Estimation de la consommation dans les réseaux de capteurs sans fils : étude de cas

Khawla LAHMAR[1], Mohamed Fahmi MEGDICHE[1], Cécile BELLEUDY[2], Mohamed ABID[1], Michel AUGUIN[2]

[1] Laboratoire CES, Ecole Nationale des Ingénieurs de Sfax, BPW 3038, Sfax, Tunisie
khawlal@gmail.com, mfm.mag@gmail.com, mohamed.abid@enis.rnu.tn

[2] Laboratoire I3S, UNCA/CNRS, les Algorithmes-Bâtiment Euclide, 2000 route des lucioles, Sophia Antipolis
belleudy@i3s.unice.fr, auguin@i3s.unice.fr

Résumé : Les Réseaux de Capteurs Sans Fils (RCSF) sont en plein essor. Cependant, le caractère autonomie de ces capteurs leur fait grand défaut et dévoile ainsi la nécessité d'optimiser la consommation de tels systèmes. L'objectif de ce travail consiste à étudier les points clés de la consommation énergétique des nœuds de RCSF. Dans ce but, nous avons tout d'abord essayé de caractériser en consommation ces systèmes. Dans ce cadre, nous avons adopté un environnement de simulation dédié au RCSF. Suite à une étude comparative des environnements de simulation dédiés aux RCSF, nous avons choisi celui qui répond le mieux à nos besoins. Cet environnement a été ensuite mis en place, étudié et testé. Ce travail a focalisé sur l'extension pour la simulation de la consommation de cet environnement : PowerTOSSIM.

Mots clés : Réseaux de capteurs sans fils, basse consommation, TinyOS, TOSSIM, PowerTOSSIM

1. Introduction

Les réseaux de capteurs sans fils qui ont été le sujet de plusieurs recherches depuis quelques décennies, sont devenus des équipements clé dans les applications industrielles et scientifiques. En effet, ces systèmes sont en plein essor. Grâce aux évolutions dans le domaine des réseaux de capteurs et celui des processeurs, on peut imaginer des réseaux denses, sans fils, ayant pour rôles de récolter des données d'un environnement et de les diffuser au sein d'un réseau.

Cependant, l'autonomie des capteurs est un inconvénient du point de vue durée de vie de ces systèmes. En effet, ces derniers sont généralement alimentés par une source d'énergie irremplaçable et non renouvelable. De là émane le besoin impérieux d'une bonne gestion de la consommation.

Dans ce cadre, plusieurs équipes ont focalisé leurs recherches sur la consommation dans les RCSF. Ces travaux touchent d'une part à l'optimisation de la consommation que se soit sur le niveau technologique ou logiciel, ou encore la proposition de nouvelles techniques. Dans le même cadre d'autres travaux visent l'estimation, la caractérisation et la simulation de la consommation de ces systèmes. En fait ces travaux sont complémentaires puisque l'estimation, la caractérisation et la simulation de la consommation permettent d'avoir des pistes pour l'optimisation de la consommation.

Ce papier présente une étude du sujet en se focalisant sur l'estimation de la consommation par simulation. Nous présentons une étude de cas qui s'appuie sur un environnement basé sur le système d'exploitation TinyOS. Une évaluation de cet outil a été ensuite expérimentée à travers un exemple d'application.
Dans cet article, nous proposons, en premier abord, une introduction aux RCSF. Dans un second lieu, une étude comparative des simulateurs de RCSF sera exposée. Suite à cette étude, une plateforme de travail a été adoptée, présentée et testée.

2. Généralités sur les réseaux de capteurs sans fils

Un réseau de capteurs sans fils (RCSF) «Wireless Sensor Network : WSN» est un réseau Ad-hoc comportant un grand ensemble de nœuds. Ces nœuds sont conçus pour contrôler en coopération et d'une manière autonome les conditions physiques et environnementales dans différentes parties de l'espace où ils se trouvent implantés. Les nœuds de ce réseau sont des micro-capteurs capables de communiquer entre eux via une connexion sans fils. La position de ces nœuds n'est pas obligatoirement prédéterminée. Ils sont dispersés aléatoirement dans une zone géographique, appelée champ de capture. Nous présentons dans cette section les caractéristiques de ces nœuds, leurs domaines d'application ainsi que leur architecture interne.

2.1 Caractéristiques et domaines d'applications

La taille de plus en plus réduite des micro-capteurs, le coût de plus en plus faible, la large gamme de types de capteurs disponibles (thermique, optique, vibrations, etc.) ainsi que le support de communication sans fils utilisé, permettent 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 la surveillance, le contrôle et l'automatisation. Les RCSF peuvent se révéler très utiles dans de nombreuses applications lorsqu'il s'agit de collecter et de traiter des informations provenant d'environnements à accès difficile voire même impossible. Parmi les domaines où ces réseaux peuvent offrir les meilleures contributions, nous citons les domaines : militaire, environnemental, domestique, sanitaire, sécurité etc.
Afin de remplir toutes les exigences de ces domaines d'application et de s'étendre à d'autres domaines, les RCSF sont dotés de caractéristiques uniques parmi lesquelles nous citons : une taille très réduite des nœuds qui s'approche du mm^3, une faible source d'alimentation et de faibles ressources matérielles.

Tableau1. Evolution des capacités matérielles des nœuds [1].

Nœud	WeC	Rene	Dot	Mica	Mica2	Mica2dot	Imote	BtNode
Année	1999	2000	2001	2002	2003	2003	2003	2003
Processeur (Mhz)	4	4	4	4	7	4	12	7
Flash (kb)	8	8	16	128	128	128	512	128
RAM (kb)	0.5	0.5	1	4	4	4	64	4
Radio (kBaude)	10	10	10	40	40	40	460	460
Type de radio	RFM	RFM	RFM	RFM	ChipCon	ChipCon	Zeevo BT	Ericson BT
Microcontrôleur	Amtel	Amtel	Amtel	Amtel	Amtel	Amtel	ARM	Amtel
Extensible	Non	Oui	Non	Oui	Oui	Oui	Oui	Oui

Le tableau 1 illustre les capacités matérielles des nœuds les plus répondus [1]. Nous remarquons à travers ce tableau les faibles ressources matérielles dont disposent les nœuds des RCSF en dépit de leurs évolutions dans le temps d'une plateforme à une autre D'un autre côté les RCSF doivent supporter de sévères conditions environnementales, un grand risque de panne des nœuds ainsi que des ruptures de communication surtout que ces nœuds travaillent sans supervision.

2.2 Architecture interne des nœuds de RCSF

L'architecture interne d'un nœud d'un RCSF comprend souvent quatre unités principales : l'unité de capture, l'unité de traitement, l'unité de transmission et l'unité de contrôle d'énergie.
Nous trouvons à l'entrée de la chaine d'acquisition, l'unité de capture composée de deux sous-unités : le capteur lui-même et un convertisseur Analogique/Numérique. Le capteur fournit des signaux analogiques à partir du phénomène observé, au convertisseur Analogique/Numérique. Ce dernier les transforme en un signal numérique compréhensible par l'unité de traitement.
L'unité de traitement comprend un processeur associé généralement à une petite unité de stockage et fonctionne à l'aide d'un système d'exploitation spécialement conçu pour ces micro-capteurs. Cette unité effectue tous les traitements internes du nœud. Alors que l'unité émission/réception assure toutes les communications entre les nœuds.
Enfin, nous trouvons l'unité de contrôle d'énergie qui alimente les trois unités précédemment présentées et qui forment le capteur. En effet, possédant toutes ces unités électroniques, le nœud a nécessairement besoin d'une ressource énergétique (généralement une batterie) pour alimenter tous ses composants. Seulement, faute de taille très réduite ou l'impossibilité d'accès au capteur, la ressource énergétique dont il dispose est limitée et généralement irremplaçable. Ainsi, l'unité de contrôle d'énergie constitue l'un des systèmes les plus critiques nécessitant ainsi un grand intérêt.

2.3 Points clés pour la consommation des RCSF

Comme nous venons de le mentionner, la source d'alimentation des nœuds des RCSF présente un handicap devant l'évolution de ces systèmes. A l'épuisement de la source d'alimentation, le nœud se déconnecte du réseau. En plus, dans le cas de communication multi-hop, la déconnexion d'un nœud engendre la déconnexion de tous les nœuds qui communiquent grâce à lui. Pour ce, nous avons pensé à étudier la gestion de la consommation dans ces systèmes afin de l'optimiser et par suite, augmenter la durée de vie du nœud et du réseau tout entier.
Dans la littérature, il existe plusieurs méthodes qui permettent d'obtenir un système à faible consommation. Généralement elles sont classées en trois catégories suivant qu'elles touchent le matériel et/ou le logiciel.
La première consiste à concevoir des composants spécifiques conçus pour consommer le minimum d'énergie : ceci est réalisé à base des techniques matérielles. Alors que la seconde méthode est basée sur des techniques logicielles qui consistent à optimiser le code afin de réduire sa consommation. Enfin la dernière méthode consiste à réaliser une synergie entre le logiciel et le matériel afin d'optimiser la consommation totale du système. Elle s'appuie sur des mécanismes matériels pour diminuer la consommation

mais utilise aussi le logiciel pour réaliser une adaptation dynamique de la consommation en fonction de la charge courante du système [3].
Mais avant d'étudier et implémenter ces techniques, nous avons besoin d'un environnement de simulation. Ce dernier doit être dédié aux RCSF, fournissant le plus de services, supportant des plateformes qui consomment le moins d'énergie. Et ce qui serait plus intéressant est que ce simulateur nous permette d'avoir une idée sur la consommation dans les nœuds du réseau. Pour ce, une étude comparative des simulateurs de RCSF s'avère nécessaire.

3. Estimation de la consommation par simulation

Le fonctionnement d'un RCSF peut être reproduit de façon virtuelle via un simulateur numérique. Les simulateurs de RCSF sont nombreux et divers. En effet, il existe une vingtaine de simulateurs dédiés aux RCSF. Cependant sont rares ceux qui supportent l'estimation de la consommation de ces systèmes. Nous présentons dans la suite les simulateurs de RCSF les plus répandus.

3.1 NS-2

Ns ou « network simulator » (encore connu sous le nom de ns-2) est un simulateur de réseau à événements discrets. Il est populaire dans le milieu de la recherche par son caractère extensible, sa nature de logiciel libre et la disponibilité d'une documentation riche sur internet. Ce simulateur est plutôt utilisé pour la simulation du routage et de protocoles d'émission/réception et surtout pour la recherche dans les réseaux ad-hoc. En effet, ns supporte plusieurs protocoles de réseaux et permet la simulation de réseaux sans fils et câblés aussi[1]. Récemment, ns2 vient de mettre en place un modèle d'énergie pour les nœuds de RCSF [4]. Ce modèle représente le niveau d'énergie dans chaque nœud à partir d'un niveau initial qui représente l'énergie disponible dans le nœud au début de la simulation. Cette énergie est utilisée pour l'envoie et la réception des paquets jusqu'à l'épuisement.

3.2 OPNET

OPNET (Optimum Network Performance) est un outil de simulation des réseaux qui est très puissant et très complet. OPNET est basé sur une interface graphique intuitive dont l'utilisation et la maitrise sont relativement aisées. En fait, il dispose de trois niveaux hiérarchiques imbriqués : le network domain, le node domain et le process domain [5]. Cependant ce simulateur ne supporte pas de modèle pour la simulation de la consommation.

3.4 OMNeT++

OMNeT++ est un environnement de simulation à événements discrets. Il est basé sur une architecture orientée composants dont les modules sont écrits en C++. OMNeT a

[1] http://www.isi.edu.nsam/ns/

été principalement conçu pour simuler la communication dans les réseaux mais grâce à son architecture flexible et générique, il a été aussi utilisé pour plusieurs autres applications. Malgré qu'OMNeT ne soit pas un simulateur de réseau proprement dit, il est en train de gagner une large popularité comme étant une plateforme de simulation de réseaux auprès de la communauté scientifique ainsi qu'auprès du domaine industriel[2]. Cette popularité est due au fait qu'il fournit plusieurs services dont la modélisation des communications dans un réseau, la modélisation des protocoles de communication ainsi que la validation d'architectures matérielle. En plus, cette équipe focalise ses recherches vers la simulation de la consommation pour les RCSF [7].

3.3 TOSSIM

TOSSIM est le simulateur de TinyOS[3]. C'est un simulateur/émulateur à événements discrets de RCSF accompagnant le système d'exploitation embarqué TinyOS [6]. Il est répandu pour la disponibilité de son code ainsi pour sa nature comme logiciel libre. TOSSIM peut fonctionner sous Linux ou Windows. Son grand avantage par rapport à beaucoup d'autres simulateurs de réseaux de capteurs est qu'il supporte différents types de nœuds de RCSF. En effet, TOSSIM peut simuler le fonctionnement de mica, imote, mica2, mica2-dot, micaz, telos, telos-hc08, telosa, telosb et tmote. En plus, il peut simuler simultanément un nombre très grand de nœuds. Les simulations avec TOSSIM nous donnent une idée sur le fonctionnement du réseau, les émission/réceptions, les liaisons radio entre les nœuds, les messages d'erreurs, etc. Ce qui nous intéresse le plus dans ce simulateur est son extension PowerTOSSIM. Cette dernière permet de simuler la consommation dans chaque périphérique au niveau de chaque nœud du réseau.

3.5 Synthèse

Pour mieux voir les différences entre tous ces simulateurs, nous avons dressé le tableau comparatif suivant (tableau 2). Ce tableau se base sur des critères de comparaison généraux. Nous nous sommes surtout basés sur les critères qui expriment nos besoins expliqués précédemment, c.à.d. un simulateur dédié aux RCSF qui supporte l'estimation de la consommation.
Parmi la diversité des simulateurs qui existent et ceux précédemment cités, seuls TOSSIM, OMNET et très récemment ns-2 s'intéressent à la simulation de la consommation énergétique des RCSF. Cependant OMNET++ n'as pas de modèles prêt, mais en cour de recherche. Et en comparant les modèles d'énergie de NS-2 et TOSSIM, nous remarquons que celui de TOSSIM est plus précis. En effet, PowerTOSSIM modélise l'énergie dans la CPU, l'unité radio, l'unité de capture et les diodes alors que le modèle de NS-2 tiens compte que de la consommation induite des émission/réception. C'est pourquoi nous avons adopté l'environnement TinyOS pour débuter nos travaux visant à optimiser la consommation dans les RCSF. Dans ce cadre, nous présentons dans la partie suivante l'environnement TinyOS.

Tableau 2. Comparatif de simulateurs de réseaux de capteurs sans fils

[2] www.omnetpp.org

[3] La nomination TinyOS désigne d'une part un système d'exploitation conçu spécialement pour les RCSF. D'une autre part, elle désigne l'environnement d'implémentation, de compilation et de simulation d'applications pour les RCSF.

	Ns-2	OPNET	TOSSIM	OMNeT
Licences	GPL	Commerciale	GPL	Académique et commerciale
Architecture	Orientée objet	Orientée objet	Orientée composant	Orientée composant
Langage de programmation	C++	C, C++	nesC	C++/NesC
standard	802.11, Bluetooth …	802.11, 802.16, mobile IP …	802.15.4	802.11 …
Extension pour la consommation	Oui	non	Power-TOSSIM	oui

4. Etude de cas : Estimation de la consommation avec TinyOS

Nous entamons dans cette partie l'étude d'un environnement d'estimation de la consommation par simulation. Suite à l'étude comparative des environnements de simulation les plus répondus, nous avons adopté et mis en place l'environnement TinyOS.

4.1 Présentation de l'environnement TinyOS

L'environnement TinyOS est une plateforme conçue pour l'implémentation, la compilation et la simulation d'applications embarquées fonctionnant en réseau et spécialement pour les RCSF. Cet environnement est composé du système d'exploitation TinyOS, du simulateur TOSSIM, de son extension pour la simulation de la consommation PowerTOSSIM et d'une interface graphique TinyViz. Il est à noter que l'environnement TinyOS tourne sous linux et encore sous Windows via l'émulateur Cygwin.
Le langage de programmation utilisé pour implémenter tous les composants de l'environnement TinyOS est nesC qui est une extension orientée composants du langage C. En effet, ce langage est caractérisé par une architecture basée sur des composants communicants entre eux via des interfaces bidirectionnelles afin de former un exécutable. Grâce à cette architecture basée composants, on peut donc créer une application juste par assemblage des éléments strictement nécessaires soit au niveau système soit au niveau applicatif [8]. L'architecture orientée composants de l'environnement TinyOS est dotée d'une bibliothèque de composants particulièrement complète.
En se basant sur cette architecture et en utilisant les composants fournis par la bibliothèque de TinyOS, nous avons implémenté une application « Temperature » en nesC. Cette application sera utilisée pour valider les performances du simulateur TOSSIM et de son extension PowerTOSSIM.

4.2 Estimation de la consommation avec PowerTOSSIM

PowerTOSSIM est une extension du simulateur TOSSIM de TinyOS qui permet de modéliser la consommation énergétique d'une application. En fait PowerTOSSIM permet de déterminer la consommation énergétique de chaque périphérique de tous les nœuds d'un RCSF munis de TinyOS en se basant sur la formule de l'énergie suivante (1):

$$E=V^* I^*t. \qquad (1)$$

Avec : V : la tension fixe qui alimente le périphérique
I : le courant consommé
t : la durée d'accès à chaque périphérique pendant le temps de simulation.

Ainsi pour déterminer la puissance, il faut déterminer les trois termes de la formule précédente. Le premier terme V est facile à déterminer : c'est la tension fixe qui alimente le capteur. Cette tension peut être déterminée expérimentalement ou tirée directement de la fiche technique du nœud. Mais ce qui est plus compliqué à déterminer c'est le courant I et le temps t. En effet, le courant I dépend fortement du matériel, c.à.d. du type de nœud utilisé puisque tous les types de capteurs ne consomment pas du courant de la même manière. En plus, tous les blocs d'un même nœud ne consomment pas tous la même quantité de courant. Il fallait bien donc établir des modèles d'énergie pour chaque type de nœud.

Tableau 3. Modèle d'énergie du mica2 [9]

Mode	Courant	Mode	Courant	Mode	Courant
CPU		**LEDs**	2,2 mA	TX (-20 dBm)	3,7 mA
Avtive	8,0 mA	**Sensor board**	0,7 mA	TX (-19 dBm)	5,2 mA
Idel	3,2 mA	**EEPROM**		TX (-15 dBm)	5,4 mA
ADC Noise Reduce	1,0 mA	**access**		TX (-8 dBm)	6,5 mA
Power-down	103 µA	Read	6,2 mA	TX (-5 dBm)	7,1 mA
Power-save	110 µA	Read time	565 µS	TX (0 dBm)	8,5 mA
Standby	216 µA	Write	18,4 mA	TX (+4 dBm)	11,6 mA
Extended Standby	223 µA	Write time	12,9ms	TX (+6 dBm)	13,8 mA
Internal Oscilator	0,93 mA	**Radio**		TX (+8 dBm)	17,4 mA
		RX	7 mA	TX (+10 dBm)	21,5 mA

Actuellement, il n'existe qu'un seul modèle d'énergie établi pour mica2, il est illustré dans le tableau 3 [9]. Ce modèle a été établi expérimentalement via un oscilloscope. En fait, il précise exactement la quantité de courant consommée par chaque périphérique du nœud et dans tous les états possibles que peut prendre ce périphérique.

Ce modèle a été établi pour les nœuds de réseaux de capteurs sans fils de type mica2 et les mesures ont été prises via micasb sous une alimentation de 3V. Ce tableau, nous illustre la consommation de la CPU sous les 7 modes de fonctionnement du microcontrôleur ATMEL Atmega128L du mica2. Nous remarquons aussi la consommation importante de l'unité d'émission réception par rapport aux autres unités. Ce modèle d'énergie fournit aussi des informations sur la consommation de l'unité de détection (sensor board), les diodes et l'accès à l'EEPROM.

Une fois ce modèle d'énergie établi, il ne reste qu'à déterminer le temps d'accès à chaque périphérique en précisant son état de fonctionnement. L'approche adoptée par l'équipe de PowerTOSSIM pour déterminer le temps d'exécution de chaque périphérique de chaque nœud est basée sur l'implémentation d'un composant répondant à cette fonction : power state. Power state est un composant de PowerTOSSIM qui détecte l'état de fonctionnement de chaque périphérique de l'application TinyOS et transmet un message pour chaque composant de cette application précisant l'état de

fonctionnement du périphérique. Ce message est ensuite combiné avec le modèle d'énergie précédemment établi pour donner une information détaillée sur la consommation énergétique de tous les nœuds du réseau simulé. En effet, PowerTOSSIM permet de déterminer la consommation de chaque périphérique de chaque nœud en milli joule.

4.3 Implémentation, simulations et résultats

Afin de valider les performances du simulateur précédemment choisi, nous avons implémenté une application en nesC. Cette application permet aux nœuds d'un RCSF de détecter la température dans leurs zones puis d'envoyer cette température à un nœud principal. Le nœud principal reçoit ces températures et calcule la température moyenne. La simulation du fonctionnement de cette application par TOSSIM est présentée à la figure 2. Cette simulation montre : la transmission des températures récupérées par les 9 nœuds du réseau au nœud1 (nœud principal), la réception de ces températures par le nœud principal et le calcul de la moyenne effectué au niveau du noeud1.

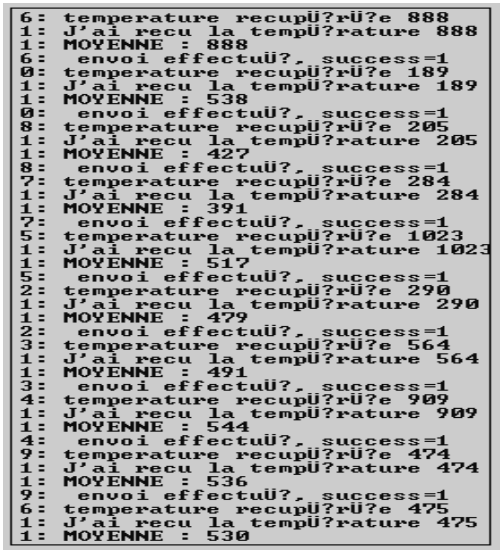

Figure 2. Simulation du fonctionnement de l'application pour un réseau de 10 nœuds.

La simulation de la consommation énergétique de cette application avec PowerTOSSIM pour un réseau de 10 nœuds pendant 60 secondes est illustrée à la figure 3. Cette simulation indique la consommation de chaque périphérique au niveau de chaque nœud du réseau en mJ. A titre d'exemple, le nœud 7 a consommé en totalité 1962.589002mJ, dont 726.143223 consommés par l'unité centrale de traitement et 1236.445778 consommés par l'unité de transmission radio.
Grâce à ces valeurs nous pouvons conclure que l'unité radio est l'unité la plus gourmande en énergie. En effet, elle monopolise presque 60% de la consommation totale du nœud. C'est pour cette raison que plusieurs travaux de recherches visent à optimiser la consommation de ce module. Dans ce cadre, une bonne gestion du fonctionnement de l'unité émission /réception fera la cible dans nos travaux futurs en implémentant des techniques de mise en veille.

Les valeurs données par PowerTOSSIM sont fournies avec une erreur de 4.7% en moyenne [9]. C'est pourquoi nous remarquons des variations de l'énergie consommée suite à des simulations successives de la même application et pour le même temps de simulation. Il faut noter que pour lancer une simulation, il suffit de compiler l'application via Cygwin en fixant la plateforme du nœud visée, déterminer le temps de simulation et le nombre de nœuds.

```
Mote 7, cpu total: 726.143223
Mote 7, radio total: 1236.445778
Mote 7, adc total: 0.000000
Mote 7, leds total: 0.000000
Mote 7, sensor total: 0.000000
Mote 7, eeprom total: 0.000000
Mote 7, cpu_cycle total: 0.000000
Mote 7, Total energy: 1962.589002

Mote 8, cpu total: 710.439698
Mote 8, radio total: 1089.426471
Mote 8, adc total: 0.000000
Mote 8, leds total: 0.000000
Mote 8, sensor total: 0.000000
Mote 8, eeprom total: 0.000000
Mote 8, cpu_cycle total: 0.000000
Mote 8, Total energy: 1799.866169

Mote 9, cpu total: 710.439698
Mote 9, radio total: 1209.684460
Mote 9, adc total: 0.000000
Mote 9, leds total: 0.000000
Mote 9, sensor total: 0.000000
Mote 9, eeprom total: 0.000000
Mote 9, cpu_cycle total: 0.000000
Mote 9, Total energy: 1920.124157
```

Figure 3. Simulation de la consommation de l'application avec PowerTOSSIM

Nous remarquons aussi que lors de la simulation d'un RCSF de 10 nœuds comme illustré à la figure 4, la consommation des nœuds 9 et 10 est toujours plus faible que celle des autres nœuds. Il faut noter ici que la caractéristique illustrée à figure 4 est obtenue suite à 20 simulations successives de l'application « temperature » pour un réseau de 10 nœuds pendant un temps de simulation égal à 60 secondes.

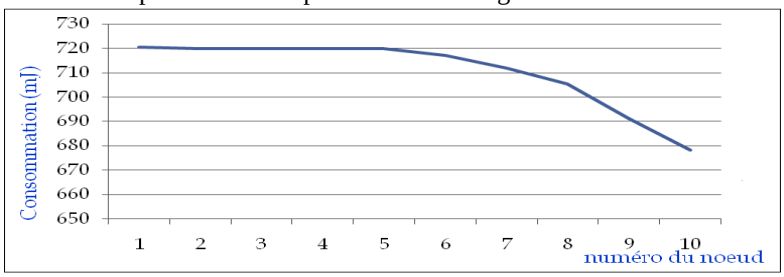

Figure 4. Consommation moyenne de l'unité de traitement d'un réseau de 10 nœuds.

Les variations obtenues sont dues d'une part au temps de simulation qui ne permet pas à tous les nœuds de fonctionner de la même façon et au taux d'erreur du simulateur. En effet, ces variations restent dans l'intervalle du taux d'erreur déclaré par les développeurs de l'outil [9].

Il faut noter aussi que cette variation n'est pas due aux emplacements des nœuds. En effet, TOSSIM suppose que tous les nœuds émettent avec la même puissance, indépendamment de la distance qui sépare les sépare. Ceci est une des limitations du simulateur TOSSIM. En effet, le module émetteur récepteur est le plus consommateur d'énergie, ainsi une erreur d'estimation au niveau de ce module aura un grand impact sur la précision de l'estimation de la consommation totale du nœud.

5. Conclusion :

Le présent travail a traité un sujet d'actualité : l'estimation de la consommation dans les RCSF. Nous avons introduit ce papier par une présentation générale des RCSF. Une étude du problème de la consommation a été ensuite présentée. Cette étude nous a menés à l'optimisation de la consommation par simulation. Dans ce cadre, nous avons entrepris une étude de cas par la mise en place et l'étude de l'environnement TinyOS. Cet environnement d'implémentation, de compilation et de simulation est bien adapté aux RCSF par son architecture orientée composants, son système d'exploitation TinyOS, son simulateur TOSSIM et son extension pour la consommation PowerTOSSIM.
Cette étude nous a permis de dégager des réflexions sur les points de recherche afin d'augmenter la durée de vie d'un réseau de capteurs sans fils.
En effet, grâce à PowerTOSSIM, nous avons localisé l'unité la plus gourmande en énergie : l'unité d'émission/réception. Ainsi dans nos travaux futurs visant à optimiser la consommation des nœuds dans un RCSF, nous focalisons nos recherches sur la gestion du fonctionnement de cette unité.
Néanmoins, il faut utiliser Power TOSSIM avec précaution puisque il n'est pas très précis. En effet, il donne une idée sur la répartition de la consommation dans le réseau et entre les périphériques qui forment le nœud mais les valeurs données ne sont pas très fiables. L'optimisation d'un tel simulateur peut être une perspective très intéressante en ajoutant des modèles d'énergie pour d'autres plateformes de nœuds comme micaz ou imote et surtout en prenant compte la distance séparant les nœuds pour déterminer la puissance d'émission.
Comme autre perspective de ce travail nous envisageons la définition d'algorithmes d'ordonnancement optimisant la gestion de la consommation.

Références

1. Bouillaguet Mathieu et Valero Mathieu. OS pour réseaux de capteurs : TinyOS et Zigbee, 2006
2. Michel Hubin. Traité sur les capteurs et la conception instrumentale, 2000 http://perso.orange.fr/xcotton/electron/coursetdocs.htm
3. Frédéric Parain, Michel Banâtre, Gilbert Cabillic, Teresa Higuera, Valérie Issarny et Jean-Philippe Lesot. Techniques de réduction de la consommation dans les systèmes temps-réels », Mai 2000
4. Le projet VINT : une collaboration entre les équipes de recherche à UC Berkeley, LBL, USC/ISI et Xerox PARC. The ns Manual, le 18 Février 2008
5. Rohit Goyal, Raj Jain, Sonia Fahmy, Shobana Narayanaswamy. Modeling traffic management in ATM networks WITH OPNET,1998
6. Philip Levis and Nelson Lee. TOSSIM: A Simulator for TinyOS Networks. Le 17 Septembre 2003
7. C. Mallanda, A. Suri, V. Kunchakarra, S.S. Iyengar R. Kannan et A. Durresi. Simulating Wireless Sensor Networks with OMNeT++. Le 24 Janvier 2005.
8. David Gay, Philip Levis, David Culler, Eric Brewer. NesC 1.1 Language Reference Manual, May 2003
9. Victor Shnayder, Mark Hempstead, Borrong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the Power Consumption of Large Scale Sensor Network Applications. Division of Engineering and Applied Sciences, Harvard University 2003