the generalized stochastic Petri nets. Moreover, some components of the sensors may be subject to random failures. It is therefore important to consider the unreliability of these components when modeling sensor networks. We propose in this project a model of communication traffic between a sensor and its neighbors using the generalized stochastic Petri nets to evaluate the performance of a sensor network by taking into account the unreliability, buffer size, the phenomenon of recall and the change of state of the sensors. Indeed, the application of generalized stochastic Petri nets in the field of wireless sensor networks would be interesting for modeling various aspects and assessment of relevant performance parameters.

Keywords: wireless sensor networks, Performance Evaluation, Modeling, generalized stochastic Petri nets, Disciplines of failure phenomenon reminder, sensor state.