



Universidade Federal do Rio de Janeiro

Claudio Miceli de Farias

**A FRAMEWORK FOR DEVELOPING
SMART SPACE APPLICATIONS USING
SHARED SENSOR NETWORKS**

TESE DE DOUTORADO



Instituto de Matemática



Instituto Tércio Pacitti de Aplicações
e Pesquisas Computacionais

UNIVERSIDADE FEDERAL DO RIO DE JANEIRO
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Claudio Miceli de Farias

**A framework for developing Smart Space Applications using Shared Sensor
Networks**

Rio de Janeiro

2014

Claudio Miceli de Farias

**A framework for developing Smart Space Applications using Shared Sensor
Networks**

A thesis submitted in partial fulfillment of the requirements
for the degree of Doctor (Computer Science) in Programa
de Pós-Graduação em Informática, Universidade Federal do
Rio de Janeiro.

Advisors: Luci Pirmez
Flávia Coimbra Delicato

Rio de Janeiro

2014

FICHA CATALOGRÁFICA

F224 Farias, Claudio Miceli

A framework for developing smart space applications using shared sensor networks / Claudio Miceli Farias. – 2014.

179 f.: il.

Thesis (Doctor in Computer Science) – Universidade Federal do Rio de Janeiro, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Programa de Pós Graduação em, Informática, Rio de Janeiro, 2014.

Advisors : Luci Pirmez ; Flávia Coimbra Delicato.

1. Wireless Sensor and Actuator Networks. 2 Shared Sensor Networks. 3. Smart Spaces. 4. Scheduling Algorithms. 5. Fusion Methods.- Thesis. I. Pirmez, Luci (Adv.). II. Delicato, Flávia Coimbra (Adv.). III. Title.

CDD

Claudio Miceli de Farias

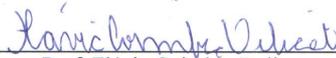
A framework for developing Smart Space applications using Shared Sensor
Networks

A thesis submitted in partial fulfillment of the requirements
for the degree of Doctor (Computer Science) in Programa
de Pós-Graduação em Informática, Universidade Federal do
Rio de Janeiro.

Rio de Janeiro, 20th of March, 2014.



Prof. Luci Pirmez – Advisor
D.Sc., COPPE/UFRJ, Brasil



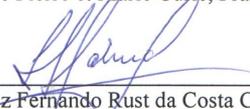
Prof. Flávia Coimbra Delicato – Advisor
D.Sc., COPPE/UFRJ, Brasil



Prof. José Neuman de Souza
Dr., Université Pierre et Marie Curie, França



Prof. Luís Henrique Maciel Kosmalski Costa
Dr., Université Pierre et Marie Curie, França



Prof. Luiz Fernando Rust da Costa Carmo
Dr., UPS, França

Rio de Janeiro

2014

DEDICATION

Dedico este trabalho a minha família, amigos e alunos que me apoiaram, e suportaram o meu interminável mimimi durante esse período.

ACKNOWLEDGEMENTS

This thesis is written in English, but since it is my acknowledgements section I will write it in Portuguese. Sorry... There is always Google translate =).

Agradeço, em primeiro lugar, a Deus pela vida e pela disposição para lutar por meus objetivos. Agradeço pelas pessoas que ele colocou em meu caminho, para que eu pudesse crescer e aprender.

Agradeço à minha família, por ter suportado meu constante mau humor, tristezas e dúvidas esses anos todos e por me apoiarem incondicionalmente, mesmo quando eu mesmo duvidava de mim (o que aconteceu com uma frequência bastante desagradável). Em especial aos meus pais porque sempre buscaram me dar o melhor mesmo em detrimento próprio. Muito obrigado e me perdoem.

Agradeço às professoras Luci Pirmez e Flávia Delicato, por toda a paciência e esforço nos últimos 4 anos. Quase nunca foi fácil, mas elas continuaram tentando quando muitos outros teriam desistido. Devo muito as senhoras.

Agradeço ao Professor Paulo Aguiar por ter me dado oportunidade de trabalhar e aprender quando eu não acreditava em mim. Devo muito ao senhor.

Agradeço a todos os meus amigos e companheiros de curso e do laboratório que suportaram minhas constantes dúvidas, fraquezas e piadas sem graça. Em especial ao Hélio (por me ensinar que às vezes fazer o que se quer é mais importante para o espírito do que fazer o que faz mais sentido), ao Humberto (por mostrar que a família é o mais importante), ao Márcio Galhano (por mostrar que ser feliz é mais importante que estar certo), ao Tiago França (por mostrar que quando você quer alguma coisa sempre há jeito), ao Henrique Ribeiro (por me mostrar que você não deve insistir em algo que não te faz feliz), ao Rafael Costa (por que o trabalho é só uma parte da vida), ao Igor Leão (por mostrar que cansaço é só uma palavra), ao Érico Lemos (por mostrar que o esforço compensa), ao Fabrício Firmino (por mostrar que você faz seu próprio caminho), ao Edgar Delbem (que um grande coração faz milagres), ao Thiago Maluf (por mostrar que confiança é metade do caminho), ao André Abrantes (por que sucesso é estar feliz), a Hebe Duarte (por que alegria é importante), a Emanuele Jorge (por ser meu bom senso), ao Thiago de

Souza (que nós não somos estereótipos), ao Hua Lin (por me ensinar que julgar os outros é perda de tempo), a Cássia Novello (que paciência e trabalho fazem milagres), ao Thiago Delou (por me lembrar de que eu sou mais do que o que eu faço), ao Horácio França (autenticidade não tem preço), ao Paulo Heckmaier (por me ensinar perseverança), ao Rafael Vilar do (follow your dreams, wherever they guide you to), ao Elton (leia as letras miúdas), Thomaz Barros (não há longe), Andressa de Jesus (mantenha o sorriso), Gustavo (mantenha a criança), Franckini (só você sabe o que passa na sua vida), Carina (a vida é uma festa, divirta-se), Paula (personalidade faz diferença), Carlos Eduardo (você conduz a sua vida e não o contrário) e ao Marcílio (ser um mestre é muito mais que um diploma). Obrigado pelo tempo que passamos juntos e pelo que me ensinaram durante todos esses anos. Se esqueci-me de alguém peço desculpas, mas como já disse um grande mestre: “A vida é assim, porque se lamentar?”

Aos meus alunos em especial ao Matheus, Marina, Adriano, Victor, João, Anna, Augusto, Márcio, Gabriel e tantos outros. Aprendi muitos mais com vocês do que vocês comigo. Estou orgulhoso de vocês e desejo toda a sorte do mundo.

Agradeço aos membros da banca: professor José Neuman, ao professor Luís Henrique e em especial aos professores Rust e Lilían que acompanharam minha trajetória desde o começo do mestrado. A todos meu muito obrigado.

Agradeço aos professores do PPGI pelo esforço hercúleo de colocar algum conhecimento em uma cabeça tão teimosa. É claro que não podia deixar de agradecer ao Aníbal, Adriana e a Tia Deise por estarem sempre por perto para me ajudar a resolver os mais diversos problemas.

Agradeço à Universidade Federal do Rio de Janeiro pelo curso que estou concluindo, e a todos seus funcionários do departamento pela dedicação e carinho.

Agradeço, também, a todos aqueles que durante os últimos anos, me causaram dificuldades, intencionais ou não, pois ao superar cada dificuldade imposta, cresci e me tornei algo diferente do que eu era antes.

RESUMO

FARIAS, Claudio Miceli. **A framework for developing smart space applications using shared sensor networks**. 2014. 179 f. Thesis (Doctor in Computer Science) – Programa de Pós-Graduação em Informática, Instituto de Matemática, Instituto Tércio Pacitti de Aplicações e Pesquisas Computacionais, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

A degradação do meio ambiente e do aquecimento global são alguns dos maiores desafios globais enfrentados atualmente. As Tecnologias de Informação e Comunicação (TIC) desempenham um papel vital na solução de problemas ambientais causados pela degradação da natureza. Um dos campos de pesquisa relacionados com a utilização das TIC como provedores de soluções para os desafios ambientais são os ambientes inteligentes. Um dos desafios a enfrentar no contexto de um Ambiente Inteligente diz respeito ao seu monitoramento.

Tradicionalmente, o monitoramento é realizado através de sensores cabeados. Uma alternativa à abordagem de monitoramento tradicional é usar Redes de Sensores e Atuadores sem Fio (RASSF). Nos últimos anos, o campo RASSFs sofreu várias mudanças que impactaram o projeto e operação destas redes. Entre essas mudanças, há o surgimento das Redes de Sensores e Atuadores Compartilhadas (RSACs), que em vez de assumir um projeto de rede específico por aplicação permite que a infra-estrutura de comunicação e sensoriamento sejam compartilhadas entre vários aplicativos.

O fato de que RSAC compartilham o mesmo sensor e infra-estrutura de comunicação entre vários aplicativos faz esse tipo de rede de uma das soluções mais promissoras para aplicações de ambientes inteligentes. No entanto, apesar desse potencial, a adoção de RSACs apresenta novos desafios, que devem ser superados para desfrutar plenamente de seus benefícios. Um primeiro desafio é desenvolver metodologias / mecanismos descentralizados para aplicações para ambientes inteligentes que fazem uso de RSACs para permitir integração de diferentes

aplicações dentro da rede. Outro desafio é como adaptar / conceber novos algoritmos de escalonamento para múltiplas aplicações em RSAC, a fim de fazer melhor uso dos recursos dos sensores. Finalmente, uma terceira é a criação de novos métodos de fusão de dados que consideram os dados coletados a partir de uma mesma fonte e para diferentes aplicações que executam tantas reduções quanto possível, sem qualquer perda de semântica de dados.

Em busca de investigar soluções para os desafios descritos, nesta tese é proposto e especificado um arcabouço, e implementar uma instância desse arcabouço, doravante denominado Asgard, para permitir o compartilhamento da infra-estrutura de sensoriamento e comunicação em RSAC para apoiar o desenvolvimento de aplicações para ambientes inteligentes.

Palavras-chave: Redes de sensores e atuadores sem fio. Redes de sensores compartilhadas. Ambientes inteligentes. Algoritmos de escalonamento. Métodos de fusão.

ABSTRACT

Farias, Claudio M. A framework for developing Smart Spaces Applications using Shared Sensor Networks. Rio de Janeiro, 2014. Thesis (Doctorate in Computer Science) - Programa de Pós-Graduação em Informática, Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2014.

The environment degradation and the global warming are some of the greatest global challenges currently faced. The Information and Communication Technologies (ICTs) perform a vital role in solving environmental problems caused by the nature's degradation. One of the research fields related to the use of ICTs as providers of solutions for the environmental challenges are the Smart Spaces. One of the challenges to be tackled in the context of a Smart Spaces regards its monitoring.

Traditionally, the monitoring is realized through wired communications. An alternative to the traditional monitoring approach is to use Wireless Sensor and Actuator Networks. In recent years the WSNs field as undergone several changes that impacted the design and operation of these networks. Among these changes, there is the emergence of the Shared Sensor and Actuator Networks (SSANs), which instead of assuming an application-specific design allow the sensing and communication infrastructure to be shared among multiple applications.

The fact that SSAN share the same sensing and communication infrastructure among several applications makes this kind of network one of the most promising solutions for Smart Spaces applications. However, despite this potential, the adoption of SSANs poses new challenges, which must be surpassed to fully enjoy its benefits. A first challenge is to develop decentralized methodologies / mechanisms for Smart Spaces applications that make use of SSAN to allow integrating different applications within their network. Another challenge is how to adapt / conceive new scheduling algorithms for multiple applications in a SSAN in order to make better use of the sensors resources. Finally a third is to create new data fusion methods that consider the data collected from the same source and for different applications performing as many reductions as possible without any loss of data semantics

In search to investigate solutions to the outlined challenges, in this thesis we propose and specify a framework, and implement an instance of this framework, hereinafter called ASGARD, to enable the sharing of sensing and communication infrastructure in a SSAN to support Smart Spaces Applications.

Keywords: Wireless sensor and actuator networks. Shared sensor networks. Smart spaces. Scheduling Algorithms. Fusion Methods.

LIST OF FIGURES

Figure 1. Process flow for literature review50

Figure 2. A taxonomy of shared sensor networks54

Figure 3. Logical architecture of the framework76

Figure 4. Collector Nodes Deployment Diagram.....83

Figure 5. Decision Maker Nodes Deployment Diagram.....83

Figure 6. Actuator Nodes Deployment Diagram84

Figure 7. Fusion Manager Subsystem87

Figure 8. Scheduling Manager Subsystem87

Figure 9. Asgard Activity Diagram88

Figure 10. Transmission Tower in the smart grid95

Figure 11 MDFs Behaviour in T1 for each application105

Figure 12 Traditional MDFs Behaviour in T1 applied to both applications simultaneously105

Figure 13 Enhanced MDFs Behaviour in T1 applied to both applications simultaneously106

Figure 14 MDFs Behaviour in T2 for each application107

Figure 15 Traditional MDFs Behaviour in T2 applied to both applications simultaneously107

Figure 16 Enhanced MDFs Behaviour in T2 applied to both applications simultaneously107

Figure 17 MDFs Behaviour in T3 for each application109

Figure 18 Traditional MDFs Behaviour in T3 applied to both applications simultaneously109

Figure 19 Enhanced MDFs Behaviour in T3 applied to both applications simultaneously109

Figure 20 MDFs Behaviour in T4111

Figure 21 Traditional MDFs Behaviour in T4 applied to both applications simultaneously111

Figure 22 Enhanced MDFs Behaviour in T4 applied to both applications simultaneously111

Figure 23 Difference among the results in the real and simulated experiments in relation to the accuracy.....114

Figure 24 Pseudo code of proposed algorithm121

Figure 25 System lifetime with different number of applications125

Figure 26 SSR with a different number of applications125

Figure 27 Comparison of system lifetime126

Figure 28 SSR for different time intervals127

Figure 29 SSR with different levels of packet loss127

Figure 30 The pseudo code of SERAPH136

Figure 31 System lifetime with different number of applications	142
Figure 32 ASR with a different number of applications	143
Figure 33 System lifetime comparison among the selected algorithms	144
Figure 34 ASR for different time intervals	145
Figure 35 The ASR performance of selected approaches under different levels of data loss ...	146
Figure 36 ASR in different stages	147

LIST OF TABLES

Table 1. The final selected sources used in the search stage of this review44
Table 2. The search terms which we used in the online searches45
Table 3. Descriptive analysis and summary of results for included articles51

LIST OF ABBREVIATIONS

ASR – Allocation success rate

SSN – Shared Sensor Network

SSAN – Shared Sensor and Actuator Network

WSAN – Wireless Sensor and Actuator Network

ICT – Information and Communication Technologies

ATN – Actuator Node

CN – Collector Node

DN – Decision Making Node

OPLM – Overhead Power Line monitoring

BM – Battery Monitoring

SSR – Scheduling success rate

HVAC – Heating, Ventilation and Air conditioning

SN – Sink Node

QoS – Quality of Service

Table of Contents

1 Introduction	21
1.1 Hypothesis	27
1.2 Contributions	29
1.3 Organization.....	29
2 Basic Concepts.....	31
2.1 Wireless Sensor Networks	31
2.2 Shared Sensor Networks	33
2.3 Data Fusion for Wireless Sensor Networks	33
2.4 Task Scheduling for Wireless Sensor Networks	44
2.5 Conclusions	42
3 Shared Sensor Network Design to Support Multiple Applications Deployment: A Survey .43	
3.1 Systematic Literature Review	43
3.1.1 Research Questions	43
3.1.2 Search Process	44
3.1.3 Inclusion and Exclusion Criteria	46
3.1.4 Data Collection and Analysis	47
3.1.5 Search Results	48
3.2 Shared Sensor Networks Requirements	52
3.3 Taxonomy of Shared Sensor Networks	54
3.4 Node Level Support	55
3.4.1 Virtual Machine Approaches	55
3.4.2 Middleware Approaches	57
3.5 Network Level Support	58
3.5.1 Resource Management	58
3.5.2 Information Sharing	60

3.5.3 Routing	62
3.6 System Level Support	63
3.7 Applications of Shared Sensor Networks	67
3.7.1 Smart Buildings	67
3.7.2 Smart Grids	68
3.7.3 Food Transportation	69
3.7.4 Health and Medical Applications	70
3.8 Discussion	71
3.9 Conclusions	74
4 A Framework for developing Smart Spaces applications using a SSAN	76
4.1 Framework Logical Architecture.....	76
4.2 Databases Description	79
4.3 Interfaces Description	80
4.4 Physical Architecture	83
4.5 ASGARD's Implementation	85
4.6 Description of ASGARD's Operation	89
4.7 Conclusions	91
5 Multisensor Data Fusion in Shared Sensor and Actuator Networks	92
5.1 Introduction	92
5.2 Motivation Scenario	95
5.3 Enhanced Information Fusion Techniques	97
5.3.1 Enhanced Bayesian Inference	97
5.3.2 Enhanced Dempster-Shafer	97
5.3.3 Enhanced Moving Average Filter	98
5.3.4 Enhanced Fault Tolerant Averaging	99
5.4 Experiments using the enhanced fusion methods	100

5.4.1 Environment Configuration	100
5.4.2 Application Scenario	101
5.4.3 Metrics	102
5.4.4 Evaluating the Accuracy of the enhanced MDFs	104
5.4.5 Experiment to Evaluate the Resource Consumption of WSAN	113
5.4.6 Real node experiments	113
5.5 Conclusions	115
6 A Scheduling algorithm for shared sensor and actuator networks	117
6.1 Introduction	117
6.2 Models and Problem definition	118
6.2.1 System Model	118
6.2.2 Application Model	120
6.3 The proposed Algorithm	120
6.3.1 New DAG formation	122
6.3.2 Sensor Availability	122
6.3.3 Scheduling	123
6.4 Experiments	123
6.4.1 Experimental Settings	124
6.4.2 Parameter Setup	124
6.4.3 Performance Metrics	125
6.4.4 System Lifetime	125
6.4.5 Scheduling Success Rate	126
6.4.6 Comparison with Greedy and Random Task Allocation Schemes	126
6.5 Conclusions	128
7 Adaptive QoS-Aware Service Selection and Allocation for Multiple Applications Execution in Heterogeneous Service-Oriented Wireless Sensor Networks	130
7.1 Introduction	130

7.2 Modelling and Problem definition	133
7.2.1 Service Modelling	133
7.2.2 Application Modelling	134
7.2.3 System Modelling	134
7.3 SERAPH: An Approach for service composition in heterogeneous WSNs	135
7.4 Experiments	140
7.4.1 Experimental Settings	141
7.4.2 Performance Metrics	142
7.4.3 Experiments for evaluating System Lifetime	142
7.4.4 Experiments for evaluating the Allocation Success Rate	143
7.4.5 Comparison with Selected Service Allocation Schemes	145
7.4.6 Adaptability Experiment	147
7.5 Conclusions	148
8 Conclusions	149
8.1 The Answer to the Fundamental Questions	149
8.2 Major Contributions and published articles	153
8.3 Future Works	162
References	164

1. Introduction

The environmental degradation and the global warming are among the major global challenges facing us nowadays. Such challenges are targets of intense research and investment by private and governmental organizations in pursuit of strategies for more efficient energy consumption, as well as for minimizing of the pollutants gas emissions at the atmosphere.

The Information and Communication Technologies (ICTs) perform a vital role in solving environmental problems caused by the nature's degradation. In spite of being part of the problem (they consume energy and are a source of pollution), ICTs have the potential for contributing to major reductions of energy consumption, through the optimization of operations in several areas (such as generation and distribution of electrical energy, traffic control, construction, industrial control) and, consequently, reducing energy waste (WEBER *et al.* 2009). Thus, ICTs assume an important role in the search of solutions for the sustainable and green growth of nations.

One of the research fields related to the use of ICTs as providers of solutions for the environmental challenges are the Smart Spaces (AUGUSTIN *et al.* 2004). A smart space (or pervasive computing environment) can be characterized as an environment with several devices, connected through networks, endowed with processing and sensing capabilities that assist end users to perform their tasks more efficiently.

The Smart Grids (NIST 2010) can be mentioned as one example of Smart Spaces. A Smart Grid is a set of software and hardware tools that enable generators to route power more efficiently, reducing the need for exceeding energy production capacity and allowing two-way, real time information exchange with their customers for real time demand side management (DSM). It improves efficiency, energy monitoring and data capture across the power generation and T&D (transmission and distribution) network (NIST 2010). The Smart Grid uses ICTs for predicting the behavior of the electrical system and, in case of problems (like power outages), to take actions (such as, for example, to activate a battery cooling system) (DENHOLM *et al.* 2005). In the Smart Grid, reliability and online information are indispensable for a safe power supply delivery (FARIAS *et al.* 2012).

In electric power systems today, the safety and reliability have become the most critical issues among all other tasks performed by these systems. System breakdown caused by component

faults, environmental factors, or mis-operation can cause huge economical losses and public concerns. It is commonly known that many of these power grid and facility breakdowns could be avoided or at least largely alleviated if the critical system components were better monitored and the protection devices were better coordinated. With its low cost as one of the main advantages, the Wireless Sensor Networks (WSN) (AKYILDIZ *et al.* 2002) provide a feasible and cost-effective sensing and communication solution for the remote system monitoring and diagnosis systems (GÜNGÖR *et al.* 2010).

A Wireless Sensor Network is a network composed of intelligent sensors, devices that are endowed with processing, storage, sensing and wireless communication resources. The communication capability allows the sensor nodes to be grouped, offering as benefits: (i) redundancy of communication channels that leverages fault tolerance (which does not occur with wired sensing systems); (ii) flexibility of installation and configuration and (iii) low maintenance costs. WSN nodes are devices with an energy source (usually non rechargeable batteries) and limited computational capabilities. WSNs encompass a sheer number of such devices, often in the order of hundreds or thousands, that act collaboratively with the purpose to monitor physical and environmental variables such as temperature, humidity, vibration and light intensity, to name a few. Sensing devices typically used in the context of Smart Spaces are, among others, accelerometers, temperature sensors, humidity sensors and magnetometers. The data acquired by these sensing devices are sent to one or more sink nodes, which are computing devices that do not have the power limitations of the sensors and have higher processing capabilities. Sink nodes are part of a WSN architecture and act as entry points for submitting application requests and as sensing data collection points.

Despite the growing use of WSN in several application domains, sensors are passive devices and highly limited in terms of energy resources, processing capabilities, storage and communication, which basically collect and disseminate data, but do not act on the environment. On the other hand, Wireless Sensor and Actuator Networks (WSAN) (DELICATO *et al.* 2006) include actuator devices capable of performing actions in the physical environment in response to events detected by the sensors. The actuator nodes are able to convert electrical signals into physical phenomenon. In the context of Smart Grids, an example of action can be to shutdown batteries in case of an overheating (DENHOLM *et al.* 2005).

In recent years the WSNs field has undergone several changes that impacted the design and operation of these networks. Among these changes, there is the emergence of Shared Sensor and Actuator Networks (SSAN) (EFSTRATIOU *et al.* 2010) which, instead of taking into account a fit-for-purpose design with the primary aim of supporting a single application that belongs to a single authority (usually the owner of the infrastructure), allows the communication and sensing infrastructure to be shared by multiple applications that may belong to different users, thus optimizing the use of resources. The fact that SSAN share the same sensing and communication infrastructure among several applications makes this kind of network one of the most promising solutions for Smart Spaces applications (XU *et al.* 2010), since without the infrastructure sharing there would be unnecessary replication of the sensing and communication infrastructure as the number of applications increases. However, despite this potential, the adoption of shared sensor networks poses new challenges, which must be surpassed to fully enjoy their envisioned benefits.

One of such challenges is how to efficiently adapt the traditional methods of Multisensor Data Fusion (MDF) (KHALEGHI, KHAMIS, KARRAY and RAZAVI, 2011) to the SSAN scenarios. MDF is a set of methods to enable synergistic combination of sensing data from multiple sensory devices for assisting in the execution of an application by a group of sensor nodes in a way so that it is hardly performed by individual sensors separately. Furthermore, MDF is an effective way to provide optimum utilization of large volumes of data from multiple sensor and sources. By combining information from multiple sensors and sources, MDF methods allow performing inferences about this information that are not feasible from a single sensor or source. Another advantage is that due to the reduction on the number of messages transmitted throughout the nodes, MDFs are able to extend the network operational lifetime. It is important to note that existing MDFs are all designed for an application-specific network design. This means that traditional MDFs process all the sensed data under the specific data semantics of a single target application. Besides, traditional MDFs also assume that all the sensed data are weighted and handled equally and have the same data range. Nevertheless in order to execute properly and efficiently in SSAN scenarios that encompass multiple applications, MDFs have to consider the distinct data semantics of each application. By data semantics we mean a pattern that describes an application and enables interoperability and integration between applications (WAGNER, SPEISER and HARTH, 2010). The problem of dealing with distinct data semantics of each application is particularly important and common in a distributed fusion system (TAPIA, BAJO

and CORCHADO, 2010). If such semantic differences were not taken into account, the fusion techniques would produce unreliable results for different applications. So, by taking the semantics into account, it is possible to enhance the MDF's accuracy. Moreover, in SSAN, in order to avoid the aforementioned problem of processing all applications equally with same priority and data range, MDFs need to consider that the same sensed data may have different degrees of importance for different applications and also different data ranges.

Considering the aforementioned discussion, we claim that existing MDFs are not suitable to be used in a SSAN scenario, since they were not conceived considering the SSAN specific features and needs. Therefore, there is a need for adapting such MDFs to deal with these features in order to achieve energy efficiency and reliable results in this emergent scenario. In this sense, the challenge posed consists in creating new data fusion methods that consider the data collected from the same source and for different applications performing many reductions as possible without loss of data semantics. The first advances in this area, namely the proposal of enhancements of MDF (Multisensor Data Fusion) to deal with multiple applications simultaneously in the SSANs context, were presented by our group in (FARIAS *et al.* 2012). Such paper presented EMAF, an enhanced fusion method (an extension of Moving Average Filter (MARZULLO 1990)) for Shared Sensor and Actuator Networks. EMAF augments the traditional fusion technique to work in the Shared Sensor and Actuator Networks environment, by exploring application similarities on data thresholds. In this thesis, we present our proposed adaptation of several well-known MDFs to the SSAN scenario (EFSTRATIOU, LEONTIADIS, MASCOLO and CROWCROFT 2010), namely, Bayesian inference, Dempster-Shafer inference, False Tolerant Interval and moving average filter.

Another challenge posed by the emergent scenario of SSAN concerns the aspect that executing a larger number of applications can potentially increase energy consumption due to processing and transmissions by the sensor nodes. However, in such a network environment shared by several applications there may be situations where some of these applications can perform common tasks (such as sensing of the same physical or environmental variable) that do not need to be performed several times. Tasks are defined as execution units which make up an application, for example, collection of a specific physical or environmental variable. Beyond that, in this same environment there can be applications with higher priorities, regarding the requested response time (for

example, critical applications) or the amount of resources provided to the application (for example bandwidth, sensing coverage) if compared with others that are sharing the SSAN, and therefore those applications must be executed first. Traditional task scheduling algorithms (ROY *et al.* 2010) (BATCHARYA *et al.* 2010) (WEI *et al.* 2011) are responsible for finding a group of sensor nodes which are most suitable for the execution of a certain task so as to increase the network lifetime. Although several efforts have been made, these works (ROY *et al.* 2010) (BATCHARYA *et al.* 2010) (WEI *et al.* 2011) concern about the case of running an application in a WSN with the only objective to achieve energy efficiency. Task scheduling algorithms designed for SSANs should not only pay close attention to the energy saving by choosing the best node to a given task, but should also perform the tasks common to different applications only once, sharing the result to all nodes to further improve the use of limited resources. If the scheduling algorithm does not properly address this common task issue, it will consume system energy in a less efficient way by repeatedly performing them. Therefore, the challenge is how to adapt/conceive new scheduling algorithms for multiple applications in a SSAN in order to make better use of the sensors. In this thesis, we present two task allocations algorithms for SSANs. The first one, described in (FARIAS *et al.* 2013) is an energy-efficient scheduling algorithm to perform multiple applications in SSANs. We modeled applications as DAGs (Directed Acyclic Graphs) where each node of a DAG is a task and the edges represent the dependencies between tasks. To maintain task dependencies, we combine the DAGs with common tasks into a new DAG without damaging the integrity of each task graph. Once the DAGs are combined, the applications are then allocated to the SSAN based on DAG's composition (SAKELLARIOU 2006). The second one (FARIAS *et al.* 2014) is a task selection and allocation algorithm called SERAPH (adaptive QoS-aware SERvice selection and allocation for multiple APplications execution in Heterogeneous service-oriented WSNs) that efficiently utilizes the underlying resources in SSANs, and yet can provide nodes able to execute desired tasks within a dynamic environment regardless their hardware-level or OS-level differences to the end-users.

Another challenge refers to leverage application integration in SSAN scenarios. Since there is a large number of applications in the SSAN, the decision-making process of an application can influence the decision-making process of another application causing (unpredictable and potentially undesirable) changes in the behavior of both applications. In order to efficiently enable control and decision making for smart space applications (such as Smart Grids), the notion

of integration arises. Integration is defined as the ability to communicate, collaborate and exchange information between applications to achieve common goals (BYUN and PARK 2011). The main purpose of integrating applications on Smart Spaces is to save energy (both from the space and from the SSAN) by reducing worthless task repetitions and fostering collaboration between applications. Such integration further minimizes energy waste by avoiding undesirable states (such as to turn on an exhauster in the occurrence of fire based on a fire detection application instead of an HVAC application). Other advantages of integrating applications are: (i) more efficient use of resources, such as energy, computational and even human, (ii) fast and more coordinated responses to physical events monitored by the network, (iii) the ability to correlate information between applications in order to optimize the decision process, (iv) decision chaining between integrated applications, i.e., a decision made in a given application may trigger another decision on a different application. So the challenge in this context is to develop decentralized methodologies/mechanisms for Smart Spaces applications that make use of SSAN so as to allow integrating different applications within the network. Although characterized as a challenge, the application integration issue is out of the scope of this thesis. Some works done by our research team (FARIAS *et al.* 2013) (FARIAS *et al.* 2014) and (SOARES *et al.* 2012) presented a proposal of a decision making system, called CONDE, that tackles this challenge. CONDE is a decentralized control and decision-making system for Smart Spaces applications, which contributes to improving the energy efficiency of the monitored space. It is decentralized once the whole decision-making process for the applications is performed within the network. Therefore, there is no need of transmitting raw data to a centralized entity (the base station) on a hop-by-hop basis, neither the final decisions back to the network, thus reducing radio transmissions and saving network energy.

These presented challenges (data fusion, task scheduling and application integration) relate to the basic functions necessary for network operation and management. Developers of WSN applications are usually experts in their field of knowledge, not in networks. Therefore, the implementation of routines for managing the use of resources and scheduling of active nodes is a difficult task for such developers. In traditional WSNs, in the same way as in distributed systems, the use of a framework facilitates the work of application developers, freeing them from dealing with the complexity of distribution. A framework is an abstraction in which software providing generic functionality can be selectively changed by additional user-written code, thus providing

application-specific software. A framework for WSNs should be situated between applications and the operational system performing infrastructure services. Shared Sensor Networks could also benefit from the use of a framework that provides an environment for the implementation and execution of applications, providing abstractions of the infrastructure network functionalities, such as the outlined as current challenges in the field.

In this thesis we propose and specify a framework, and implement an instance of this framework, hereinafter called ASGARD, to enable the sharing of sensing and communication infrastructure in a SSAN to support Smart Spaces Applications. The proposed framework includes various subsystems identified as necessary for an environment of SSAN adapted to Smart Spaces. By adopting the proposed framework it will be possible for multiple applications to share a SSAN, seeking to allocate tasks and performing data fusions of multiple applications that make up the Smart Spaces in order to save resources (both the network and the environment) and get more accurate answers on the state of Smart Spaces.

1.1. Hypothesis

The objective of this work consists on building a framework for Smart Spaces using SSAN. In addition to the design and construction of the framework's architecture, this work has also the objective of proposing solutions to the data fusion and scheduling challenges previously mentioned. Such solutions will be implemented as components present on the subsystems that integrate the proposed framework. The integration is out of the scope of this thesis. The hypotheses considered in this thesis are described below.

Hypothesis 1: A framework for developing Smart Spaces applications using SSAN should: (i) be able to deal with the different requirements of multiple applications simultaneously; (ii) schedule tasks from multiple applications; (iii) execute data fusion for multiple applications; (iv) make decisions about their behavior; (v) be scalable; and (vi) maintain the decision making process as much as possible within the network.

To provide an integrated and scalable solution, this thesis proposes a decentralized framework which incorporates new data fusion methods and scheduling algorithms appropriate to the shared sensor networks scenario. Data fusion methods found in the literature will be evaluated; problems/limitations that traditional methods present when dealing with multiple applications and

new solutions for the SSN will be presented. We will also evaluate the traditional scheduling algorithms and present new scheduling algorithms tailored to the SSAN scenario.

Based on the presented hypothesis, the following key issues were defined to guide the research:

- What are the requirements of a framework able to manage the decisions made in a SSAN?

Hypothesis 2: Data fusion techniques applied to SSAN must maintain data semantics and accuracy.

This hypothesis relates to the data fusion challenge for SSANs previously described, and the following question was defined to guide the investigation of this hypothesis:

- How to execute semantic data fusion involving multiple applications, so that the result of the data fusion is accurate, has the least possible delay and a high fusion degree?

Hypothesis 3: Algorithms for SSAN scheduling should take into consideration the applications priority and the dependencies between tasks of an application, as well as common tasks among various different applications.

This hypothesis relates to the SSAN task scheduling challenge described above and the following key issues were defined to guide the investigation of this hypothesis:

- How to schedule multiple applications so as to avoid activities to be repeated unnecessarily?

Hypothesis 4: Decision Systems for SSAN should take application integration into consideration, as well as trying to maintain the decision in-network.

This hypothesis relates to the SSAN decision system challenge described above and the following key issues were defined to guide the investigation of this hypothesis:

- How to build a decision system so as to integrate decisions and maintain the decision as much as possible in-network?

1.2. Contributions

The aim of this thesis is to propose a decentralized control and decision framework for SSAN, especially considering the issues of task scheduling and data fusion for multiple applications. Specific contributions of the thesis are listed below.

- Present a survey about Shared Sensor and Actuator Networks. This survey presents the new tendencies, challenges and a taxonomy about the Shared Sensor and Actuator Network design;
- Develop a decentralized framework for developing applications in shared sensor networks. All the components that are part of the framework, as well as their interaction and their operation will be presented. The development of an instance the proposed framework will also be covered by this work.
- Propose enhancements of well-known MDFs to make them able to handle multiple applications simultaneously. This thesis encompasses enhanced methods of Bayesian inference, Moving Average Filter, Dempster-Shafer, Average Fault Tolerant.
- Propose new task scheduling algorithms able to work in the SSAN scenario. Algorithm that takes advantage of the formation of forests, services and roles for task scheduling in SSAN will be presented.

1.3. Organization

Chapter 2 describes the basic concepts necessary for understanding this work. This chapter presents the concepts of wireless sensor networks, shared sensor networks, smart environments, data fusion techniques for wireless sensor networks and scheduling algorithms.

Chapter 3 presents the state of the art on the main topics discussed in this thesis. The articles dealing with shared sensor networks and all its aspects are presented through a systematic review.

Chapter 4 describes the proposed framework, its architecture and key components. It is also presented in this chapter an instance of the framework now called ASGARD.

Chapter 5 presents the new data fusion methods for shared sensor networks and tests are made to prove its accuracy and energy consumption reduction.

Chapter 6 presents a new scheduling algorithm for Shared sensor networks that exploits common tasks to make effective task-sensor assignments explicitly take full advantage of applications with common tasks and avoid repeating tasks unnecessarily.

Chapter 7 presents another new scheduling algorithm for Shared Sensor networks working on multiple SSAN that besides exploiting common services to make energy-efficient service-sensor assignments. In addition, it explicitly takes full advantage of fuzzy system and sensor roles to adapt the given network condition changings.

Chapter 8 presents the conclusions and future work. There are in this chapter discussions about the results, and next steps of the work.

2. Basic Concepts

This chapter describes the basic concepts that support the themes and are necessary to understand the proposed work. The chapter is organized as follows: (i) section 2.1 presents the basic concepts about wireless sensor networks; (ii) section 2.2 about shared sensor networks; (iii) section 2.3 about data fusion for wireless sensor networks; (iv) section 2.4 about task scheduling and (v) section 2.5 presents some conclusions.

2.1. Wireless Sensors Network

In the micro technology context, the Wireless Sensor Networks (WSN), unlike the conventional sensor networks (wired and analogical sensor network), are composed of tens to thousands of low-cost and reduced-size battery powered devices, which are capable of information sensing, processing, and transmission through a wireless network (DELICATO *et al.* 2005). The sensors collect data gathered by the sensor units from monitored physical structures (bridges, buildings, factories, transmission lines, etc.) and transmit to one or more exit points of the network, called sinks, to be analyzed and processed. The sensors act collaboratively in order to obtain more reliable data and communicate with each other (multihop communication) to bring the sensed information to the sink. The presence of a large number of low-cost sensor nodes in these networks enables, on the one hand, to increase the sensed area and, on the other hand, to increase the application robustness when facing the failure of the devices. However, as sensor nodes are battery operated, it is extremely important that the protocols and algorithms are designed for the efficient use of energy to maximize the system life cycle.

The dynamic nature of WSN in terms of both organization and topology requires them to have the capacity to self-organize and self-configure (DELICATO *et al.* 2006). A WSN must adjust to the loss and insertion of sensor nodes when: (i) there occurs energy depletion of their batteries, (ii) they are disabled, in other words, put to "sleep" to save energy, (iii) they are activated, in other words, "awakened", (iv) new sensors are added to the network, and (v) there is a failure.

In addition, the WSN should be adaptable since both the environment conditions, in which a network is installed, and the application's requirements may change. In other words, sensor networks must operate in a robust and decentralized manner, adapting when the environment

changes (MOSTARDINHA 2006; MOSTARDINHA 2007). Additionally, sensor networks must operate autonomously, in other words, without human assistance for long periods and without any human intervention from base stations and other sensors due to: (i) the huge amount of sensors generally existing in a network and (ii) its installation being performed in areas of difficult access.

An advantage of employing a WSN rather than a wired sensor network in a given application is that a WSN reduces costs by reducing the number of inspections into the monitored structures (periodic inspections performed by experts in order to verify if the structures need repair) and thus the costs of these inspections. Another advantage consists in the possibility of installing sensors in locations of difficult access without the need to install cables in these locations. Other benefits with the use of wireless sensor networks are mainly related to the improvement of the collected data in a shorter time period and their higher reliability when compared to wired monitoring in certain scenarios such as smart grids (GUNGOR *et al.* 2010). However, different from wired sensors that could receive energy from the cables, power consumption becomes a crucial factor, as it will determine the lifetime of such networks (DELICATO *et al.* 2004) (DELICATO *et al.* 2005a) (DELICATO *et al.* 2005b). Mechanisms to extend the WSN lifetime have become a challenge in the field present in various levels of the protocol stack.

Two mechanisms are essential to extend WSN lifetime: the task scheduling, and aggregation and/or data fusion inside the network. The task scheduling mechanism is responsible for allocating the tasks that will be performed by a sensor node in the network; in other words, it is responsible for defining in which nodes the tasks of an application will be allocated. Considering the mechanism of aggregation and/or data fusion inside the network, it is used in order to reduce power consumption by preventing unnecessary transmissions and making them more effective.

Traditionally WSN was designed for a single purpose, a single application. Specifically, each network node was programmed to collect and process a specific data type using protocols and custom data format. This approach is known as fit-for-purpose (EFSTRATIOU *et al.* 2010).

While this is an approach for short-term and small-scale deployments, in sensor network deployments that consist of thousands of nodes with a life span of multiple years, inducing high costs of deployment and maintenance. The single-application approach can lead to inefficient use

of resources and low cost-benefit results, since each new application would represent the requirement for dedicated sensing infrastructure to support new applications belonging to different organizations that can lead to unnecessary replication of sensing infrastructure. These factors led to development of shared sensor networks (SSN).

2.2. Shared Sensors Networks

A shared sensor network is a WSN which serves as a flexible infrastructure capable of supporting resource sharing between applications. Users can submit new applications to the shared sensor network through the base station. Applications can be deployed dynamically at different times based on user demand. Furthermore, different applications can have different priorities according to their importance. Node sensors can be heterogeneous in terms of supported sensors and resources (such as memory, processing ability, among others). A shared sensor network works as a highly flexible infrastructure that supports different levels of resource sharing between applications. For example, multiple applications can share (1) one sensing unit in a sensor node (for example, a magnetometer can be used to detect parked cars and to detect moving vehicles), (2) a sensor node with multiple sensing units and (3) one network with multiple applications on different sensor nodes. The SSN represents a total decoupling of applications and physical infrastructure of sensing and transmission (EFSTRATIOU *et al.* 2010).

2.3. Data Fusion for Wireless Sensor Networks

Generally data fusion can be seen as "a multilevel process for dealing with the detection, association, correlation and estimation of data from multiple sensors "(USA Department of Defense 1991). In the WSN domain, simple data aggregation techniques (arithmetic averages, the search for maximum and minimum, among others) have been used to reduce the data traffic in order to reduce the energy consumption of sensor nodes. The data aggregation can be defined as the data combination from different source nodes using trivial functions (i.e., maximum, minimum, average) that suppress redundant messages and consequently reduces the amount of data. The efficiency of data aggregation algorithms depends on the correlation between the data generated by the different sources of information (FASOLO *et al.* 2007). The correlation can be spatial, when the generated values by near sensors are related; temporal, when readings of sensors change slowly over time, or semantic, when information in different data packages can be classified under the same semantic group, such as data that is generated by sensors placed in the

same room. This aspect favors the elimination of redundancy (one of the goals of the data aggregation techniques), but also ensures data accuracy. This is important, because the data summarization may represent a loss in accuracy (NAKAMURA; LOUREIRO; FRERY, 2007), which is a typical requirement for many WSN applications. The accuracy can be defined as the degree of proximity between the observed measurement and its real expected value. With an efficient correlation of the original data it is possible to achieve a larger reduction of the amount of data for the same accuracy of the aggregated data.

Other two important concepts for data aggregation mechanism efficiency are: degree and latency. The degree of aggregation is defined as the number of aggregated packets in a single packet transmission, while the latency can be measured as the time between received packets in a sink node and the data generated in the source nodes (RAJAGOPALAN; VARSHNEY, 2006). It is important that the relationship between these two concepts is balanced so that there is efficiency in the reduction of the data amount on the one hand, but on the other hand so that there is no excessive delay in the final data delivery. The data fusion can be categorized according to several aspects, namely: the relationship between data sources, level of abstraction and the purpose of data fusion. According to the relationship between the data sources, the data fusion can be classified as complementary, redundant and cooperative (DURRANT and WHYTE 1988).

- Complementary - When the information provided by the sources represents pieces of a bigger scenario, the fusion can be applied to obtain a more complete amount of information on the scenario. The complementary fusion searches completeness, forming new information by joining several other sources (such as sensors that check for the presence of people in four different corners of a room and by fusion this information we have the full view of the room).
- Redundant - If two or more independent sources provide the same piece of information, these pieces can be merged to increase the information's reliability. The redundancy fusion can be used to increase the reliability, accuracy and credibility of the information. In WSN, the fusion of redundancy can provide high quality information and prevent sensor nodes from transmitting identical data (various temperature sensors evaluating the temperature of an industrial boiler).

- Cooperative - Two sources are cooperative when the information provided by them is merged into a new piece of information (usually more complex than the original) which, from the application's point of view, better represents the reality (A temperature sensor and a smoke sensor combining information to detect a fire).

Regarding the level of abstraction (LUO *et al.* 2002), data fusion can be classified into four levels as described below.

- Signal (signal) – it deals with uni or multi-dimensional signals coming from the sensors (usually raw data coming from sensors). Signal can be used in real time applications or as an intermediate step between fusions.
- Pixel – it is used in images that can be used in multimedia processing tasks.
- Characteristic (feature) - it deals with characteristics (or attributes) extracted from signals (such as the temperature of a room) and images, such as shape and speed.
- Symbol (symbol) - in this type of fusion, the information is a symbol that represents a decision (for example, a symbol indicating the alarm trigger action in case of fire), and therefore, this type of fusion is also known as fusion of decisions.

According to the level of abstraction of the manipulated data, the fusion of the information also can be classified into 4 categories: Low-level fusion, Mid-level fusion, High-level fusion and Multilevel Fusion.

- Low-level fusion - Also known as signal level fusion. Unprocessed data is used as fusion input; combining into new data that is more accurate than the original (this includes data extracted from the sensing units such as voltage, amperage or electromagnetic field).
- Mid-level fusion - attributes or characteristics of an entity (such as shape, texture and position) are merged to obtain a map of characteristics. This type of fusion is also known as fusion of attributes (temperature and electromagnetic field data for finding damage on transmission lines).
- High-level fusion - decisions or symbolic representations are used as input and are combined to obtain a decision that is more reliable and has a broader view of the scenario (for example, the decision that a transmission line is damaged and the decision that the battery is damaged generating the decision to use another power transmission line).

- Multilevel Fusion -it happens when the data fusion uses data from different levels of abstraction (for example, when symbol data is merged with characteristic data type, such as data coming from a presence sensor, indicating that there are no people in the room, combined with a heating system decision while carrying out the merge may conclude that an air-conditioner should be turned off).

Dasarathy *et al* (DASARATHY *et al.* 1997) present another well-known classification for data fusion that considers the input and output data abstraction. Dasarathy identifies five categories: Data In-Data Out (DAI-DAO), Data In-Feature Out (DAI-FEO), Feature In-Feature Out (FEI-FEO), Feature In-Decision Out (FEI-DEO) and Decision In-Decision-Out (DEI-DEO).

- DAI-DAO - In this category, the data fusion deals with data at the signal level and the result also at the signal level, possibly more accurate or reliable.
- DAI-FEO - In this category, the data fusion uses raw data as input to extract attributes or characteristics that describe an activity as output.
- FEI-FEO - In this category, the data fusion works over a set of characteristics to improve or refine a characteristic or attribute, or to extract new ones.
- FEI-DEO - In this category, the data fusion uses a series of an entity's features, generating a symbolic representation or a decision.
- DEI-DEO - In this category, decisions can be merged to obtain new decisions or give emphasis to previous decisions.

Yet another form of classification is based on the purpose of the fusion methods, in other words, what kind of information one seeks to extract from the collected data (NAKAMURA *et al.* 2007). According to this criterion the data fusion can be performed with different purposes such as inference, estimation, classification, aggregation and compression.

Inference methods are often applied in decision mergers. In this case, a decision is made based on knowledge of the perceived situation. The inference refers to a transition from a proposition that is probably true, from which truthfulness is credited as a result of an earlier inference. Classic inference methods are the Bayesian inference (BAYES 1863) and the theory of accumulation of beliefs by Dempster-Shafer (DEMPSTER and SHAFER 1975).

Compression and aggregation methods are only used to reduce the data amount. Aggregation is used to solve the implosion and *Overlapping* problems. In the former, the data sensed is duplicated on the network due to some routing strategy. *Overlapping* happens when two different nodes disseminate the same data. The compression methods are not data fusion methods, per se, since they only consider the data coding strategies. The Huffman code fits in the methods of data compression.

2.3.1. Traditional Data Fusion

In this section, the traditional fusion methods that will be extended in this thesis will be presented. These methods were chosen for their importance and application in the wireless sensors network area (KHALEGUI *et al.* 2011). They are: Bayesian Inference, Dempster-Shafer, Moving Average Filter and Fault Tolerant Average Algorithm.

A. Bayesian Inference

The Bayesian inference uses a combination of evidence according to certain probability rules. The uncertainty degree is represented by conditional probabilities as shown in equation 1:

$$\Pr(Y|X) = \frac{\Pr(X|Y)\Pr(Y)}{\Pr(X)} \quad (1)$$

Where $\Pr(Y | X)$ represents the probability that hypothesis Y is true given the information X. This probability is obtained by multiplying $\Pr(Y)$, the previous probability of hypothesis Y, by $\Pr(X | Y)$, the probability that X is true given that Y is true, $\Pr(X)$, the previous probability of hypothesis X, can be considered a normalizing constant. The greatest problem of Bayesian inference is that the probabilities $\Pr(X)$ and $\Pr(X | Y)$ must be known a priori without any guarantee of the real values. However, the Bayesian inference is used in the WSN scenario due to its simplicity and low resource consumption.

B. Dempster-Shafer

This method is based on the theory of accumulation of beliefs (chances of being in a certain state in a given time), a theory introduced by Dempster-Shafer (DEMPSTER and SHAFER 1974) that generalizes the Bayesian theory. Dempster-Shafer deals with beliefs or mass functions in the same way that Bayesian theory deals with probability. The probability theory provides a

formalism that can be used for the representation of incomplete knowledge, combination of evidence and updating beliefs.

A fundamental concept in the Dempster-Shafer inference system is the discernment frame, which is defined as follows. Taking $\theta = \{\theta_1, \theta_2, \dots, \theta_N\}$ as the set of all possible states that describe the system, θ is mutually exclusive and exhaustive meaning that the system is in only one given state θ_i , where $1 \leq i \leq N$. The interval from 1 to N is called the discernment frame because its elements are used to discern the system's states.

The elements of the 2^θ set are called hypotheses. In the Dempster-Shafer theory, based on the E evidence, a probability is associated to each hypothesis $H \in 2^\theta$, according to the function of the mass m that satisfies the following conditions:

For $\theta = \{\theta_1, \theta_2, \dots, \theta_N\}$ the set of all possible states, the H hypothesis is the set such that $H \in 2^\theta$.

$$\begin{cases} m(0) = 0 \\ m(H) \geq 0, \forall H \in 2^\theta \\ \sum_{H \in 2^\theta} m(H) = 1 \end{cases} \quad (2)$$

The degree of belief of a hypothesis is defined by the function $bel(H)$ as seen below:

$$bel(H) = \sum_{A \subseteq H} m(A) \quad (3)$$

From this belief we can calculate two other degrees of belonging: doubt (dou) and plausibility (pl) as shown in equations 4 and 5:

$$dou(H) = bel(\neg H) = \sum_{A \subseteq \neg H} m(A) \quad (4)$$

$$pl(H) = 1 - dou(H) = \sum_{A \cap H = 0} m(A) \quad (5)$$

Thus, to match the effect of two sets of probabilities (m_1 and m_2) over the same hypothesis, the Dempster-Shafer theory defines the following combination rule:

$$m_1 \oplus m_2(H) = \frac{\sum_{X \cap Y = H} m_1(X)m_2(Y)}{1 - \sum_{X \cap Y = 0} m_1(X)m_2(Y)} \quad (6)$$

In Dempster-Shafer inference, probabilities are not associated with the hypotheses a priori. Instead, probabilities are associated only when the supporting information is available, in other

words, when there is collected data to be evaluated. On the other hand, the Dempster-Shafer inference is totally dependent on the collected data, which means that if there is too little collected data the result can be inaccurate or tendentious. To overcome this problem, the minimum amount of incoming data required for each application should be analyzed.

To choose between the Bayesian inference and Dempster-Shafer is not a trivial decision, since there is a tradeoff between the accuracy of Bayesian inference and flexibility brought by Dempster-Shafer (NAKAMURA *et al.* 2007). In a non static scenario (like the SSN), the accuracy of Bayesian inference is reduced due to changes in the environment. In this case, the flexibility brought by Dempster-Shafer inference becomes interesting, since the belief function will adapt to the new data while the probabilities of Bayesian inference remain static. (KHALEGUI *et al.* 2011)

C. The Moving Average Filter

The moving average filter is a method widely used in digital signal processing because it is simple and capable of reducing signal noise. The filter computes the arithmetic average of a number of entry signals to produce each point of output signal. Given a signal $z = (z_1, z_2, \dots)$, the true signal $x = (x_1, x_2, \dots)$ is estimated by:

$$x(k) = \frac{1}{M} \sum_{i=0}^{M-1} z(k-i), \forall k \geq M. (7)$$

where M is the fusion window size, $z = \{z_1, z_2, z_3 \dots\}$ is the input data and $x = \{x_1, x_2, x_3 \dots\}$ is the data estimated by the method.

The fusion window is the most important parameter for the equation since M is used for the detection of any change in signal level; the larger the M value, the clearer the signal will be.

Although easy to understand and not very complex, the moving average filter deals only with measurements at the signal and feature levels, but cannot handle higher semantic levels, such as decisions.

D. Fault-tolerant Average

The fault-tolerant Average algorithm was introduced by (MARZULLO 1996) for the synchronization of distributed systems. Thereafter, it has been used in the fusion data domain to

merge the data sensed by a set of n **real** sensors to form a more reliable **abstract** sensor that is correct even when some of the original sensors are incorrect. The **reliable abstract** sensor concept was introduced by Marzullo to define one of three types of sensors: **concrete**, **abstract** and **reliable abstract**. A **concrete** sensor is a real device that monitors a physical variable of an environment. The **abstract** sensor consists of a range of values that represent the observation provided by a **concrete** sensor. Finally, the **reliable abstract** sensor is the interval (or the set of intervals) that contains the real value of the physical variable monitored.

The algorithm assumes that up to f out of n sensors are incorrect in their measurements, where f is a parameter. Let $I = \{I_1, \dots, I_n\}$ be the set of intervals $I_i = [x_i, y_i]$ provided by n **abstract** sensors referred to the samples of a given physical variable (such as temperature) taken at the same measurement interval, in other words, different views of the environment provided by n sensors for the same time interval. Considering that at most f of n sensors have incorrect measurements, the fault-tolerant average algorithm computes $M_{nf}(I) = [low, high]$, where *low* is the lowest value in at least $n-f$ intervals in I , and *high* is the highest value in at least $n-f$ intervals in I .

Since the algorithm computes an intersection of intervals, depending on the intervals in I , the $M_{nf}(I)$ result may be more accurate than any sensor in I . However, $M_{nf}(I)$ cannot be more accurate than the more accurate sensor in I when $n = 2f + 1$.

The result of M_{nf} certainly contains the correct value when the incorrect number of sensors is not greater than f . However, the algorithm can present an unstable behavior in a way that any change in the input data can produce very different outputs.

2.4. Task Scheduling for Wireless Sensor Networks

In the WSN context, the task scheduling algorithms are responsible for allocating tasks to sensors minimizing time required to execute all submitted tasks, known as makespan, and saving energy. However, it is also important that these scheduling algorithms ensures fairness between applications; in other words, distribute resources in order to meet the needs of all running applications (LOUREIRO *et al.* 2006).

In this context, recently much emphasis has been given to the task assignment to sensors considering energy efficiency. Although a lot of effort has been made in the development of task scheduling proposals for heterogeneous and distributed computing systems ((TAPKER and

SUTER 2009); (HENAN and SAKELLARIOU 2010), (CHEN and MAHESWARAN 2008)) as well as for WSN ((BYERS AND NASSER 2000); (BYERS and NASSER 2000); (MULLEN 2006); (ROWAIHY 2010) and (ROY 2010)), these efforts are not directly applicable to the SSAN scenario, since they do not seek the existent relationships between applications, in other words, common tasks are repeated without distinction.

The works of (BYERS and NASSER 2000) (MULLEN 2006) (ROWAIHY 2010) and (ROY 2010) discuss the problems of task scheduling in WSN, seeking to minimize the power consumption and the task *makespan*. They consider WSN able to handle only one application at a time.

In the context of homogeneous WSN (in other words, networks where all the nodes have the same hardware configuration), several frameworks have been proposed for task allocation with different allocation objectives. In (BYERS AND NASSER 2000), Byers and Nasser present a framework that models the problem of allocation tasks to sensors by using utility functions and task cost. The utility function of a task is evaluated in terms of the accuracy of the collected data for the task at hand. The cost is measured in terms of the energy consumed for activating, operating and any other possible energy consumption associated with the tasks selection. The objective of this paper was to develop a scheduling algorithm that maximizes the task's usefulness respecting a predefined budget (of energy).

Another energy-aware task scheduling algorithm is proposed by Rowaihy (ROWAIHY 2010) that uses a utility function to find the task graph which shows the lowest energy consumption in order to extend the lifetime of the network. Their work also presents another solution having makespan as metric for the execution of the application. In such solution, energy efficiency is obtained through fairness in the allocation of tasks to sensors.

Still in the context of WSN, the authors in (ROY 2010) presented a proposal for a tasks scheduling algorithm used in finding the best relationship between fidelity aware resource allocation and the sensor node selection to support multiple concurrent applications. Fidelity is defined as a concept that depends on the application and can be measured in terms of a variety of requirements, including communication latency, data quality and data redundancy. In order to support the simultaneous execution of an increasing number of applications in a sensor even

while losing some of its quality, the algorithm reduces the fidelity value of various tasks belonging to the applications to achieve the desired *makespan* values (application-dependent) and to reduce memory consumption, causing the reduction of resources available for the application.

2.5. Conclusions

This chapter has presented the basic concepts necessary for understanding this thesis. We have presented concepts about wireless sensor networks and how the fit-for-purpose design is not adequate to the smart environments scenario that has in its nature multiple applications sharing the underlying infrastructure. We have also presented the Shared Sensor and Actuator Networks as a viable approach to this scenario. As the number of applications increases, it also increases the concern with the network system lifetime. Data fusion methods and scheduling algorithms are two essential mechanisms to reduce the WSN energy consumption. In this chapter we have presented traditional methods for data fusion and traditional scheduling algorithms for Wireless Sensor Networks. In the following chapters we will discuss about the new challenges about scheduling and data fusion methods involved with Shared Sensor and Actuator Networks.

3 Shared Sensor and Actuator Network Design to Support Multiple Applications Deployment: A Survey

In this Chapter we present a systematic literature review on shared sensor networks. The main objective of the performed survey is to investigate the state of art on Shared Sensor and Actuator Networks. Through this survey we will be able to: (i) better understand the Shared Sensor and Actuator Networks paradigm; (ii) identify open issues in the already existing solutions and (iii) identify new research opportunities, some of them addressed in this thesis.

This chapter is organized as follows: In the next section, we present the procedures of how we have performed our systematic literature review on SSNs. In Section 3.2, we first describe the requirements of a shared sensor and actuator network. Subsequently, in Section 3.3 we introduce a taxonomy that we derived from the survey findings, and give an overview of our understanding of how to provide supports at different levels of a SSN. The identified support necessary at each level to construct a shared sensor network will be explicitly addressed in the Sections 3.4, 3.5, and 3.6. Finally, findings related to SSAN potential applications are described in Section 3.7. Section 3.8 presents some discussions about Shared Sensor and Actuator Networks. Section 3.9 presents some conclusions.

3.1 Systematic Literature Review

This study has been carefully planned as a systematic literature review based on the methodological framework previously introduced by (KITCHENHAM *et al.* 2009), which provides a set of well-defined steps carried out in accordance with a predefined protocol. In this section, we will explain the process adopted to perform our systematic review and how the method in (KITCHENHAM *et al.* 2009) was modified to explore the research fields of wireless sensor network so as to better fit the goals and objectives of our review. In addition to the guidelines, we have also used the systematic review conducted by (LILLEGRAVEN and WOLDEN 2010) as a quick reference on how to perform the different stages of a rigorous review process.

3.1.1 Research Questions

As opposed to traditional wireless sensor network, a shared sensor network is normally considered as an environment that enables a number of WSN applications to simultaneously run on top of the same set of physical sensor nodes. Even with such fundamental difference in

designs, these two approaches still share the same sensor network architecture and protocol stack. Hence, the research issues in shared sensor network can be generalised across multiple levels, ranging from the low-level sensors control to the high-level application design.

As the first and the most critical step of all steps performed in our systematic literature review, we translate the goals of our review into the following set of research questions:

RQ1: Which are the requirements of shared sensor network designs?

RQ2: What different levels of services and supports are required for running multiple applications on the same sensor network infrastructure?

RQ3: Which existing/ongoing/potential applications can benefit from the use of a shared sensor network design?

3.1.2 Search Process

At this stage, we would perform manual search to retrieve all the literature relevant to answer the above specified research questions. For achieving this objective, we employed a two-step procedure, where the first step was to specify the sources that can provide the most recent relevant studies for our review, and the second step was to decide how to search the sources.

To determine our search sources we first aimed at examining all the online digital libraries used in the studies (KITCHENHAM *et al.* 2009) and (LILLEGRAVEN and WOLDEN 2010) and only employed the relevant libraries with significant WSNs publications. To prevent the omission of valuable publications, Google Scholar is also used as a supplemental tool to ensure the integrity of the primitive search results. Finally, before starting processing the subsequent search procedures we went through the list entirely and confirmed that each source was still active and available. The final list of sources to be searched encompasses the most well-known online digital libraries in the field of WSNs as shown in Table 1.

Table 1. The final selected sources used in the search stage of this review

Source	Type	URL
ACM Digital Library	Digital library	http://dl.acm.org/dl.cfm
IEEE Xplore	Digital library	http://ieeexplore.ieee.org/Xplore/home.jsp
ScienceDirect	Digital library	http://www.sciencedirect.com
SpringerLink	Digital library	http://link.springer.com/
ISI web of knowledge	Digital library	http://apps.webofknowledge.com
Wiley Inter Science	Digital library	http://onlinelibrary.wiley.com/advanced/search

CiteSeer	Digital library	http://citeseerx.ist.psu.edu
EI Engineering Village (includes Compendex, GEOBASE, GeoRef)	Digital library	http://www.engineeringvillage.com
Google Scholar	Search Engine	http://scholar.google.com

After completing the list of sources, we moved on to defining search terms as well as the procedure for searching papers in the online digital libraries. To create our search strings, we first selected multiple key words from our previously defined research questions and then we formed 4 groups of search terms as shown in Table 2. Each group contains search terms that are either synonyms (different forms of the same word), or terms that have similar or related semantic meaning within the field. Each group aims at retrieving different sets of research studies: Group 1 encompasses the term “shared sensor network” and its synonyms. Group 2 contains the set of term used to search for requirements related issues. Group 3 encompasses the set of terms that define types of services or runtime supports at different levels of a shared network. Group 4 contains the terms related to the potential applications of shared sensor networks.

Table 2. The search terms which we used in the online searches

	Group 1	Group 2	Group 3	Group 4
Term 1	Shared sensor network	User requirement	Layered services/supports	Application
Term 2	Multi-purpose sensor network	Design objective	Node level service/supports	Project
Term 3	Multi-owner sensor network	Quality of service	Network level service/supports	Testbed
Term 4	Multi-functional sensor network		System level service/supports	Development
Term 5	Federated sensor network			

Most online digital libraries provide advanced search-options that allow users to enter Boolean search strings. We fully exploited this feature to construct the search strings used to query each digital library. We defined one search string to search for studies related to each one of the research questions defined in this review. To search for studies related to RQ1, RQ2, and RQ3, we combined the terms of Group 1 with 2, Group 1 with 3, and Group 1 with 4, respectively. The effect of the search strings was that all studies that included at least one of the terms in first group

and at least one term in the remainder groups were retrieved. The general form of the search string for each RQ is shown below:

RQ1: (([G1, T1] *OR* [G1, T2] *OR* [G1, T3] *OR* [G1, T4] *OR* [G1, T5]) *AND* ([G2, T1] *OR* [G2, T2] *OR* [G2, T3]))

RQ2: (([G1, T1] *OR* [G1, T2] *OR* [G1, T3] *OR* [G1, T4] *OR* [G1, T5]) *AND* ([G3, T1] *OR* [G3, T2] *OR* [G3, T3] *OR* [G3, T4]))

RQ3: (([G1, T1] *OR* [G1, T2] *OR* [G1, T3] *OR* [G1, T4] *OR* [G1, T5]) *AND* ([G4, T1] *OR* [G4, T2] *OR* [G4, T3] *OR* [G4, T4]))

The search strings were performed manually within all the listed online digital libraries. At this stage, no explicit limitation was set on the published time of the literature. The papers from the returned result of each source that addressed literature surveys of WSNs were all identified as potentially relevant. To prevent valuable publications that are not included in the above online digital libraries, we performed the same search in Google scholar. If the returned papers from Google scholar were not in the previous set of results, they were examined by the paper titles and the possible key words listed in the paper. During the period of this examination, at least two researchers (researches that participated in this activity were: Wei Li, from University of Sydney, Claudio Farias, from Federal University of Rio de Janeiro and Flávia Delicato, also from Federal University of Rio de Janeiro) independently evaluated whether the article was likely valuable to our targeted research questions. If the work was considered to be relevant, it would be saved and further processed in the later stages.

3.1.3 Inclusion and Exclusion Criteria

The purpose of this step is to progressively narrow down the number of articles found in the search stage to an appropriate collection of high quality articles that is thematically relevant for answering the research questions. To complete this task, we adopted the modified criteria used in (LILLEGRAVEN and WOLDEN 2010) to eliminate the studies that are not thematically relevant to the scope of this survey. The selection criteria presented in this section involve inclusion criteria and quality screening criteria. The criteria can be further defined as a three-stage process:

3. Abstract inclusion criteria screening,
4. Full-text inclusion criteria screening,

5. Full-text quality screening.

In the following, we will describe the three-stage process in sequence and present the detailed descriptions of the results from each stage of the selection process.

- ◆ In stage one, we simply eliminated the articles that were found in the search phase based on the information provided in the abstract. Articles were only kept for further processing if the abstract satisfied the key inclusion criteria: the article's main focus is shared sensor network. For the papers with very little information in the abstract, we temporarily keep such papers in the list to be processed in the next stage. Note that, at this stage, we do not consider the quality of the papers.
- ◆ In stage two, we further eliminated the articles that failed to address the search terms (presented in table 2) in shared sensor network. Those papers that although had the strings in the abstract, it only represented a minor aspect of the paper.
- ◆ In stage three, the remaining articles underwent a quality screening where we eliminated studies that did not meet the following quality criteria:

QC1: Is there a clear statement of the aims of the research?

QC2: Is the proposed system architecture/algorithm/protocol in the work feasible (can it be applied to a real scenario)?

QC3: Are the simulations/experiments thoroughly analysed and explained, and the results of tests strongly support the ideas presented in the work?

All the articles were accessed by researchers (Wei Li, Claudio Farias and Flávia Delicato) independently by answering yes/partly/no to whether each of the established criteria was met. After the assessment was completed, all disagreements were resolved for each paper and each criterion. Finally, we calculated a sum for each paper by giving 1 point for each 'yes', 0.5 points for each 'partly' and 0 points for each 'no'. All papers that scored $P_{QC1} + P_{QC2} + P_{QC3} \geq 2.0$ points were accepted and included in the set of studies used in our review.

3.1.4 Data Collection and Analysis

The goal of the data collection process was to gather the necessary data to answer research questions in a credible way depending on the quality of data. To ensure data quality, we further set the following criteria:

1. Works are published in reliable computer science venues (peer-reviewed conference, peer-reviewed journal or computer science/engineering organisation),
2. The language for publication must be in English,
3. The works are published during the period of 2000-2013.

After the above process is completed, the extracted data is processed to draw out key themes as part of the synthesis stage of the review. Similarly to (KITCHENHAM *et al.* 2009), the data extracted from each study were:

- The authors' information, including name of authors, their institution and the country where it is situated,
- The source (journal or conference) and full reference,
- Summary of the study including the main research questions and the answers,
- Technical aspects related to the shared sensor network that was addressed in the work, including modelling, proposed solutions, quality evaluation,
- Findings and conclusions.

Within this process, two researchers (i.e. Wei Li and Claudio Farias) performed the data extracting and the rest checked the resulting data. In the event of disagreements, the data were discussed in detail until agreement was reached. All the findings will be presented and discussed in the later sections.

3.1.5 Search Results

This sub-section summarises the results of our systematic literature review of shared sensor network. There were totally 1307 articles identified as potentially relevant after searching the aforementioned databases. An additional 35 articles were identified through manual search via Google scholar, increasing the total number of articles to 1342. Before we can apply the inclusion and exclusion criteria presented in Section 3.1.3, the duplicate findings were first removed from the preliminary result set, a total of 799 candidate articles remained. Then all abstracts of the remaining articles were downloaded for further processing. The responsible researchers (Wei Li and Claudio Farias) reviewed all abstracts, applying the first stage rule of the inclusion/exclusion criteria. The 473 candidate articles were excluded during abstract review, thus decreasing the total to 326. Full text documents were all retrieved for these 326 articles. During full text article review, 239 candidate articles were excluded, leaving a total of 87 articles for the quality screening. During quality screening, 26 articles were excluded, leaving a total 61 articles for

processing in the next stage, which is known as data collection and analysis presented in Section 3.1.4. At this stage, more restricted rules were applied to the remaining articles. These led to another 4 articles being eliminated due to some conditions set for this survey only, e.g. publication sources, language and publication year mentioned in Section 3.2.4. Eventually, there were total 57 articles included in our survey studies. A flow chart of the search process is shown in Fig.1, formatted according to PRISMA statement guidelines (MOHER *et al.* 2009).

As mentioned earlier, the final number of included articles in this survey was 57 and we provide a descriptive analysis, as shown in Table 3, to summarise all the included articles before we move into answering the research questions listed in Section 3.1.1. Even we did not directly answer the research questions at this stage, these questions are still used to classify the included articles. There are 17, 34 and 14 papers are closely related to the listed research questions, respectively, and 9 out of 57 papers addressed more than one research question. As shown in Table 3, for each question, most articles are from North America followed by European ones. Most of the included articles are from conferences rather than journals, but the articles targeted for RQ1 are the exception. In this category, the number of the articles from conference and journal are almost even. More than half included articles that are focused on both theoretical and technology studies, however, a few of them targeted only one aspect, especially for articles dealing with SSN applications. Eventually, from the publication period column shown in the table, we can find out that the emphasis of enabling SSN design on WSNs has shifted from the general possibilities and requirement discussions to detailed methodical studies and application developments. This information also confirms that the trend of the SSN design has become more popular and attracted more researchers to the field of WSNs.

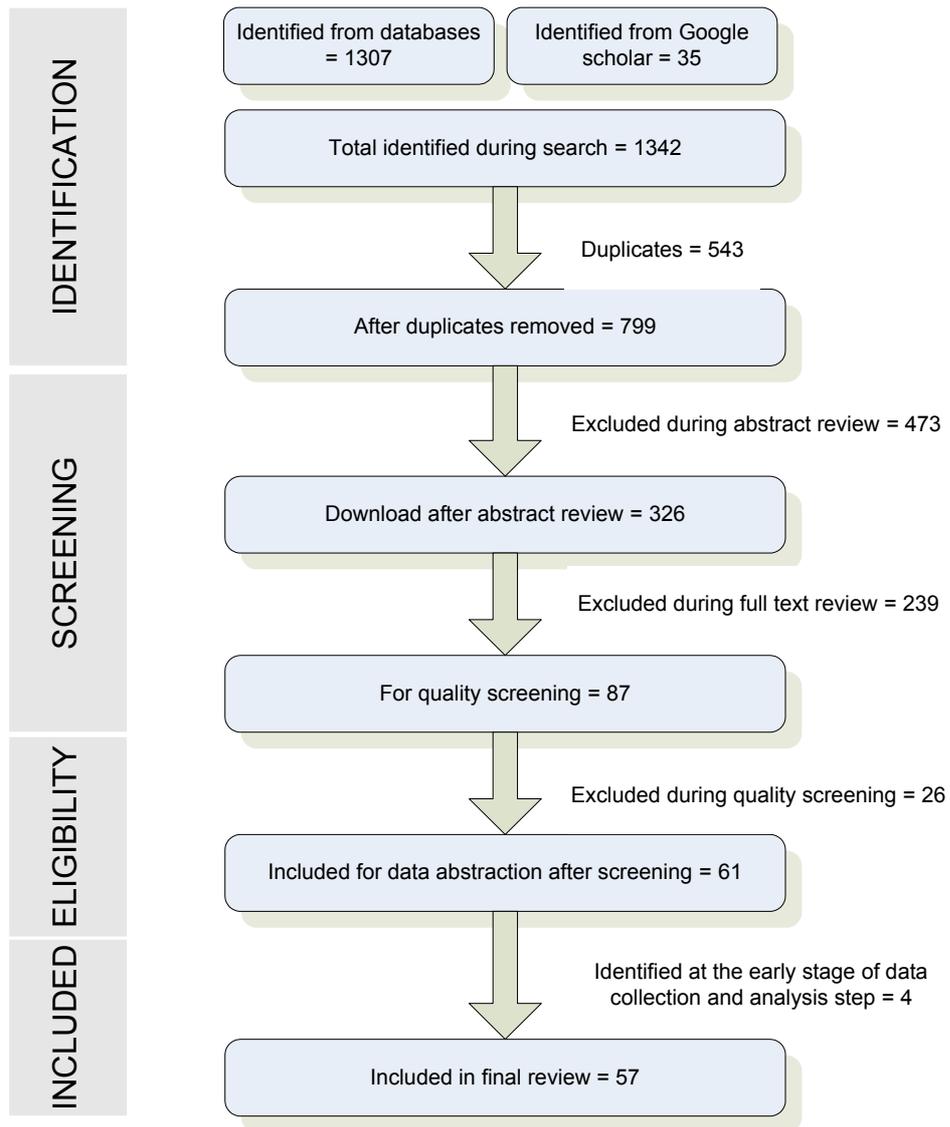


Fig.1. Process flow for literature review

In the next section, we present the results of our systematic literature review on SSNs. In Section 3.2, we first describe the requirements of a shared sensor network aiming to answer RQ1. Subsequently, in Section 3.3 we introduce a taxonomy that we derived from the survey findings, and give an overview of our understanding of how to provide supports at different levels of a SSN. The identified support necessary at each level to construct a shared sensor network will be explicitly addressed in the Sections 3.4, 3.5, and 3.6. Finally, findings related to RQ3 are described in Section 3.7.

Table 3. Descriptive analysis and summary of results for included articles

RQ	Region	Article Number	Article Type		Study Designs	Publication Periods		
			Conference	Journal		2000 - 2004	2005 - 2009	2010 - 2013
RQ1	Asia	2	0	2	theoretical study	4	7	6
	Australasia	2	1	1	theoretical study/technology trail			
	Euro	2	1	1	theoretical study/technology trail			
	North America	10	5	5	theoretical study/technology trial			
	South America	1	0	1	theoretical study			
RQ2	Asia	1	0	1	theoretical study	5	11	18
	Australasia	2	1	1	theoretical study/technology trial			
	Euro	8	6	2	theoretical study/technology trial			
	North America	18	14	4	theoretical study/technology trial			
	South America	5	4	1	theoretical study/technology trial			
RQ3	Australasia	1	1	0	theoretical study	2	5	7
	Euro	4	2	2	technology trial			
	North America	8	7	1	technology trail			
	South America	1	1	0	technology trail			

3.2 Shared Sensor Network Requirements

In this section, we first present our findings to answer RQ1 and describe the requirements needed for designing a shared sensor network. The design and development of a successful SSN is not trivial. It is necessary to deal with different kinds of challenges imposed by the nature of a WSN on one hand and the requirements of multiple applications on the other hand. In the following, we explore some important requirements necessary for constructing a SSN from the perspective of application and network requirements.

Energy efficiency: This is perhaps the most stringent requirement for almost all WSN applications, since sensor nodes are generally powered by non-rechargeable batteries, which are inevitably limited in energy capacity. Once deployed, the sensor nodes might be difficult or even impossible to be manually maintained or replaced. Due to this characteristic, WSN applications are generally required to efficiently make use of the limited energy resources to effectively reduce energy consumption of sensor nodes and thus prolong the network lifetime. The coexistence of multiple applications on SSN certainly increases the energy consumption of sensor nodes compared to the conventional case (LI *et al.* 2012). Energy efficiency becomes the most prominent performance guarantee of SSNs and must be seriously considered in every aspect of their design and operation (STEFFAN *et al.* 2005), (EFSTRATIOU 2010). SSNs should also provide a relatively simple and easy way to allow users to deploy their own applications and achieve a decent level of energy-efficiency. Meanwhile, it is preferable that SSNs can offer fine-grained power control (SUGIHARA and GUPTA 2008) to satisfy the specific requirements of different applications.

Dynamic resource allocation: The far-end users can submit diverse applications to the SSN at any time. When applications arrive in the SSN, they will be deliberately distributed to the selected sensor nodes for further processing according to several determining factors, such as the more recent sensor nodes status, user demands and the priority of the applications (BHATTACHARYA *et al.* 2010), (WU *et al.* 2012), (LI *et al.* 2013). Sensor nodes need to dynamically allocate their sensing, computation and communication resources not only to satisfy the needs of simultaneously running multiple applications without causing interruption, but also to comply with policies specified by different stakeholders. Dynamic resource allocation thus

becomes a fairly important requirement for SSNs to maximize the network lifetime whilst giving explicit consideration to efficiently utilising the resources of the nodes.

Collaboration and information sharing: This is an increasingly important requirement for many sensor network applications, especially in SSNs. The existing WSN applications can be generally divided into two types: information processing applications and data collection applications (SUGIHARA and GUPTA 2008). For the information processing applications such as location tracking applications (MOHER *et al.* 2009), (LÉDECZI *et al.* 2005) the focus is to extract valuable information by collaboratively processing raw data from a number of relevant sensor nodes. However, for the data collection applications such as habitat (MAINWARING *et al.* 2002), environmental (MARTINEZ *et al.* 2004), and structural monitoring (XU *et al.* 2004) applications, the major responsibility is to continuously/periodically transmit the collected data back to a device with sufficient computing and storage resource for further processing. Limited wireless communication bandwidth and energy of sensor nodes could be quickly consumed due to the inappropriate design that can further shorten the system's lifetime. To achieve better energy conservation in this type of applications, the in-network processing (YUAN and EKICI 2007) is often employed to reduce the amount of data transmission by way of summarizing and compressing the data within the network, which let the data collection applications exhibit some sort of collaboration between nodes. Information sharing further extends the benefits of collaboration to SSNs since some intermediate results can be directly reused among applications and the associated repeated operations can be mostly avoided. More importantly, information sharing is generally designed as a cross-layer approach (VIJAY *et al.* 2011), which not only can overcome the limitations of the layered protocol architecture, but also enable flexible or even highly configurable data processing among sensor nodes without being affected by the potential complicated underlying network infrastructure. This can easily lead to improved performance of the SSNs during their lifetime even in the presence of simultaneously running multiple applications with conflicting goals.

Dedicated platform construction: SSN design is fully compatible with the traditional application-specific WSN by constructing virtual sensor network (ISLAM *et al.* 2012) over the same physical network as a dedicated platform for each application. Each virtual sensor network comprises a subset of the resources of the underlying physical network while it is logically isolated from others. To accomplish such design is not trivial, especially when the underlying

Before going into the details of the selected literatures addressing RQ2, let us first examine the challenges associated with a shared sensor network design. In this sub-section, we classify them into two groups: low-level network design and high-level network design, where low-level network design is concentrated on abstracting hardware and allowing flexible control of sensor nodes, and high-level network design is focused on enabling collaboration among sensor nodes to complete the assigned tasks from WSN applications. High-level network design can be further refined into two types: network-level support and system-level support. Network-level support aims at providing the supports for handling volatile and dynamic resource requirements from different applications, as well as sharing common information between applications for achieving better energy conservation. System-level support builds on the top of the network-level support and goes further by treating multiple networks as a single abstraction.

Figure 2 depicts the taxonomy for developing shared sensor network derived by systematically analyzing the designed architecture and the implementation approaches of the literature found in the previous search procedure. Node-level supports are discussed in Section 3.4. High-level network design, specifically network-level support and system-level support are discussed in Sections 3.5 and 3.6, respectively. All these three sections intend to provide the detailed answers to RQ2 listed in Section 3.1.1.

3.4 Node Level Support

The technical problems and research issues of constructing shared sensor network raised at the node level are mainly related to two aspects: (i) how to handle node heterogeneity and (ii) how to create flexible execution environments for multiple applications execution and their requirements inside a single sensor node. In the following subsections we discuss the different approaches commonly adopted to deal with such aspects.

3.4.1 Virtual Machine Approaches

In a shared sensor network, heterogeneous nodes are deployed in the target area and applications employ those nodes to perform their desired services. The major challenge existing in this level is to appropriately handle the heterogeneity in sensor nodes so as to allow application developers efficiently controlling the underlying infrastructure without knowing the low-level hardware details. For addressing this challenge, in-node virtualization has been suggested as a possible approach to abstract the sensor node hardware and to expose the nodes' functionalities in the

form of services, relay sensed data to upper layers, and provide common interfaces to application programmers/developers.

The current implementation of in-node virtualization mainly relies on interpreter-based virtual machines (SUGIHARA and GUPTA 2008) (MOTTOLA and PICCO 2011), which aim at providing virtualization for heterogeneous sensor nodes relying on the lightweight byte code interpreters that reside inside each sensor node and offering a fine-grained control over the execution of applications. A well-known interpreter-based virtual machine is Maté (LEVIS and CULLER 2002) implemented on the top of TinyOS (HILL *et al.* 2000), which is a nesC (GAY *et al.* 2009) re-implemented event-driven operating system specifically designed for sensor nodes with very limited resources. However, Maté is originally designed as an application-specific virtual machine due to the fact that, at the early stage, the general observation is that WSNs were mostly designed for running a specific application. For addressing this issue, Yu *et al.* presented a significant extension of Maté, called Melete (YU *et al.* 2006), for enabling reliable storage, creating and maintaining a dedicated execution space for each application, and separately compiling the code for each application to avoid variable sharing across applications so as to construct multiple application-specific environments within a single sensor node for eventually support concurrent applications execution. In heterogonous WSNs, Java is an attractive alternative for implementing virtual machines on sensor nodes since its distinctive characteristic—platform independency compared to other language solutions, such as C/C++. Although Squawk (SIMON *et al.* 2006), a Sun Microsystems developed open source virtual machine, reduces the node resource requirements, Darjeeling (BROUWERS *et al.* 2009) is perhaps the first well-known feasible open-source Java compatible virtual machine for most sensor nodes used in WSNs with limited memory resource (less than 10 KB of RAM). It provides a step-by-step compilation process which firstly translates source code to bytecode and then to an intermediate bytecode and eventually is interpreted by the virtual machine. Mote Runner (CARACAS *et al.* 2009) is another Java compatible virtual machine for WSNs similar to Darjeeling. These two Java compatible virtual machines both implement a 16-bit instruction set, but the major difference between these two solutions is that Darjeeling used dynamic sized thread stacks, whereas Mote runner used event callbacks to support multiple applications execution inside sensor nodes. It is worth noting that in (SHARMA *et al.* 2009), the authors claimed they successfully implement a finite state machine to represent all the possible settings of each

actuator within sensor nodes, called Vsense, in the extended Xen (BARHAM *et al.* 2003) virtual machine monitors, a widely used high performance resource-managed virtualization platform, for providing fine-grained concurrent access to sensors with programmable actuators. Compared to Melete or other in-node virtualization approaches, Vsense is primarily designed for high-power sensors (sensors with large batteries and more accurate sensors), which are typically devices that transfer data streams to applications, e.g. pan-tilt-zoom camera.

3.4.2 Middleware Approaches

Due to the increased need to employ multiple sensors for satisfying applications' requirements, the idea of middleware has recently been adopted for providing node level abstractions to hide the heterogeneity of sensor nodes without concerning differences in node architectures, operating system and programming languages. A SSN middleware can also be designed to be seamlessly interoperable with various devices such as devices fabricated with different kinds of sensors, RFID (Radio Frequency Identification) and actuators in order to enable rapid application development and reliable operation. Impala (LIU and MARTONOSI 2003), as the middleware layer of ZebraNet Project (JUANG *et al.* 2002), is the early inspiring work in the field, which was implemented on resource-constrained ZebraNet hardware nodes to efficiently handle scheduled operations as well as to ensure reliability and ease of upgrades for long-running applications. One important feature of Impala is to provide dynamic programmability and rapid updates to the operation of sensor nodes, that is, the capability of allowing the running nodes to be updated on-the-fly without service disruption. This feature is important in the context of SSN since it enables loading code for different incoming applications (meaning applications that are not known at the design time, but instead are to be executed after the network deployment, in a dynamic way). Several following works by other authors, e.g. TinyLime (CURINO *et al.* 2005), actorNet (KWON *et al.* 2006) and Agilla (FOK *et al.* 2009), propose more innovative and complex schemes, which made possible the deployment of interpretable codes that could freely move from one node to another within a multi-hop WSN. In addition, the idea of service oriented middleware (PAPAZOGLU *et al.* 2007), (DELICATO *et al.* 2003) has been gradually extended to the SSN, where service oriented middleware views the heterogeneous sensor nodes as a distributed system and provides a framework representing business processes as services with clear and accessible interfaces to far-end users. TinySOA (AVILÉS-LÓPEZ and GARCÍA-MACÍAS 2009) is an example of such kind of approach that directly deploys the lightweight

code units on top of the nodes' operating system for allowing programmers to access nodes from their applications by using a simple service-oriented API via the language of their choice. It develops mechanisms for WSN infrastructure registry, node discovery and associates with a gateway component that acts as a bridge between a WSN and external Internet applications. With these components, web services can be easily used to allow applications to access WSNs. The similar functionalities are also found in the work (DELICATO *et al.* 2010).

3.5 Network Level Support

The technical problems and research issues of constructing shared sensor network raised at the network level are mainly related to three aspects: firstly, how the sensor nodes within the same network can efficiently handle volatile and dynamic resource requirements from different applications; secondly, how to share common information between applications for achieving better energy conservation, and finally, how to successfully transmit the data to the desired destination with minimal cost in between multiple logical isolated networks.

3.5.1 Resource Management

In order to solve the first issue, the concept of virtual sensor node was proposed in (KABADAYI *et al.* 2006) for providing indirect measurements of abstract conditions by combining sensed data from a group of heterogeneous physical sensor nodes. In this work, a virtual sensor node is not a real physical sensor; instead it appears as a programming abstraction to supply the desired sensed data to different applications. This approach provides a higher level of sensor node abstraction to the application developers, in comparison to programming directly the hardware. More importantly, the declarative specification of a virtual sensor node allows developers to directly invoke the specified functions without explicitly handling the underlying details of how the sensing task is be programmed in the nodes, besides hiding the underlying hardware details from the applications.

Inspired by the work (KABADAYI *et al.* 2006), follow-up proposals extended the ideas of virtual sensor node to virtual sensor network (VSN) for better supporting concurrent applications execution on the same physical network. The concept of VSN is derived from network virtualization (CHOWDHURY and BOUTABA 2009), which mainly focuses on how to enable a set of virtual networks to jointly share the resources of the same underlying physical network, as well as to decouple infrastructure from services as it is done in the traditional ISPs. The primary

goal of the VSN is therefore to construct a number of application-specific sensor networks simultaneously running over the same physical network without causing interruption, where each application-specific sensor network is formed by a subset of nodes that might not have direct communication with each other. In addition, the membership of a VSN can change over time according to the overall network status and incoming applications. In (JAYASUMANA *et al.* 2007), the benefits of using VSN were further discussed and two real-world applications, (i) geographically overlapped sensing applications and (ii) underground contaminant plume tracking, were chosen as examples to support the opinions expressed in their work. Instead of giving a practical implementation, the authors identified the main issues involved in the implementation of VSN and elaborated the possible functions to be provided in their proposed design. They classified the major functions of VSN into two closely related categories: VSN maintenance and membership maintenance. Unlike traditional sensor networks, the membership in VSN is generally dynamic so that the efficient membership management in the VSN directly affects the energy conservation, as well as the network lifetime. The basic node membership maintenance functions include joining VSN and leaving VSN, where these functions are represented as decisions made by nodes themselves. After joining a VSN, the physical sensor node is identified as a current member and will require process the tasks from different applications. VSN maintenance involves not only the existing members of the VSN, but also the non-existing members of the VSN. The non-existing members serve as the supporting nodes for successfully implementing the VSN maintenance functions including broadcasting within VSN, fusion two VSNs, splitting VSNs and deriving contour of boundaries. The functions are required to be implemented with minimal overhead, while taking other limiting factors in wireless sensor network into account.

Yang *et al.* proposed a similar implementation as described in (YU *et al.* 2006) based on the above-mentioned principles. In their work, the selected sensor nodes formed one logical group, called associated group, which is dedicated to a single application. When the WSN is deployed, all the sensor nodes are set to be member of a default associated group during their lifetime and they can be assigned membership of multiple groups. Thus, each sensor node can dynamically join or leave associated groups according to the needs of incoming applications and different user requirements. Once the VSN is generated, the tasks of the application will be selectively distributed to the associated group members or reactively distributed only when they are required.

FRESNEL (EFSTRATIOU 2010) is a recently launched shared sensor network project focused on building a large scale federated sensor network with different applications sharing the resources from the same underlying hardware infrastructure. SenShare (LEONTIADIS *et al.* 2012) as a part of the FRESNEL project, also attempts to address the technical challenges arise from the network level by constructing overlay sensor networks which are not only responsible for providing the most suitable members to perform the tasks from applications, but also isolating the network traffic of a target application from the network traffic generated by other applications or the supportive mechanisms used to maintain the network overlay. For achieving the goal of traffic isolation, SenShare extends each application packet at the runtime with a 6 bytes long application routing header, but the entire network message is still formatted under the IEEE 802.15.4 standard. Since the sensor nodes of a VSN can be located anywhere within the network, the nodes with allocated tasks and physical neighbours that can communicate with single hop messages are then formed a cluster. This generally results in a number of clusters that are isolated from each other. For constructing a VSN from these clusters as a single connected application-specific network, virtual links between the clusters need to be established with the help of nodes that are not performing tasks from the target application. Virtual links between clusters are incrementally generated by three consecutive steps, where 1) identify the nodes that are on the edges of a connected node cluster, 2) discover optimum paths from the nodes selected in the previous step that connect the local cluster to other clusters, and 3) ensure all the clusters are connected together and can access the network's sink. There are some other related projects, such as WebDust, VSNs, which also propose solutions to address the same issues at the network level. Interested readers are referred to a recent VSN survey (ISLAM *et al.* 2012) for further details.

3.5.2 Information Sharing

The next critical issue relevant to a shared sensor network approach at the network level is to achieve better energy efficiency by sharing information between applications. Energy efficiency is perhaps the most common and most severe requirement for every WSN. A wide range of energy conservation mechanisms and techniques have been successfully developed and used in WSNs for prolonging network lifetime. However, almost all the existing solutions are targeted on the application-specific WSN and they do not provide extra benefits on energy conservation when applied to shared sensor network. In order to further prolong the network lifetime of shared sensor networks, a new energy conservation technology is actually required for providing a

decent level of energy efficiency without spending too much effort and exploiting intrinsic features of a shared infrastructure.

Task sharing is a kind of information sharing mechanism. It is a specially designed energy conservation technique tailored to shared sensor network based on a key observation that maximising the sharing of tasks among multiple applications. It can greatly assist the accommodation of concurrent applications in the same network so as to reduce the energy usage across the applications with common tasks. A multi-application task sharing approach was proposed in (NIRMALYA *et al.* 2010) as a prototype implementation. The basic idea of the solution is to utilize a number of selected tasks from the same application to construct an affiliation set. Multiple WSN applications will then generate multiple affiliation sets and the tasks in these affiliation sets could simultaneously request the same resource from the same sensor. Instead of offering the limited hardware resources multiple times, the approach performs a single task from such tasks and then returns the result to all of them. The work of (FARIAS *et al.* 2013) proposed another solution to address the task sharing issue in SSN by avoiding repeatedly executing the common tasks belonging to multiple WSN applications. For example, if two targeted applications have a common task, this task will only be performed once and the generated result will be shared by both applications. In this solution, the task allocation is mainly relied on the remaining energy of sensor nodes. For further prolonging the system lifetime, the common task set with more intensive sharing effect among applications is assigned higher execution priority. Li *et al.* proposed a practical task sharing solution in (LI *et al.* 2012) by taking several associated factors, e.g. application arrival time, data accuracy data freshness, into consideration. This task sharing solution is inspired by the idea of session persistence, which is a widely used job dispatching strategy to ensure that the same user is connected to the same server for the duration of the session. It can help maximising the intersection of execution time of the tasks with the same resource requirement from different applications while ensuring the minimum data accuracy for all these tasks. At the meantime, the data freshness setting prevents the expired data becomes a risk factor to affect the final result.

Finally, we noted that information fusion can be extended to SSNs for achieving the objective of energy saving with adequate modification. When a specific sensor node receives useful information from other nodes, the information should be effectively fused with the available local information to reduce the amount of data transmission and thus minimising the possible energy

consumption on wireless communication modules of sensor nodes, which have been identified as the major energy consumer by a number of studies (XIONG and SVENSSON 2002), (NAKAMURA *et al.* 2007), (LI *et al.* 2013). In general, the information fusion approaches range from simple rules to model-based techniques with different objectives on performance and robustness. Simple fusion rules are superior at robustness but suboptimal, while more sophisticated and higher performance fusion rules might be sensitive to the underlying infrastructure. More importantly, almost all the proposed information fusion algorithms are designed for application-specific WSNs without considering the possible information fusion between different applications. Until fairly recently, the first work presented by (FARIAS *et al.* 2012) began to notice this key issue and proposed a first simple rule based information fusion algorithm EMAF (Enhanced Moving Average Filter) into SSNs. EMAF assigns different weights to the applications running on the SSNs and provides user-configurable parameters to indicate the importance of data to the specific application. However, limited by the characteristics of simple rules based information fusion algorithm, more sophisticated model-based information algorithms are expected to be migrated into SSNs and further improve the accuracy of information fusion.

3.5.3 Routing

The last but not least critical issue for constructing SSNs at the network level is routing. The conventional routing protocols developed for application-specific WSNs usually attempt to achieve routing efficiency by exploiting the application layer query semantics (ROCHA *et al.* 2012) and proposing an all-in-one solution that weaves the routing concern with other application layer concerns, such as data-centric and service-centric routing. They also tend to optimise routing performance for a specific communication pattern inspired by a specific class of WSN applications. In a SSN, where multiple applications run within the same network infrastructure, each application presents its own set of requirements that must be dealt with and exploited by routing protocols in order to guarantee energy efficiency while forwarding data.

One of the possible approaches that tackles routing issue in SSNs is described in (ELTARRAS and ELTOWEISSY 2010). The authors proposed associative routing as a class of routing protocols that enables dynamic semantically-rich descriptive identification of network resources and services. As such, associative routing presents a clear departure from most current network addressing schemes, eliminating the need for a separate phase of resource/service discovery.

Since in essence, resource discovery operates similarly to path discovery, then both can be performed in a single phase, leading to significant reduction in traffic load and communication latency without any loss of generality. They also propose a framework for associative routing and present adaptive multi-criteria routing (AMCR) protocol as a realization of associative routing for sensor networks. AMCR exploits application-specific message semantics, represented as generic criteria, and adapts its operation according to observed traffic patterns. The routing is based on the needs of each application. It uses packet loss, delay and other QoS metrics to formulate rules to be evaluated for these applications. Based on these rules, AMCR form common paths to several applications.

The work of (SHAH and SZYMANSKI 2012) presents a dynamic multipath routing protocol in which packets from different applications dynamically choose their paths towards the sink node or other nodes by taking into account the price to be paid for taking each path and their ability to pay. It is proposed a mechanism in which the prices reflect congestion on routers (nodes in the path) and thus the waiting time for passing the router by a packet and ability to pay is defined by the application priority and the packet waiting time. These prices increase as usage of shorter routes increases. As a result, low priority applications tend to avoid paths with high prices. Instead, they go via low price routes which may be longer but faster to pass by avoid waiting for passage at congested routers. This enables high priority traffic to get through quickly via short paths as their priority (often representing their social value) enables them to pay high prices with little wait. Thus, the proposed approach segregates traffic flows of different applications and lowers congestion and delays for all applications. The authors further show that the dynamic path allocation technique proposed performs well both in normal and emergency situations in which network is partially damaged.

3.6 System Level Support

At the system level, one or more hierarchical WSNs belonging to a single authority are treated as a whole and are regarded as a multi-service system. The primary concern at this level is how to collaboratively complete multiple WSN applications by properly allocating their component tasks to a set of sensor nodes potentially located at different hierarchical WSNs. For solving this issue, the major approach is task allocation or task scheduling (ZOMAYA 1996), which is a classical technique usually used in parallel and distributed computing systems. The most important difference between task allocation and task scheduling is that task scheduling addresses the task

dependencies and their execution sequences which task allocation does not address. In addition, one major difference between task scheduling in WSN and traditional scheduling algorithms is that the latter ones focus on shortening the makespan (the time of a generated scheduler takes to finish a batch of tasks). Instead, in WSNs the major concern is not only time, but also energy, since each sensor node has limited power supply and the system is expected to run as longer as possible after deployment. Considering this vital requirement, the task scheduling problem becomes a multi-objective scheduling problem of which the cost function includes energy consumption, time and other possible influencing factors. The existing approaches can be classified, according to the network topology, into two categories: task allocation/scheduling on a single WSN and task allocation/scheduling on multiple WSNs.

Within the task allocation/scheduling on single WSN category, existing studies can be further classified into two sub-categories: single-hop non-hierarchical WSN task allocation/scheduling and multi-hop non-hierarchical WSN task allocation/scheduling. For single-hop non-hierarchical WSNs, the task allocation/scheduling problem has been addressed and well-studied. Yu and Prasanna (YU and PRASANNA 2005) developed an energy-balanced task allocation (EBTA) algorithm to meet the deadline of a real-time application running on homogeneous sensor nodes connected via multiple wireless channels. They formulated the problem as an integer linear programming issue and presented a polynomial 3-phase heuristic solution. However, they did not consider the broadcasting nature of wireless communication in their model and the multiple wireless channel technique is not widely used in the real world sensor nodes. EcoMaps (YUAN *et al.* 2005) algorithm is proposed for the energy-constraint applications with no deadline requirements. It aims to map and schedule the tasks jointly to achieve the minimum schedule length which meets the requirement of minimizing energy consumption. Yuan *et al.* presented RT-Maps in (YUAN *et al.* 2006) which can guarantee the real-time application deadline with minimum energy consumption. Their algorithm also considers utilizing the broadcast nature of wireless communication to conserve energy. Xie and Qin (XIE and QIN 2008) presented a task scheduling algorithm called BEATA (Balanced Energy Aware Task Allocation) to solve the energy-delay dilemma that exists in heterogeneous WSNs. BEATA aims at minimizing the energy consumption while confining schedule lengths through task allocation.

The aforementioned techniques concentrated on information processing in a single hop range, but in real world applications sensors are normally randomly deployed in an area of interest and

form an irregular topology. Therefore, since the single-hop non-hierarchical WSN scheduling algorithms cannot be directly applied to real world scenarios, proposals to tackle the issue of scheduling in multi-hop non-hierarchical WSNs have been attracting researchers' attention recently. In (YUAN and EKICI 2007), the authors proposed a multi-hop in-network processing algorithm called MTMS (Multihop Task Mapping and Scheduling). In their algorithm, the nature of multi-hop communication is handled by two steps. First, they extended the representation of tasks from DAG to a Hyper-DAG by replacing a communication edge as a vertex and connecting this new vertex to the vertexes which it originally starts from and ends to. The cost of the communication vertex is equal to the communication load in the DAG. Second, all the sensors are assumed to connect to a virtual communication controller called C. In a specific time slot, the algorithm running on the controller C determines whether processing a communication task in the system will cause interference with other communication tasks which are simultaneously being performed or not. If so, the algorithm will seek another time slot to process, otherwise, it allocates this time slot for the communication task to process. However, their communication scheduling algorithm only considers one-hop communication collision and some kinds of hidden communication collision may occur. Furthermore, their model does not suitably address the communication concurrency; in most cases, the wireless communications in the selected sensors are performed sequentially.

Since all the previous works assume that the application is only performed inside a single WSN, they are not able to explore the potential cooperation among multiple WSNs in a shared sensor network design approach. In order to better utilize the resource of multiple WSNs and extend their lifetime, the researchers proposed several task allocation/scheduling solutions tailed to the SSN scenarios. In (BHATTACHARYA *et al.* 2010), the authors presented Utility-based Multi-application Allocation and Deployment Environment (UMADE), which is a task allocation system for distributing various applications based on QoM (quality of monitoring). QoM is a distributed quality metric for monitoring a physical phenomenon of interest that is based on the accuracy of measurements. Thus, the quality of QoM depends on the monitoring performed by all nodes allocated to an application. Unlike traditional approaches that usually allocate nodes in the networks to a single application according to metrics such as the amount of resources used, delay, processing and power consumption, UMADE dynamically allocates nodes to multiple applications according to the application's QoM. An inherent property of an application is that

sensing data belonging to sensors that are allocated to a same application are naturally correlated. As a consequence, the contribution of a sensor node for an application QoM is dependent of other sensor nodes allocated for the same application.

In (XU *et al.* 2010), the authors propose a greedy algorithm to schedule applications onto a specific SSN. The algorithm performs the task allocation taking into account the QoM of the applications. The applications' QoM depends on the node to which the application was allocated. Therefore, it is important that the allocation algorithm seeks to optimize the allocation among multiple applications of a SSN to maximize the QoM. The proposed work uses the property of QoM sub-modularity. The QoM sub-modularity is due to the fact that the readings from the sensors of different nodes are often correlated. For example, once the temperature readings from different nodes in the same room are correlated with each other, the assignment of a new node in the room to perform the monitoring temperature does not produce a considerable QoM improvement. The work of (WU *et al.* 2012) is an extension of (XU *et al.* 2010) and presents a distributed game-theoretic approach to application allocation in shared sensor networks. The authors transform the optimal application allocation problem to a sub-modular game and then develop a decentralized algorithm that only employs localized interactions among neighbouring nodes. The authors prove that the network can converge to a pure strategy Nash equilibrium with an approximation bound of $1/2$. The authors validated their results through simulations based on three real-world datasets (Intel dataset, DARPA dataset and BWSN dataset) to demonstrate that their algorithm is competitive against a state-of-the-art centralized algorithm in terms of QoM. However, all of these approaches are task allocation solutions without addressing the latency issue which often occurs in practical WSN applications. For further addressing this issue, Li *et al.* (LI *et al.* 2013) proposed a heuristic called TPTS (Three Phase Task Scheduling) for enabling the task scheduling over multiple WSNs. TPTS is designed to find out a scheduling scheme that minimizes the overall energy consumption and balances the workload of the system while meeting the application deadline. However, TPTS considers that all sensor nodes contain the same sensing devices, which means each WSN within the system is homogeneous in terms of its capability. In the shared sensor network environment, the sensor nodes within WSNs may contain many different types of sensing devices such as seismic, low sampling rate magnetic, thermal, visual, infrared, acoustic and radar, which are used to detect the different events. Moreover, TPTS considers that each application is collaboratively processed by all WSNs within the system.

In their recent work (LI *et al.* 2012), a more general solution to the problem of task scheduling on multiple WSNs, called HTPTS (Heuristic Three-phase Task Scheduling), was proposed for addressing the constraints observed in TPTS. HTPTS is intentionally designed as a multi-objective scheduling algorithm with user tuneable parameters, e.g. energy consumption, load balance and data accuracy so as to elastically extend the system lifetime, while successfully completing the time-constrained applications before their deadlines.

3.7 Applications of Shared Sensor Networks

The early stage of WSNs development was motivated by the needs of military applications (AKYILDIZ *et al.* 2002). However, nowadays these networks have been extensively applied to various civilian applications, many of them being easily found in our daily lives, including electric power transmission and distribution, health and medical applications, smart environments, environmental monitoring, security, structural monitoring, agricultural monitoring and many more. This section presents a brief review of selected applications of shared sensor network to reveal the benefits and potentialities offered by such kind of design, thus providing answers for RQ3. The material is organized in four sub-sections devoted to summarizing applications focused on: (1) Smart Buildings, (2) Smart Grids, (3) food transportation and (4) healthcare and medical applications.

3.7.1 Smart Buildings

A smart building (MEYER and RAKOTONIRAINY 2003), (SNOONIAN 2003) is an environment where a variety of sensors, actuators and computational units work continuously and collaboratively to allow the occupants to customize the functionality of their living environment, such as homes and offices, industrial plants and leisure environment. Smart building is a common application of SSN, which is a dwelling that contains highly advanced automatic systems for constantly monitor and intelligently control the building conditions, and providing appropriate services to the people staying in it according to their needs. Sensors and actuators deployed in the building can make the internal environmental conditions more comfortable and safer in several aspects. For example, the room temperature can be adapted to the owner's preferences and to the weather; the room lighting can be changed along with the time of the day; the domestic incidents can be avoided with appropriate monitoring and alarm system; and energy can be saved by automatically switching off the appliances when not needed.

Several works have realized the advantages of shared sensor network and then used such approach to implement their smart building projects. Actually, smart buildings are one of the first real world scenarios to motivate the design of shared networks, since they consist in an environment instrumented with sensors which are generally of the same owner, and should ideally run various applications relevant to the building operation. In (BHATTACHARYA *et al.* 2010), the authors implemented their design as an integrated environment, called UMADE. UMADE can simultaneously support five representative applications in the context of smart building, including temperature monitoring, humidity monitoring, air quality monitoring, light monitoring for lighting control and acoustic signal monitoring for noise control. Each application periodically samples and transmits the processed sensing data to the sink node without causing any interruption, and then the sink node will send corresponding commands to the actuators to adjust the local condition. The aforementioned platform SenShare (LEONTIADIS *et al.* 2012) has also been successfully deployed by the authors in the office space of their research institution as a prototype demonstration of a smart office project for supporting two long-term primary applications and several short-term experimental applications. Two long-term primary applications are room environmental conditions and office occupancy, where the first one is responsible for recording temperature and humidity level for all rooms, and the second one is responsible for monitoring employees' collective working time by recording how much time they spent at their desk. The short-term experimental applications are mainly focused on controlling the working mode of appliances, such as coffee machines, without disrupting pre-existing applications. Several other smart home projects show the similar research trends by exploring the use of shared sensor network to monitor the well-being of individuals in the home environment, including: Aware Home (KIENZT *et al.* 2008) from the Georgia Institute of Technology, PlaceLab (INTILLE *et al.* 2005) from the Massachusetts Institute of Technology, and Gator-Tech Smart House (HELAL and CHEN 2009) from the University of Florida.

3.7.2 Smart Grids

The electricity grid is a collection of power plants and transmission and distribution facilities that are responsible for energy generation, power transmission and electricity distribution processes and for delivering reliable electricity service to customers in real-time (EROL-KANTARCI and MOUTFTAH 2011). The techniques used in energy generation, power transmission and electricity distribution are mostly mature and stable, whereas the upward trend in population

along with the increasing consumer electronics market have substantially increased the energy dependence world widely. A Smart Grid is a modernized electrical grid that utilises information and communication technology to gather and act on relevant information, such as information about the behaviours of suppliers and consumers, in an automated manner in order to improve the efficiency, reliability, economics, and sustainability of the production and distribution of electricity (EPRI 2008). SSNs have a broad range of applications in the Smart Grid context and we will briefly discuss such applications in the following.

In the traditional power grid, energy generation facilities are generally monitored with wired sensors which are limited in amount and located only at a few critical places (GÜNGÖR *et al.* 2010). One of the objectives of the Smart Grid is to strengthen the role of renewable energy sources in the energy generation cycle. These renewable energy generation facilities could be located in sparsely populated areas, and operate in harsh environments where continuous monitoring with low cost sensors becomes necessary. SSNs offer an ideal solution for monitoring and control of the generation facilities in the Smart Grid, since multiple monitoring applications need to exchange data and work collaboratively to ensure the entire system is in good condition (FARIAS *et al.* 2012). In the transmission and distribution segment of the power grid (ISHMANOV *et al.* 2011), the components that need to be continuously monitored in near real-time are the substations (GÜNGÖR *et al.* 2010), overhead power lines (FARIAS *et al.* 2012) and underground power lines (GALLI *et al.* 2011). A single equipment failure or breakdown has chance to cause blackouts, and it may even become a health and security threat to the public society.

3.7.3 Food Transportation

The freshness of perishable foods such as fruits of the season, fresh-cut vegetables, butcher's meats, and dairy products can drastically affect the quality of our lives. From the place of production to the consumption sites, the foods could be transported as far as thousands of kilometres away and potentially exposed to the changing environment. During the transportation the conservation status needs to be carefully monitored to prevent food spoilage.

For achieving such goal, a demonstration example was presented in (STEFFAN *et al.* 2005) by using shared sensor network design. In such work, the perishable foods are placed into containers and a number of sensors are deployed in these containers for different purposes, e.g. monitoring environmental conditions, detecting tampering or leakage of dangerous goods, and

providing an RFID-based (Radio Frequency Identification) real-time inventory. The authors considered all the sensor nodes to form a shared sensor network that can support multiple applications simultaneously. The sensor nodes within the same container are treated as a logical sensor network mainly performing the environmental condition monitoring with different sampling rates, thresholds and other settings for different foods. Some nodes within a container capable of communicating with the nodes located at other containers are thus treated as new logical sensor networks, called inter-container networks. The inter-container networks can be used for different purposes during the journey, such as detecting the containers that went overboard or were lost in the road or determining the position of a specific container. In addition, all the food containers are possibly belonging to different authorities, the collected information such as inventory or condition of foods is business-critical and should be confidential and only be available to the responsible personnel.

3.7.4 Health and Medical Applications

As the world population is aging, human healthcare becomes an increasingly prevalent issue worldwide. WSNs techniques have long been applied to the healthcare domain as efficient and reliable medical aids. In medical healthcare systems, sensor nodes can be placed on, near or within human body to provide continuous and uninterrupted human vital signs, such as heart rate, blood pressure, body temperature, and respiration. For this reason, this kind of WSN is also known as wireless body sensor network (WBSN). In addition to data collection, WBSNs are also used for people tracking, where tracking is the function aimed at the identification of a person in motion. This includes both real-time positions tracking, such as the case of patient-flow monitoring to improve workflow in hospitals and tracking of motion through choke points, such as access to designated areas.

Most WBSNs are special purpose designs with fixed applications, mainly for recording sensor traces. However, researchers have recently realized the profits of applying shared sensor network to WBSNs. In (BUI *et al.* 2012), the authors proposed a prototype body sensor platform that supports dynamical applications deployment, their concurrent running, data sharing between multiple applications and handle different sensors and their configurations. Two applications, namely posture detection and activity monitoring are simultaneously running on the demonstrated WBSNs platform, where the posture detection application is used to detect the real-time events, such as standing, laying and sitting; while the activity monitoring application is used to detect an

activity event when the user is standing and doing an exercise with his arm. For each application, the corresponding sensors collect the relevant data to be kept separated in different data tables and the sensed data are then processed by two different algorithms. In order to reduce the algorithm complexity and accurately determine the user's behaviour, the activity event application also subscribes to the posture events produced by the posture detection application. With the data from both applications, the activity monitoring application can finally determine the user behaviour in an efficient and accurate manner.

3.8 Discussion

In this section, we first analysed how each of the requirements listed in Section 3.1 is addressed in SSNs and their applications. Please note that any selected requirement is not completely independent from the remainder requirements in the list, and thus any of them can work in favour of or against other requirements. Instead, there is a clear dependency among requirements: sometimes a relationship of synergy while sometimes of opposition. For instance, reducing the information exchanging frequency of sensor nodes can definitely improve energy efficiency, but it may jeopardize the collaboration among them.

As we mentioned before, energy efficiency is perhaps the most important requirement for SSNs, which has been intensively addressed in different approaches. To meet this requirement at the node-level, the efficient code update is the main method used in the virtual machine based approach, where code update refers to the capability of injecting new codes into sensor node on site dynamically. Maté (LEVIS and CULLER 2002) and Melete (YU *et al.* 2006) use compact intermediate code to reduce the communication overhead so as to achieving the energy efficiency requirement. Another noticeable approach is to model the sensor nodes as a multi-modules device and each module has its own codes. With the needs of different applications, only the corresponding module codes are upgraded and the wireless communication broadcasting nature can be well utilised to further reduce the communication overhead. This is normally used in middleware approaches such as TinySOA (AVILÉS-LÓPEZ and GARCÍA-MACÍAS 2009), Agilla (FOK *et al.* 2009) and Impala (LIU and MARTONOSI 2003). However, energy efficiency mechanisms employed in the low-level network design barely take flexible QoS control into account, which means users can not precisely control the trade-off between energy efficiency and other concerned QoS metrics. This issue quickly raised considerable attention and it has been gradually resolved in the high-level network design, especially in the task allocation or task

scheduling approaches. The common way to handle this issue is to model the user requirements as a multi-objective optimisation problem. For instance, BEATA (Balanced Energy-Aware Task Allocation) (YU and PRASANNA 2005), (XIE and QIN 2008) is a bi-objective problem, MTMS (Multihop Task Mapping and Scheduling) (YUAN and EKICI 2007) is a tri-objective problem, and TPTS (Three Phase Task Scheduling) (LI *et al.* 2013) and HTPTS (Heuristic Three-Phase Task Scheduling) (LI *et al.* 2012) are quad-objective problems.

The primary approach to achieving dynamic resource allocation is again task allocation and task scheduling. In general, the proposed approaches employ adaptive mechanism for handling dynamic application arrivals according to the previous and current behaviour of the system. Most solutions work in a centralized manner, where a single central point called global scheduler is responsible for collecting information about the available sensor nodes and processing incoming applications. Each application will be decomposed into a number of dependent/independent tasks at the global scheduler and distributed to the selected nodes directly. Centralized scheduling is a very efficient mechanism in an ideal environment, because all information about resources is available at the global scheduler. This advantage is naturally inherited by the proposed centralized SSN task allocation/scheduling approaches. However, the disadvantage is that the centralized scheduling approach is neither scalable nor fault-tolerant, especially when the global scheduler is suddenly disconnected from the network, which will cause the entire system malfunction. As opposed to the centralized approach, the distributed approach has more advantages regarding scalability and fault-tolerance. In such design, the central point is only used as a pool to store all the incoming applications. The task allocation/scheduling decision is collaboratively determined by multiple nodes, also known as local schedulers, and the selected node will retrieve the task from the task pool in either push or pull manner. To prevent serious task queuing in the task pool due to the large size of tasks waiting for processing resources, the distributed approach adopts an execution strategy that can dynamically assign tasks with different priorities, but it should not interfere with the planned scheduler. This execution strategy can be further utilized in SSNs since multiple time slacks exist in between tasks, thus small size tasks can be assigned with higher priority and execute in the suitable time slacks without breaking the existing schedulers. Nevertheless, this feature requires that the running time of tasks can be estimated. More importantly, in the distributed scheduling approach it is hard to schedule task efficiently and optimal solutions cannot be guaranteed due to the lack of the global information at

the decision stage. As a consequence, almost no SSN task scheduling/allocation approach is designed in a purely distributed manner. The hybrid scheduling approach is a positive combination of centralized approach and distributed approach and it allows different policies to be used for global and local scheduling. Up to present, only TPTS (LI *et al.* 2013) and HTPTS (LI *et al.* 2012) are designed as hybrid scheduling approach. Considering the characteristics of SSN, we believe that the hybrid approach is the best way to achieve the dynamic resource management requirement.

Collaboration is an increasingly important requirement in SSNs. The in-network processing cannot acquire accurate results by simply retrieving the information from a single sensor node. This requires the relevant sensor nodes work as a group to provide the necessary information for generating better results, value added information for the application. Each group is known as a virtual sensor network in SSNs and it is easier for the users to describe collaborative behaviour among nodes. The virtual sensor network can be established by adopting multiple criteria, energy efficiency being one of them. To better address this issue, data sharing between applications is either explicitly or implicitly enabled within a VSN. The solutions proposed by (NIRMALYA *et al.* 2010) and (FARIAS *et al.* 2013) exploited this feature.

Task persistence is a different way to realize collaboration in SSNs. Instead of defining a VSN and performing tasks on it, users only need to declare the data that they need and the required freshness of the data. If such previous data already exist, these (new) tasks will be automatically considered as the same (existing) task and the up-to-date results will be shared among them. A demonstration example of such solution is found in (LI *et al.* 2012). The collaboration among sensor nodes should be constantly guaranteed whatever data sharing mechanism is used.

The last but not least requirement listed in Section 3.1 is flexible dedicated platform construction, which is tightly coupled with other requirements. As we discussed earlier, the rationale behind SSNs is to create multiple VSNs to perform the applications, one per VSN. In order to hidden the underlying hardware details and enable user rapid application development and deployment, each VSN is usually presented and accessed through a common program interface to/by the users. This is generally accomplished by employing a middleware layer, such as TinySOA (AVILÉS- LÓPEZ and GARCÍA-MACÍAS 2009) and (DELICATO *et al.* 2005). One sensor node can simultaneously belong to more than one VSN, and it can also dynamically join or leave VSNs while the collaboration is required from different applications. The

collaboration among sensor nodes can be represented as data sharing between applications or a selected node can act as a communication relay to forward the intermediate results from the nodes belonging to other VSNs to the desired destination. In addition, the underlying hardware details can be hidden by using virtual machine solutions, such as: Maté (LEVIS and CULLER 2002) and Melete (YU *et al.* 2006), so that the users can fully focus on their application design.

3.9 Conclusions

The increasing capabilities of WSNs open exciting possibilities for their applications. In order to reduce the deployment and administrative costs, a noticeable design trend - shared sensor network, has been considered as a promising approach to support concurrent applications sharing the hardware resources from the same WSN, thus further increasing the usability and efficiency of the underlying infrastructure. In this chapter we presented a systematic literature survey conducted as part of this thesis in which we have provided a thorough search and study of the related WSN literature and try to cover the most important aspects of shared sensor network. However, given the broad aspects encompassed in the WSN field, and the dynamism of this research field, our survey is still by no means complete. To better contextualize our findings, we presented a simple but useful taxonomy that classifies the works of shared sensor network into three levels, namely node level, network level and system level, based on their degrees of abstraction. We also investigated the benefits that can be obtained from running multiple applications in a shared sensor network infrastructure. The final goal of our survey was to shed light on the requirements and implications of a SSN design and also on existing open issues and future directions towards fully exploiting the potential of such approach.

While this chapter aims at providing a comprehensive framework for guiding people researching and/or developing shared sensor network, we have left the issue of security unaddressed. Adopting an appropriate security framework in the design certainly has a positive influence on the overall network performance, but this is often driven by pragmatic issues such as the security level needed to be achieved, the possible interest conflicts between different authorities, etc. Based on our understanding of the shared sensor network, dominant among the security requirements is the need to support concurrent applications running on the same platform without causing interruption and to hide the underlying hardware details from the users to enable rapid SSN applications development and deployment subject to the different QoS requirements.

Through this survey we were able to find research opportunities that have guided the development of our framework. The proposed framework took into consideration the requirements found in Section 3.2. As stated in Chapter 1, we will present new MDFs for SSANs and new scheduling algorithms. The motivation behind these contributions is related to the opportunities envisioned in Sections 3.5.2 (Information Sharing) and 3.6 (System Service Level).

4 A Framework for developing Smart Spaces applications using a SSAN

In this thesis we present the proposal of a framework for developing Smart Spaces applications on a shared sensor networks, as well as an instance of it, called ASGARD, to enable the sharing of sensing and communication infrastructure among multiple applications. This framework is considered decentralized because the SSAN nodes collaborate among themselves to accomplish the designed functions.

The proposed framework includes several subsystems identified as being necessary for a SSAN used to support smart space applications. With the adoption of the framework it will be possible to coordinate the task scheduling of multiple applications that share a SSAN, seeking to enable interaction of multiple applications that compose the smart space in order to save resources (both of the network and the environment). It will also enable getting more precise answers about the state of a Smart Space through the use of data fusion methods tailored to SSAN context.

This chapter describes: (i) the logical architecture of the proposed framework, in terms of its elements and relationships among them; (ii) a description of ASGARD, an instance of the proposed framework; (iii) a description of the phases that define the operation flow of the proposed framework; (iv) conclusions.

4.1 Framework Logical architecture

The logical architecture of the proposed framework, shown in Figure 2, consists on the following subsystems: *Monitoring Manager*, *SSAN Manager*, *Decision Manager*, *Actuation Manager*, *Routing Manager*, *Fusion Manager* and *Scheduling Manager*.

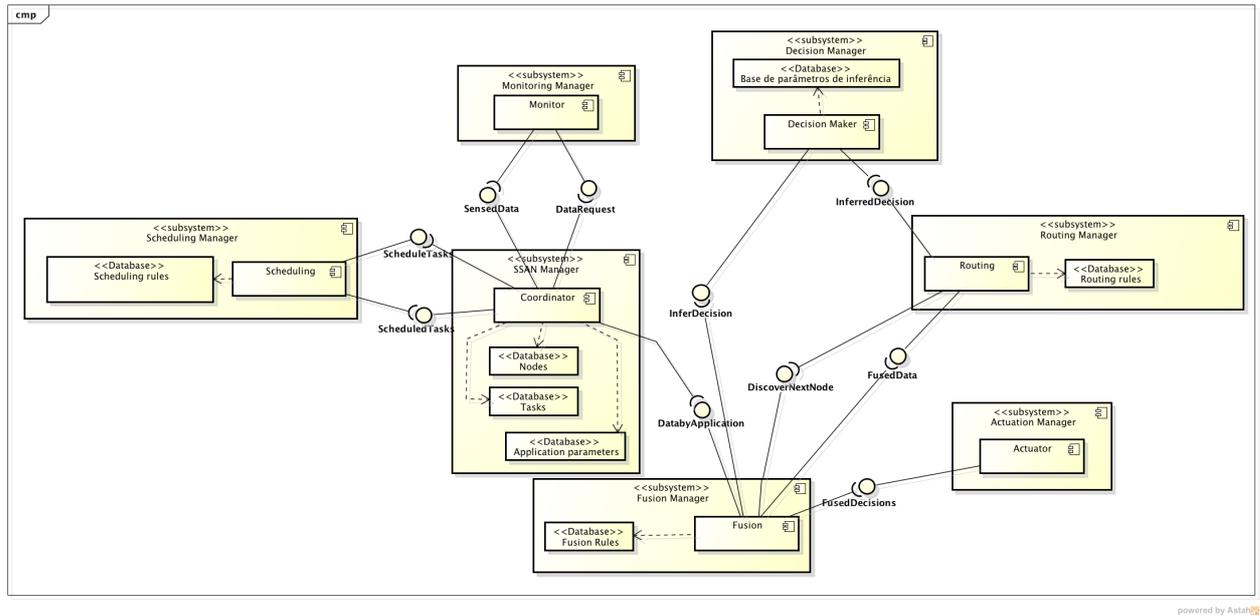


Figure 3. Logical architecture of the framework.

The *SSAN Manager* is the core subsystem of the architecture, responsible for managing the operation of all other components and subsystems and for coordinating the actions performed in the monitored environment. The *SSAN Manager* is responsible for managing the applications arrival in the SSAN, which means to insert in the *scheduling rules database* the *application ids*, the tasks that are to be executed. It is also responsible for grouping (by application) the collected data. The *SSAN Manager* subsystem is composed of 3 databases and a software component. The databases of this subsystem are the *parameters application database*, *task database* and *node base* that are accessed by a component called *coordinator*.

The *Monitoring Manager* coordinates the monitoring of the environment, that is, this subsystem must be, necessarily, deployed/installed in a physical device capable of performing sensing tasks. The subsystem interacts with the existing sensing units available in a sensor node. Each node can have one or more types of sensors capable of sensing different environmental variables such as light, temperature, vibration, etc. The *Monitoring Manager* subsystem is composed of a component called *Monitor*.

The *Actuation Manager* manages the operations performed over the monitored environment. Actuators are the SSAN physical devices responsible for acting upon the physical environment according to the decisions taken and sent by the *SSAN Manager*. Therefore, in order to reduce the

need for transmissions and consequently minimize the SSAN power consumption, this subsystem must be installed on *actuator* physical devices. If this subsystem isn't installed on an actuator, it becomes necessary communication between the physical device where the subsystem is installed and the actuator device, increasing the framework power consumption. Actuators can be electrical switches, relays, LEDs, among others, and they have the capability of turning on/off lamps, refrigerators, transmission lines, batteries, traffic lights or other devices present in the Smart Spaces. It is also important to emphasize that the actions performed by an actuator can be both logical and/or physical. In the same way an actuator can request the shutdown of a device, the actuator can ask certain applications to be withdrawn from SSAN due to a factor judged as a risk to the system, as malicious behavior or excessive consumption of resources. The *Actuation Manager* subsystem has a component called *Actuator*.

The *Fusion Manager* is the subsystem responsible for applying the data fusion techniques to the collected data and the inferred decisions. The *Fusion Manager* subsystem has a component called *Fusion* and a database called *fusion rule database*. The *Fusion* component is the responsible for performing the tasks of the *Fusion Manager* subsystem. The *Fusion Manager* subsystem addresses the fusion challenge presented in this thesis introduction.

The *Scheduling Manager* is the subsystem responsible for scheduling tasks of different applications in a SSAN. The *Scheduling Manager* subsystem has a component called *Scheduling* and a database called *scheduling rules database*. The *Scheduling* component is responsible for performing the tasks of the *Scheduling Manager* subsystem. The *Scheduling Manager* tackles the challenge of Scheduling presented in this thesis introduction.

The *Routing Manager* is the subsystem responsible for performing the messages routing in a SSAN. This is the subsystem responsible for managing the sending of messages between different nodes in the network. The *Routing Manager* subsystem comprises a component called *Routing* and a database called *routing rules database*.

The *Decision Manager* is the subsystem responsible for processing the collected data, making decisions for each active application and performing, wherever possible, the integration of applications running in the SSAN. The *Decision Manager* Subsystem is composed of a component called decision maker and a base of rule called *inference parameters database*. The

Decision maker component is the one responsible for performing the tasks of the *Decision Manager* subsystem. This subsystem has been the subject of investigation of another member of the research team which the author integrates (SOARES *et al.* 2012).

4.2 Databases Description

The framework architecture includes the following databases: *Fusion rules*, *Scheduling rules*, *Application parameters*, *Routing rules*, *Node*, *Tasks* and *inference parameters*.

The *routing rules database* contains for each application, its identifier (an integer of 16 bits) and types of messages (control and data messages) along with their respective sizes. This happens since different applications may have different needs. There are two types of control messages of the framework: messages transmitted by the *Actuation Manager* Subsystem to outside the SSAN (for recording actions taken), with 15 bytes in size, and messages transmitted to the *Actuation Manager* from outside the SSAN (an order given by a human operator), with 16 bytes in size. The data messages have 16 bytes in size.

The *inference parameters database* defines the input and output variables of the inference method chosen. It also stores the rules for application integration. Each rule takes the following format: IF (*condition*) THEN *id_action* (*parameters*), and it is associated with an application. The *id_action* is an integer value that represents the implementation of the Actuation mechanism present in the *Actuation Manager*, which can receive parameters for activation. As an example of integration rule, the lighting application is correlated to the HVAC application by the output variable "controlellum". It is specified in the database that, IF the output variable "controlellum" of lighting application is in the "Off" state (controlellum = 0) THEN the output variables of HVAC application also should be found in the "Off" state (control_temp = 0).

The *tasks database* contains the identifiers for the tasks (one 16-bit integer) that the framework can perform. These tasks are methods implemented in the framework and they are divided into 3 groups: reading, decision and sending Messages. The Reading tasks represent the collection of data from the monitored environment; the decision task involves decisions taken for the applications; sending Messages represent the exchange of messages between the nodes in the SSAN.

The *scheduling rules database* contains, for each application to be scheduled, an identifier of the application and the set of identifiers of the tasks in which the application is subdivided. For example, a simplified fire detection application could be composed of a reading task (Temperature), a decision task (if the temperature is higher than a certain threshold) and a task of sending messages to another node.

The *fusion rules database* contains identifiers for each fusion method (MDF IDs) and the application list (the respective identifiers) associated with a particular fusion method.

The *application parameters database* contains the application identifiers, their priority (represented as an 8 bit integer value given by an expert in the application domain), types of sensors required (sensing interfaces, 8 bit integers), sensing rates, data sending rate and the frequency (in milliseconds) with which the decision process of each application is triggered.

The *node database* contains information about the sensor nodes available in the network. For each node, there is an identifier (a 16-bit integer), its sensing capability (identifiers the sensing interfaces), current status (residual energy and operation mode, that is, the role played by the node in the network) and geographical location (a tuple with two integers {x, y, z} representing the geographical coordinates).

4.3 Interfaces Description

In this subsection the interfaces used to exchange information between different subsystems of the proposed framework are described. The interfaces are presented in the following way: interface_name (provider_component, requester_component).

DataRequest Interface (*SSAN Manager, Monitoring Manager*) - used by the *SSAN Manager* subsystem for notifying the *Monitoring Manager* which sensing interfaces must be activated according to the applications requirements. Contains the primitive described below:

- Sensing – primitive used by applications to request the collection of data, that is, an information collection about a specific environment variable. This primitive can be implemented synchronously, in the case of an instantaneous query about the monitored phenomenon, or asynchronously, when it wants to monitor events or consultations (periodic) of long duration.

ScheduleTasks Interface (*SSAN Manager, Scheduling Manager*) - used by *SSAN Manager* subsystem for selecting which tasks must be performed by the framework and in which order, given the applications present in the network. Contains the primitive described below:

- Schedule – Primitive used to request the scheduling of tasks performed by a set of applications. This primitive sends the applications identifiers and a list of sensors available (identifiers and sensing interfaces, the description of the tasks is in *Tasks database*) desired for the *Scheduling Manager Subsystem*.

DatabyApplication Interface (*SSAN Manager, Fusion Manager*) - used by the *SSAN Manager* subsystem to group data per application for these to be sent to the *Fusion Manager* subsystem. Contains the primitive, listed below:

- DataFusion - primitive used to request to apply a fusion technique on the collected data. This primitive sends a data vector to the *Fusion Manager* subsystem as well as to which applications these data should be fused.

SensedData Interface (*Monitoring Manager, SSAN Manager*) - used by the *Monitoring Manager* subsystem to send the sensed data to the *SSAN Manager* subsystem. Contains the primitive, listed below:

- ReturnData - primitive used to return the collected data grouped by application. That is, each data set associated with an application identifier. This primitive can be deployed synchronously, in the case of an instantaneous query about the monitored phenomenon, or asynchronously when you want to monitor events or perform queries of long duration.

FusedData Interface (*Fusion Manager, Routing Manager*) - used by the *Fusion Manager* subsystem to pass the result of a data fusion done with the data that came from the *Monitoring Manager* subsystem for the *Routing Manager* subsystem. Contains the primitive below:

- FusionReturn - primitive used to return the result of a fusion. That is, to transmit the result of the fusion method used, the identification of the applications that will use this result and the identifier of the fusion method used.

ScheduledTasks Interface (*Scheduling Manager, SSAN Manager*) - used by the *Scheduling Manager* subsystem to transmit to the *SSAN Manager* subsystem which tasks will be performed (transmission of identifiers of tasks) and by which sensors present in the network. Contains the primitive below:

- TaskListReturn - primitive used to return the list of scheduled tasks. Returns a list with the application identifiers, a list of the task identifiers and sensors associated with them.

DiscoverNextNode Interface (*Routing Manager, Fusion Manager*) - used by the *Routing Manager* subsystem to select the next node in the network (using a routing protocol). Contains the primitive, listed below:

- Route - primitive used to choose the next node in a SSAN. Returns an application identifier and the list with the tasks identifiers.

InferredDecision Interface (*Decision Manager, Routing Manager*) - used by the *Decision Manager* subsystem to transmit a decision (the result and the identifiers of the associated applications to the decision) to the *Routing Manager* subsystem, this subsystem in turn sends the decision to *the Actuation Manager*. Contains the primitive listed below:

- DecisionReturn - primitive used to return the result of a decision. That is, transmit the result of the inference method used and the identification of applications that will use this result.

InferDecision Interface (*Fusion Manager, Decision Manager*) - used by the *Fusion Manager* subsystem to transmit the result of a data fusion to the *Decision Manager*, where it will be used an inference method to evaluate the data.

- InferDecision - primitive used to transmit the result of a data fusion (including results from the fusion method, fusion method identifier and the application identifier of the application interested in this result).

FusedDecision Interface (*Fusion Manager, Actuation Manager*) - used by the *Fusion Manager* subsystem to transmit a fusion of a set of decisions that comes in the subsystem to the *Actuation Manager* subsystem. Contains the primitive, listed below:

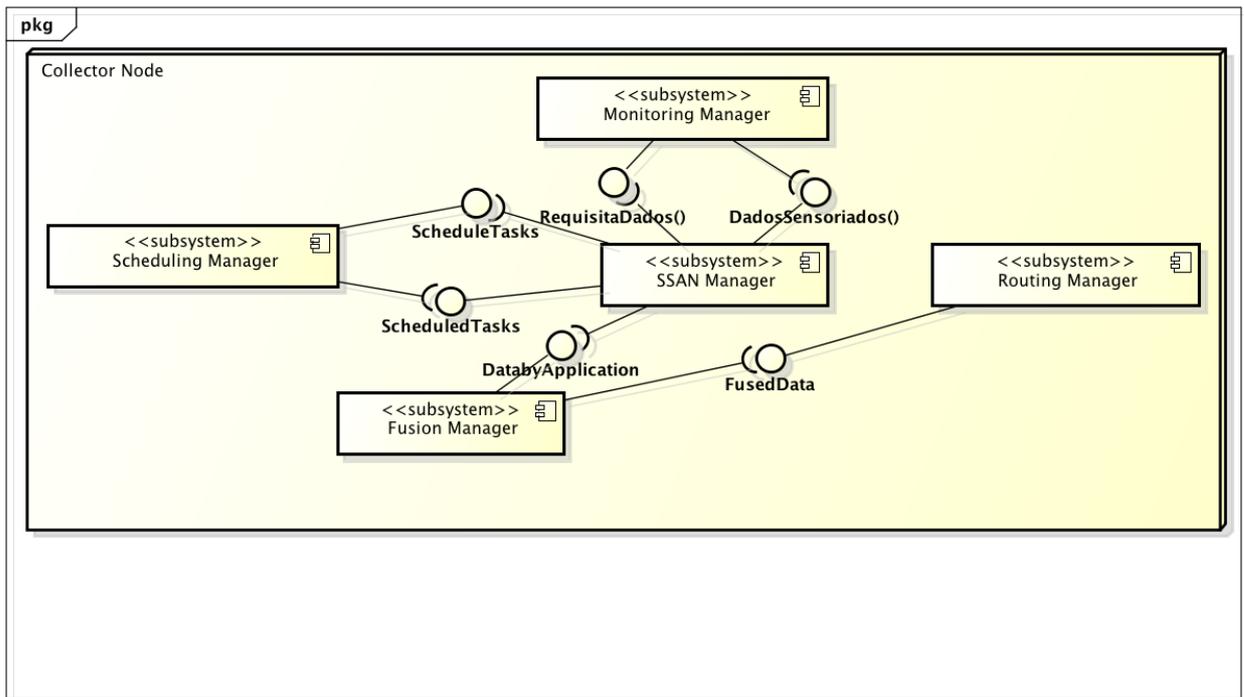
- ReturnFusionDecision - primitive used to return the result of a decision fusion. That is, to transmit the result of the fusion method used, applications identifier that will use this result and the identifier of the fusion method used.

4.4 Physical Architecture

To better describe our framework an instance of it was developed, called ASGARD.

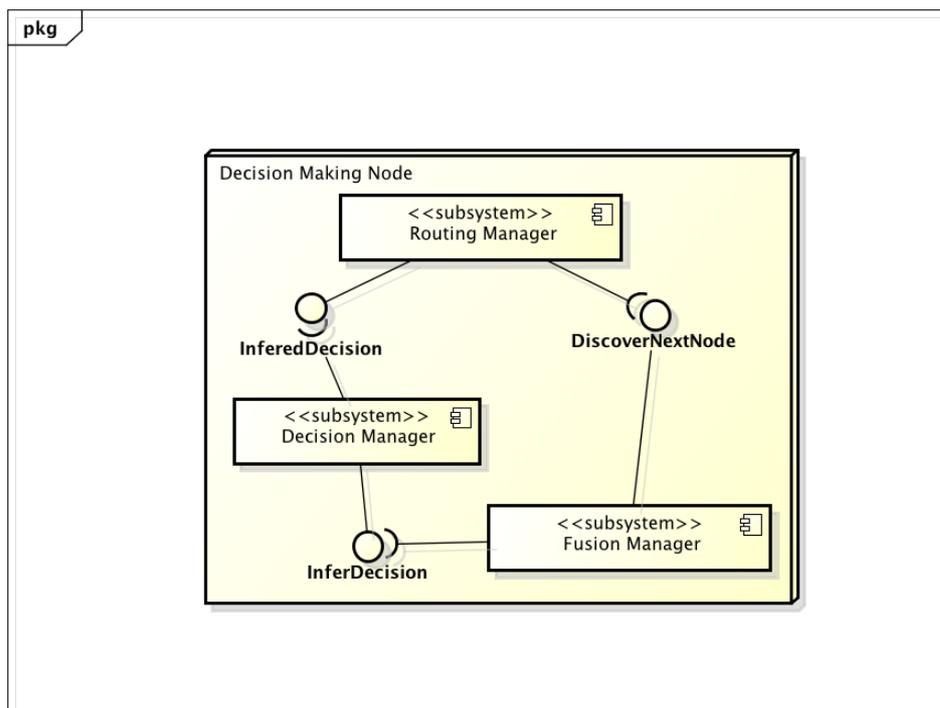
A SSAN consists of several sensor nodes, actuators, and a sink node. A sensor node can assume the role of collector node (CN) or decision making node (DN). The actuators will be called actuator nodes (ATN). Roles are related to responsibilities of each node and their functionality. The CNs are the sensor nodes that contain the sensing units and, therefore, they are responsible for collecting data and sending them to the DN. The DN is a sensor node that does not need to contain sensing units. Thus, the DN is responsible for evaluating the data brought by CNs and makes decisions about the applications and sends decisions to the sink node and commands to the actuators. Finally, the actuators are responsible for acting physically on the environment. A node cannot be both CN and DN (for reducing energy consumption). The nodes that assume the role of DN are previously chosen before the framework start.

DN contains the *Fusion Manager*, *Routing Manager* and *Decision Manager*. CN node contains the subsystems: *SSAN Manager*, *Monitoring Manager*, *Routing Manager*, *Scheduling Manager* and *Fusion Manager* Subsystems. Finally, ATN contains the subsystems: *Fusion Manager*, *Routing Manager* and *Actuation Manager*. The Deployment diagrams of each of the roles can be seen in Figures 4, 5 and 6.



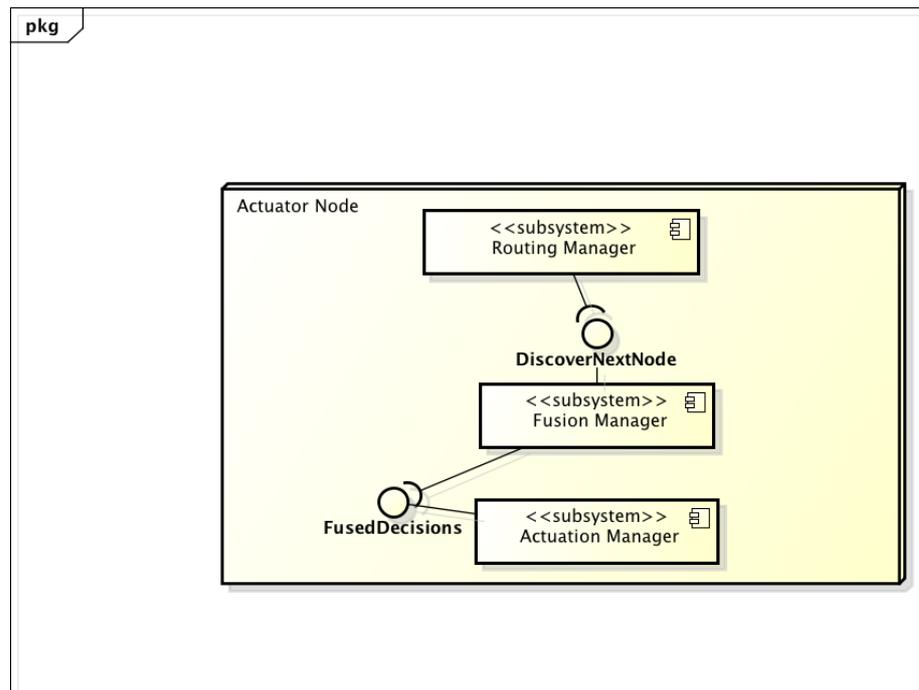
powered by Astah

Figure 4. Collector Nodes Deployment Diagram.



powered by Astah

Figure 5. Decision Maker Nodes Deployment Diagram.



powered by Astah

Figure 6. Actuator Nodes Deployment Diagram.

4.5 ASGARD's Implementation

ASGARD was developed in the SUN SPOT platform Version 8 using the Netbeans IDE Version 7.4 (WILDE, GUINARD and TRIFA 2010). The Sun SPOT is a commercial sensor platform that is particularly suitable for rapid development and demonstration of WSN applications. The SUN SPOT SDK environment includes Solarium, that contains a SPOT emulator that is useful for testing software created using SUNSPOT and/or creating scenarios with a large number of nodes. In all the following experiments, we used a mix of sensor nodes (real nodes and virtual nodes emulated by Solarium) to ensure we can fully study how algorithm performs in large scale WSNs. The Subsystems components were implemented as classes with the same name given in the Logical Architecture.

Our framework was built on top of a SUN SPOT implementation of SenShare (LEONTIADIS *et al.* 2012), an architecture for SSAN that allows the creation of an overlapped WSN over a shared physical infrastructure. In an overlapped network each sensor node allows that multiple co-sited applications use the node's hardware resources. An application can be implemented over the entire infrastructure or over a node subset of the physical network. SenShare provides a clear

separation between the system infrastructure and applications, allowing the management and ownership of infrastructure and applications to be split by different authorities. The goal of SenShare is to map applications on physical nodes to operate in a virtual environment built on top of the real network. From the point of view of application, it is designed to operate in a overlapped WSN, while the point of view of SSAN, the application is allocated in a selected subset nodes. The overlapping WSN can share real SSAN nodes.

By adopting the SenShare as underlying infrastructure for SSAN, the Sink Node (SN, a node that stores and controls the data collected by other sensor nodes) is responsible for the formation of overlapped WSN. Each application represents an overlapped WSN on SenShare. Through the use of the SenShare infrastructure, ensures that all intermediate results of an application can freely travel on between nodes that form the overlapped network and reach to the SN. For the formation of the overlapped network using Senshare, the SN will choose the CN who should receive a particular application and sends a control message to the *SSAN Manager* subsystem.

In order to implement ASGARD, we have implemented 8 java classes, named: Actuator, Decision, Scheduling, Fusion, SSAN_manager, Monitor, Routing and StartApplication. Each database was implemented as an ASCII file containing the information described in the Section 4.2. Since the *Fusion Manager* and *Scheduling Manager* subsystems are main topics of this thesis, we have made some simplifications to the other subsystems.

StartApplication is a default SunSPOT class that has the method *StartApp()*. This method is responsible for starting the framework's operation. The SSAN_Manager class queries the databases *Tasks*, *Nodes* and *Application parameters*, and sends data to the other components. The Monitor class, that implements the functionalities of the *Monitoring Manager*, coordinates the task of monitoring environmental physical data; the class is basically an API to the sensing devices in a sensor node.

The Decision class that is an implementation of the *Decision Manager* Subsystem was based on the work described in (FARIAS *et al.* 2014), consisted of sets of simple rules that assign decisions depending on the input, one for each application running in the SSAN. In such method, the input space is divided into fixed size subspaces and for each subspace a set of decisions is assigned. In other words, for any input values that belong to the same subspace, the *Decision Manager* will always have the same response, making the same decision. Therefore, for each

active application, there is a set of rules used by the *Decision Manager*. Each rule consists of one or more input intervals, i.e. a range of values that a given variable in a given application can have as conditions and Actions to be taken if these conditions are matched. Thus, an example of a rule of a HVAC application might be as follows: IF {Temperature is in [25, 31[} THEN {(termostat - 3°C) and (Ventilation + 25%)}. For each active application, the associated decisions are stored in the Local Decision Parameters database. *Decision Manager* makes decisions by comparing input data with parameters in the database. Whenever an input data satisfies the conditions of a particular rule, the decision associated with such rule is transmitted to the *Manager*. The integration performed by the *Decision Manager* consists of sets of simple rules that may chain decisions. The input data for the integration are the decisions previously made by the *Decision Manager* Subsystem. Each input decision is handled individually, even if two or more input decisions belong to the same application and have been made simultaneously. The input space for integration is divided into fixed size subspaces and for each subspace a set of decisions is assigned. In other words, for any input decision that belongs to the same subspace, the *Decision Manager* will always have the same response for integration. Therefore, for each active application, there is a set of rules used by the Integration Method. Each integration rule consists of an input decision and output decisions.

For the Routing class, we have used LQRP (Link Quality Routing Protocol) (PASCHOALINO and MADEIRA 2007) as routing protocol to transmit data among sensor nodes. The Actuation class simply emulates the activity of an Actuator by printing a warning in the screen.

The Fusion class implements the *Fusion Manager Subsystem*, shown in Figure 7. In this class, each MDF is a method. When a set of data arrives via an input interface, categorized data for the corresponding applications (identified by an application ID, an 8 bits integer) are sent to a particular fusion method (identified by an MDF ID). As stated in Chapter 1, new MDFs for the SSAN scenario will be presented on Chapter 5.

The Scheduling class implements the *Scheduling Manager subsystem*, shown in Figure 8. The Scheduling class is responsible for executing the scheduling algorithms proposed in Chapters 6 and 7, as stated in Chapter 1.

ASGARD implementation was deployed in the flash memory of the Sun SPOT devices. The SPOT platform used in this work has 4 Mbytes of flash memory. The Framework developed is

stored in the flash memory with the Java virtual machine used by the platform, which occupies 778 kbytes. Besides the virtual machine, the Sun SPOT reserves another 128 kbytes for other activities such as sensor resource management. The Fusion class consumed 2956 bytes, Scheduling class consumed 1101 bytes, SSAN_Manager 251 bytes, Routing 357 bytes, Actuator 194 bytes, Decision 1349 bytes, Monitoring 496 bytes, Start_App 458 bytes. All databases consumed 1213 bytes of Flash memory. The framework consumed 712 kB of RAM memory.

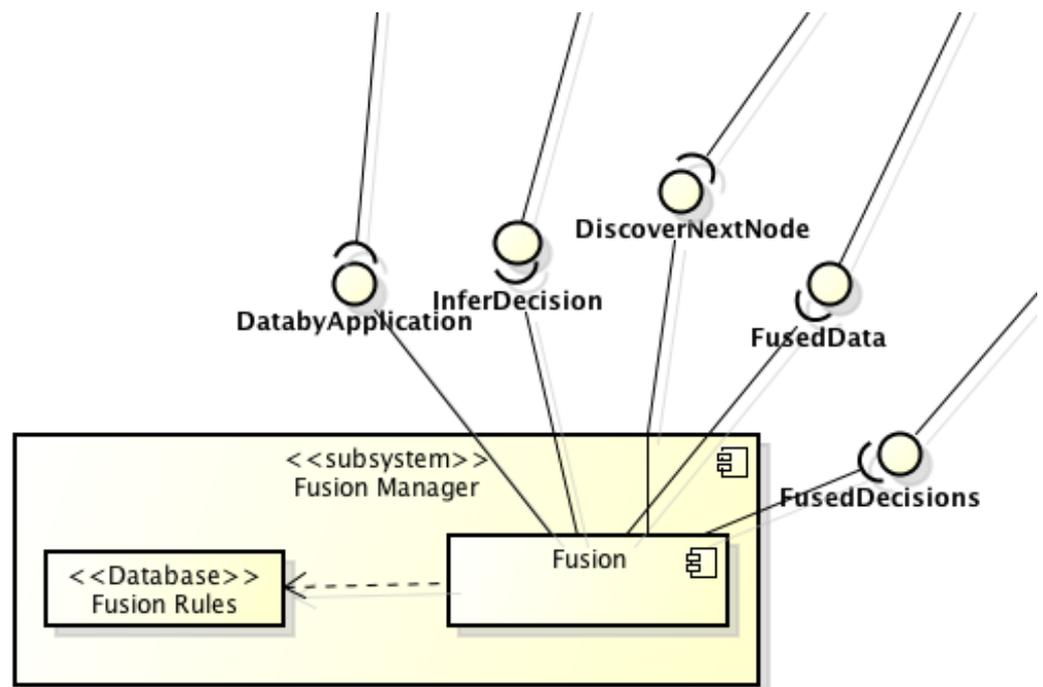


Figure 7. Fusion Manager Subsystem.

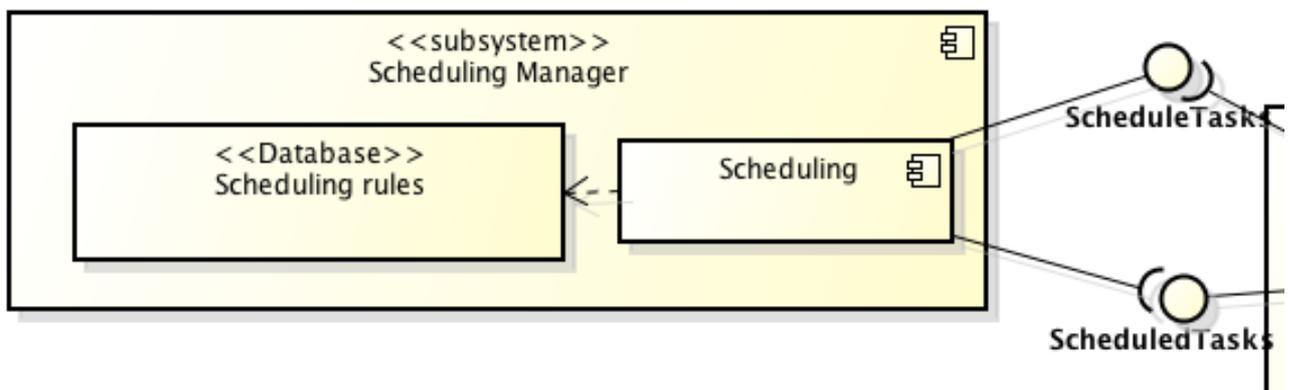


Figure 8. Scheduling Manager Subsystem.

4.6 Description of ASGARD’s Operation

The operation flow of the proposed framework is divided into two phases: **Initialization** and **Operation Cycle**. In the first, called **initialization**, the operating parameters of ASGARD are determined. It is at this stage that the databases (*Fusion rules* database, *Scheduling rules* database, *Application parameters* database, *Routing rules* database, *Node* database, *Tasks* database and *Inference parameters* base) are created and the necessary data for the operation of the framework are inserted in the databases by the *Manager* subsystem. Finally, a set of applications are chosen to be placed in execution when the system starts. In this phase, the applications to be performed on the SSAN are defined and their parameters are configured/initialized, as well as sensors to be activated.

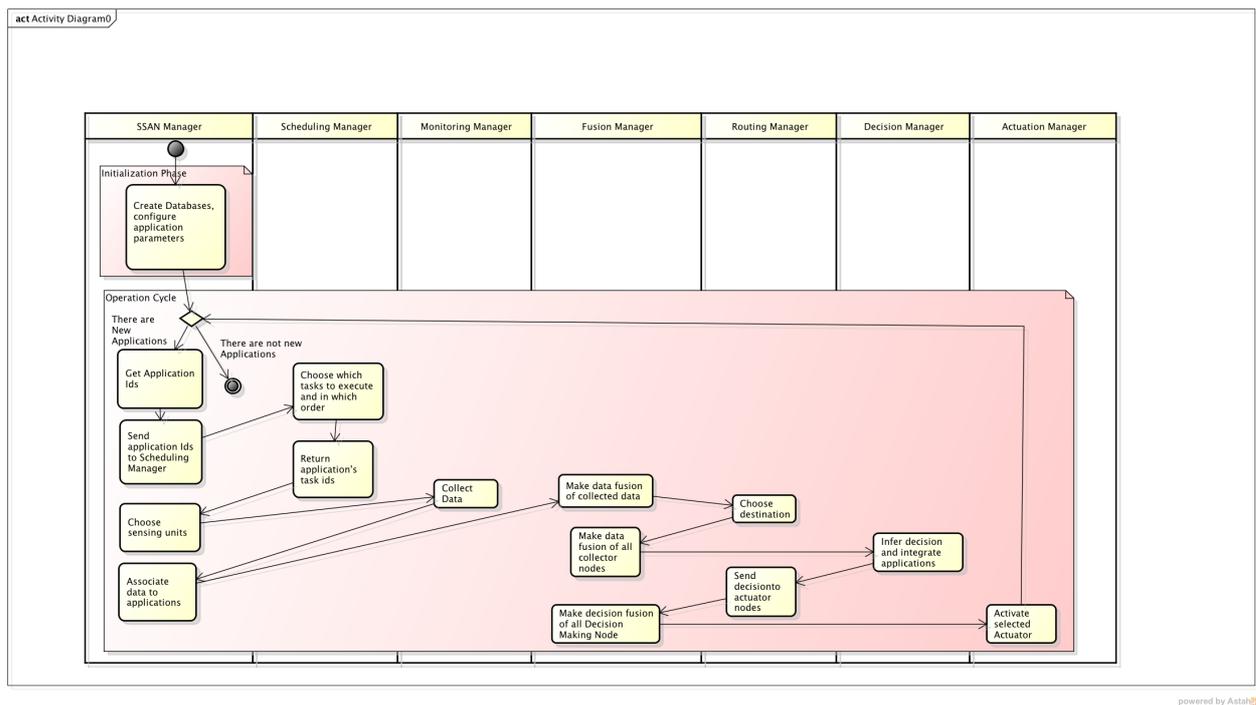


Figure 9. ASGARD Activity Diagram.

In the second stage, ASGARD enters a loop; each application will execute a series of procedures during this loop. The second stage consists in the **system operation cycle**, as shown by the activity diagram in Figure 9. In this phase, in the collector node, for each application, the SSAN

Manager triggers a set of procedures. Initially, the *SSAN Manager*, for each application obtains its identification on the *Applications parameters* database and sends it to the *Scheduling Manager*. This in turn will schedule the multiple applications and verify the tasks in common in order to save resources. Once the tasks to be performed have been decided (the tasks to be performed are located at the *Task* database), the applications and tasks identifiers are passed to the *Manager* subsystem. The *Manager* now will send a message to the *Monitoring Manager* subsystem telling what kinds of data should be collected.

The *Monitoring Manager* subsystem collects this data, regardless of the applications, and sends it to the *SSAN Manager* that associates these data to the applications. Then, the data collected are grouped by application and sent to the *Fusion Manager* that will perform data fusion. The *Fusion Manager* performs data fusion for multiple applications and sends the response to the *Routing Manager* that sends a message to the DMs.

In the Decision Maker Node, the data coming from the CN will be fused by the *Fusion Manager*, which in turn will pass the fused data, as well as the IDs of the applications to the *Decision Manager*. The *Decision Manager* process the data and takes the appropriated decisions for each application separately like in the case of battery monitoring application (BM), where an example of a rule is: "if energy demand is greater than the current supply, access the battery". Then the *Decision Manager* will consult the *Inference parameters* database to search for possible integrations to these different applications. Once these decisions are made, the *Decision Manager* will query the *Inference parameter* database to integrate different applications. At this time, the *Decision Manager* will send the resulting decisions, as well as the Id for each application to the *Routing Manager*. This in turn will send the resulting decisions and the Id of each application to an Actuator Node.

In the Actuator Node, the decisions coming from the DMs are fused by the *Fusion Manager*, which in turn will pass the fused data, as well as the application ids to the *Actuation Manager*. The *Actuation Manager* receives the data sent by the *Manager* indicating which decisions must be executed and sends a control message out of SSAN, recording the action executed and whose purpose is to control the operation and auditing the decision system.

Besides this whole process managed by the *SSAN Manager* subsystem, such subsystem can receive asynchronously, control messages from outside the SSAN. These messages are sent by a

system operator. This type of operation is used to adjust the SSAN operation, and contains the id of the application to be modified, or any other adjustments. The possible settings are: (i) modify the sensing rate, (ii) enable or disable a particular application whose databases are already loaded on the applications base, which would generate a rescheduling of activities that are already in the network and (iii) to insert a new application on the network, that is, add new information on the applications parameters base.

4.7 Conclusion

This Chapter presented a decentralized framework for developing Smart Spaces applications using SSANs, as well as an instance of this framework, called ASGARD. The subsystems, components and databases that comprise the framework, as well as the communication interfaces were presented. The description of the framework operation was also presented. As future work, we intend to incorporate into the framework subsystems that deal with applications security and other application requirements.

5 Multisensor Data Fusion in Shared Sensor and Actuator Networks

This chapter presents MDFs for SSAN. We called them Enhanced MDFs that appropriately manipulate each sensing data, data range and requirement of each application in order to avoid a single application to dominate the entire MDF process and also to take into account the different semantics and priorities of these applications. The enhanced MDFs can thus deal with different requirements of the simultaneously running applications. The content of this chapter is based on the papers published on IEEE Local Computer Networks Conference - LCN 2012: “Information Fusion Techniques applied to Shared Sensor and Actuator Networks” and on International Conference on Information Fusion - FUSION 2014.

This Chapter is organized as follows. Section 5.1 presents an Introduction. Section 5.2 presents some real-world examples that motivate us to extend the traditional MDFs to the SSNs. Section 5.3 introduces the enhanced MDFs. Section 5.4 conducts comprehensive comparative studies on MDFs from both simulations and implementations; the detailed discussion is also presented. Section 5.5 contains conclusions and points out the planned future works.

5.1 Introduction

Recent years have witnessed the emergence of the Shared Sensor and Actuator Networks (SSANs), which instead of assuming an application-specific design, allow the sensing and communication infrastructure to be shared among multiple applications. With an increasing number of sharing applications, a growing amount of sensor-generated data will be produced, from which useful information can be extracted. However, wireless sensors and actuators commonly rely on batteries as their energy sources, whose replacement is undesirable or unfeasible. Thus, to reduce the amount of data to be transmitted, thus saving energy, Multisensor Data Fusion Techniques (MDF) can be employed. It can also be used to enhance data accuracy in the SSAN scenario and to achieve inferences that are not feasible from a single sensor or source. Existing MDFs are still being utilized following an application-specific design. We present an adaptation of well-known MDFs to deal with multiple applications simultaneously in the SSANs context. Our proposal is validated through simulations and tests on real nodes in the domain of Smart Grid applications.

(NAKAMURA, LOUREIRO and FRERY, 2007)

As stated in chapter 1, a potential benefit of such design approach is to increase the utilization of sensing and communication resources, whenever the underlying network infrastructure covers the same geographic area and the sensor nodes monitor the same physical variables of common interest for different applications. On the other hand, compared with the existing application-specific design, the shared sensor network approach poses several research challenges at different aspects of WSNs.

One of such challenges is how to efficiently adapt the traditional MDFs to the SSAN scenarios, since those MDFs are all designed for an application-specific network design. This means that traditional MDFs process all the sensed data under the specific data semantics of a single target application. Nevertheless in order to execute proper and efficiently in SSAN scenarios that encompass multiple applications, traditional MDFs have to consider the distinct data semantics of each application. By data semantics we mean a pattern that describes an application and will make possible interoperability and integration between applications (WAGNER, SPEISER and HARTH 2010). This problem, to deal with distinct data semantics of each application, is particularly important and common in a distributed fusion system (TAPIA, BAJO and CORCHADO 2010); we will call it application correlation in this work. If such semantic differences were not taken into account, the fusion techniques would produce unreliable results for different applications. So, by taking the semantics into account, it is possible to enhance the MDF's accuracy. Besides the fact that the traditional MDFs process all the sensed data under the specific data semantics of a single target application, they also assume that all the sensed data are weighted and handled equally and have the same data range. In SSAN, MDFs need to consider that the same sensed data may have different degrees of importance for different applications and also different data ranges.

Considering the aforementioned discussion, we claim that existing MDFs are not suitable to be used in a SSAN scenario, since they were not conceived taking into account the SSAN specific features and needs. Therefore, there is a need for adapting such MDFs to deal with these features in order to achieve energy efficiency and reliable results in this emergent scenario. In this paper, we present our proposed adaptation of several well-known MDFs to the SSAN scenario (EFSTRATIOU, LEONTIADIS, MASCOLO and CROWCROFT 2010), namely, fault tolerant

interval algorithm, Bayesian inference, Dempster-Shafer inference, False Tolerant Interval and moving average filter.

The objective of the subsystem is to enable that enhanced MDFs appropriately manipulate each sensing data, data range and requirement of each application in order to avoid a single application to dominate the entire MDF process and also to take into account the different semantics and priorities of these applications. The enhanced MDFs can thus be dealt with different requirements of the simultaneously running applications. Our enhanced MDFs are highly likely to provide substantial energy saving for applications, since these MDFs can be executed only once for multiple requests. Compared to traditional MDF approaches, the energy consumption using the enhanced MDFs is reduced, as tests will show. Besides saving energy, our enhanced MDFs are also able to preserve the data semantics, assure and enhance the data accuracy in SSANs since it also seeks to combine information from multiple sensors, sources and applications to achieve inferences that are not feasible from a single sensor or source through probabilistic methods.

We will validate our enhanced MDFs using the Smart Grid as a motivational scenario through both simulations experiments and experiment with real sensor nodes platforms. Preliminary results of the proposed MDFs were presented in (FARIAS *et al.* 2012). In our former work, De Farias *et al.* presented a lightweight extension of the traditional moving average filter techniques (MAF), called EMAF, for properly addressing the same sensed data having different degrees of importance for different applications that have different data range. In this chapter, we will present the description of the enhanced versions of MDFs along with their evaluation in the Smart Grid application domain, namely: (i) a lightweight extension of the traditional Bayesian Inference to the scenario of SSANs, Enhanced Bayesian Inference; (ii) a lightweight extension of the traditional Dempster-Shafer Inference to the scenario of SSANs, Enhanced Dempster-Shafer; (iii) a lightweight extension of the traditional Moving Average Filter to the scenario of SSANs, Enhanced Moving Average Filter presented in (FARIAS *et al.* 2012) and (iv) a lightweight extension of the traditional Fault Tolerant Averaging to the scenario of SSANs, Enhanced Fault Tolerant Algorithm. These enhancements have as purpose allow the MDF to work with data coming from different applications with distinct semantics. As presented in Chapter 1, the goal of data fusion is to reduce the energy consumption of sensor nodes and enhance accuracy of event detection in terms of true occurrence of an event.

5.2 Motivation Scenario

In order to motivate our proposal, we present a scenario based on one of the suitable applications for SSANs: the Smart Grid (FARIAS *et al.* 2012). The Smart Grid is a modern electric power-grid infrastructure for improved efficiency, reliability, and safety, with smooth integration of renewable and alternative energy sources, through automated control and modern communication technologies (GÜNGÖR *et al.* 2009). It contains three major subsystems: power generation, power delivery, and power utilization. In such system, reliable and online information is considered as the key factor for reliable delivery of power from the generation units to the end users. The impact of equipment failures, capacity limitations, and natural accidents and catastrophes, which cause power disturbances and outages, can be mostly avoided by online power system condition monitoring, diagnostics, and protection. In this respect, SSANs have been widely recognized as a promising technology that can effectively solve the aforementioned issues, making SSANs a vital component of the Smart Grid. According to different purposes, several applications, e.g. Battery Monitoring (BM), Overhead Power Line Monitoring (OPLM), security management, fire detection, and Structural Health Monitoring (SHM), have chances to coexist in the same power grid for delivering reliable service to the end users.

In this case study, we mainly focus on how a common but important component of the Smart Grid scenario, the transmission tower (Figure 10), can benefit from the use of a SSAN design. A transmission tower (colloquially termed an electricity pylon in the United Kingdom and in parts of Europe, an ironman in Australia, and a hydro tower in parts of Canada) is a tall structure, usually a steel lattice tower, used to support an overhead power line. An overhead power line is an electric power transmission line suspended by towers. Since most of the insulation (resistance to the flow of electric current, magnetism and heat) is provided by air, the overhead power lines are generally the lowest-cost option for transmitting electric energy in large scale. As the lowest-cost option, in the developing countries, the overhead power-lines are the most common mean to transmit energy. In the Smart Grid context, the transmission tower is also responsible for storing energy using local batteries.

We have chosen two applications used in the transmission tower to test our enhanced MDFs: the overhead power line monitoring (OLPM) and the battery monitoring (BM). The OLPM application is of great importance because the loads of the power lines keep increasing in the past

few years. The knowledge about the power line temperature is necessary for determining its loading. The increasing load of the power lines causes more electric energy turns into heat. The presence of overheating in the overhead power lines could damage the power lines themselves, resulting in transmission failures in these lines or even ending up with blackout. This happens because the heat will damage the line, thus disrupting the electrical flow. The usual and safe line temperature is around 40°C - 65°C (HUNG, LEE, LUI, YANG, and ZHONG, 2010) (GAL, OLTEAN, BRABETE, RODEAN, and OPINCARU, 2011). The actuator in this case will cut off the power flow preserving the line until its temperature is reduced.

In the BM application, local batteries are seen as a solution for solving the problem of unexpected peaks in the electricity demand (DENHOLM *et al.* 2005), where peaks of demand could be addressed by the energy stored in the batteries that would reduce the number of blackouts. The BM application is also of great importance because overheating could damage the battery; therefore the temperature must be monitored. Additionally, high temperature could be an evidence of energy waste (BIYABANI 2009) (WU *et al.* 2010). The usual battery temperature is around 40°C – 144°C. In the presence of overheating, a BM application will turn off the battery. This happens because when a given temperature threshold is reached, and this threshold represents that the battery is almost fully charged and the temperatures beyond this threshold may cause battery damage. So, there is no need of continuing to use a BM application. The actuator in this case will turn the battery off until its temperature is reduced.

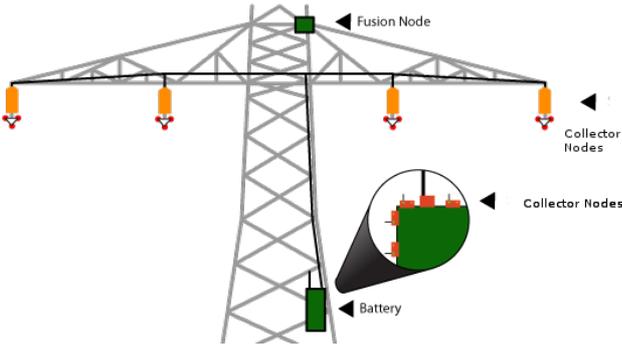


Figure 10. Transmission Tower in the Smart Grid

5.3 Enhanced Information Fusion Techniques

As previously mentioned, existing MDFs have been successfully applied to traditional WSNs, but they are not tailored to the SSAN scenarios. To address this issue, we propose extensions for the

selected MDFs, including Bayesian Inference, Dempster-Shafer Inference, Moving Average Filter and Fault Tolerant Interval. To distinguish between our proposed and the traditional algorithms, we name our MDFs as enhanced versions of the corresponding MDFs in the remainder of the text. For instance, our extended Bayesian inference will appear as enhanced Bayesian inference.

5.3.1 Enhanced Bayesian Inference

In the traditional probability techniques such as the Bayesian Inference (presented in chapter 2), an application has a set of states, which describes its behavior, using the applications presented in section 5.2, states could be battery damaged, battery healthy, power line damaged and power line healthy. The data used by the MDF to achieve these states would be gathered by the SSAN. This set of states does not consider the correlation between applications; each application has its own set isolated from any other application. This approach masks the fact that the behavior of an application can affect the behavior of another application, thus presenting a potential drawback in the SSAN scenario. In order to extend the Bayesian Inference, we propose to evaluate each application separated and then to eliminate all redundant states. In Enhanced Bayesian Inference, represented by Equation 8, $\Pr(Y | X)$ represents the belief of hypothesis that an application Y will have a determined behavior given the result of an application X.

$$BI(Y|X) = \frac{BI(X|Y) BI(Y)}{BI(X)} \quad (8)$$

In equation 8, $BI(Y)$ and $BI(X)$ are Bayesian Inferences that consider the set of states of each application isolated. However, this modification still suffers from the traditional problem of the Bayesian Inference: probabilities must be known before applications begin their operations. In order to solve this problem, we adapted the Dempster-Shafer Inference, which is a variant of the probability techniques that has overcome this limitation. The next subsection describes our Enhanced Dempster-Shafer Inference.

5.3.2 Enhanced Dempster-Shafer

In the traditional Dempster-Shafer, presented in chapter 2, an application has a set of states, similarly to the Bayesian Inference, using the applications presented in section 5.2 states could be battery damaged, battery healthy, power line damaged and power line healthy. The data used by

the MDF to achieve these states would be gathered by the SSAN. The same problems related to the Bayesian Inference could be found in the Dempster-Shafer: the set of states does not consider the correlation between applications; each application has its own set isolated from any other application. This approach masks the fact that the behavior of an application can affect the behavior of another application, thus presenting a potential drawback in the SSAN scenario. In the Dempster-Shafer inference is even worse due to the fact that the beliefs are dynamically built as the WSN collects data. In order to extend the Dempster-Shafer Inference, similarly to the Bayesian Inference, we propose, as done in the Bayesian Inference, to evaluate each application separated and then to eliminate redundant states.

In Enhanced Dempster-Shafer Inference, each application will have its own set of hypothesis. In such approach, belief will represent the belief that a given application Y will have a determined behavior given the result of another application X. In this context H is the set of states, which represents the behavior of an application. From this belief function, we assume Equation 9:

$$DS1 \oplus DS2(H) = \frac{\sum_{X \cap Y = H} DS1(X)DS2(Y)}{1 - \sum_{X \cap Y = 0} DS1(X)DS2(Y)} \quad (9)$$

where DS1 and DS2 are Dempster-Shafer Inferences over conditions of isolated applications and the first line of the equation represents the belief that both applications will be in a given state simultaneously.

Considering X as the Overhead Power Line Monitoring application and Y as the BM application, the numerator of equation 8 represents the degree of doubt when DS1 infers that application X, under a given temperature condition, should alert a transmission failure. At the same time DS2 will indicate that, under the same temperature condition of DS1, application Y will turn off the battery, because there is no need of saving energy on a battery when there is no transmission. This is important because we will be able to prevent waste of resources through resuming repeated states of an application. The denominator of equation 8 represents the plausibility of this hypothesis.

5.3.3 Enhanced Moving Average Filter

As seen in chapter 2, the moving average filter is a widely used MDF in digital signal processing since it is computationally simple and capable to reduce signal noise. Although simple to understand and use, moving average filter deals only with measurements and does not deal with

higher semantic levels, such as decisions. Besides, in the Moving Average Filter technique, all the sensed data values are similarly weighted. In an environment with multiple applications, some values could be more important to a given application than to another. Using the applications presented in the section 5.2 temperature values above 80°C are more important to the battery monitoring application than to the Overhead power line-monitoring application. In the Enhanced Moving Average Filter, we need to evaluate a given measurement considering a target set of applications. In order to make this approach possible, we have modified the traditional equation of this technique to Equation 10.

$$x(k) = \frac{1}{N} \sum_{i=0}^{M-1} \mu z(k-i), \forall k \geq M, \mu > 0, N = \sum_{i=0}^{M-1} \mu \quad (10)$$

In Equation 10, M is the window size, $z = \{z(1), z(2), z(3) \dots\}$ are input data, $x = \{x(1), x(2), x(3) \dots\}$ are the data estimated by the method, μ is the weight given to a value based on its importance to an application, and N is the sum of all weights. An application specialist should choose the weight values appropriately.

Another variant of the WMA found in the literature, the Exponentially Weighted Moving Average (EWMA) weights the data of each sensor in order to better evaluate a given environment. Our approach is different from the EWMA since instead of assigning weights to the data of different sensors, we assume that all the sensors are capable of monitoring the same kind of information (temperature for instance), but the relevance of this information will change according to the requirements of each specific application, which will use it, since we are giving different weights for each application instead of each sensor.

5.3.4 Enhanced Fault Tolerant Averaging

The Fault Tolerant Averaging technique was first introduced by (MARZULLO 1990), in the context of time synchronization in distributed systems, as seen in chapter 2. The traditional Fault Tolerant Averaging technique considers the data that appears more frequently in order to estimate the valid intervals. For such technique, the data that appears less often is the result of a sensor misreading (Nakamura 2007). As the number of sensors deployed in a given region increases, the relevance of data in this region also increases. In the scenario of SSANs, we cannot guarantee that a measurement performed by a single sensor is wrong because the value collected is considerably distant from the values available from the other sensors, since we have many applications working simultaneously. While the thresholds in an OLPM application data range

(hypothetically) from 40°C to 65 °C, a BM application relies on temperature values from 40°C to 144 °C for performing its actions. Therefore, if a temperature sensor returns the value of 75 °C; such value cannot be promptly eliminated from all applications because it can indicate the presence of a transmission failure in the OLPM application context.

In order to enhance the FTA to work in the SSAN scenario, we propose the following course of action: let $I = \{I_1, \dots, I_n\}$ be the set of intervals $I_i = [x_i, y_i]$ provided by n applications (different from the sensors used in the traditional method) referring to samples of the same physical state variable taken at the same instant. For each application, we will evaluate the intervals (in this case, we will apply the traditional Fault Tolerant Interval technique). Then we will sum the resulting intervals from the Fault Tolerant Interval technique. The Enhanced Fault Tolerant Averaging computes $Mnf(I) = [low, high]$, where low is the smallest value in the application intervals I , and high is the largest value in the application intervals in I .

5.4 Experiments using the enhanced fusion methods

This section describes the experiments conducted with the enhanced MDFs in SSAN scenarios for evaluating the following properties: (i) the enhanced fusion methods accuracy compared to traditional ones; (ii) the impact of our enhanced fusion methods (its overhead) in terms of resource consumption (memory usage and energy consumption) on the WSN nodes. We performed the experiments both in a simulated setting and with real nodes. The experiment with real nodes was conducted to validate the results obtained by simulations by comparing them with the results obtained from a real WSN platform. In the following, we first introduce the environment configuration, the application scenario, and the metrics used in our experiment. Thereafter, we discuss the results of the experiments.

5.4.1 Environment Configuration

In this work, we considered a WSN composed of MICAz sensor nodes (endowed with 4kB of RAM, 128 kB of flash memory for program storage and 512 kB for data storage, powered by two AA batteries). The enhanced MDF algorithms were built by using TinyOS (LEVIS *et al.* 2005) development environment version 2.1.1, which is a popular, lightweight, open source operating

system for wireless sensor nodes, and nesC programming language, which is a programming language for networked embedded systems.

We considered that the SSAN is composed of 8 homogeneous sensor nodes (in terms of functionalities), 1 actuator node and 1 sink node. Each sensor node can play the role of either a fusion node or a collector node, where the collector nodes are responsible for collecting data and the fusion nodes are responsible for applying the enhanced MDF. A sensor node cannot act as both fusion node and collector node simultaneously. Sink node is used for managing all the sensor nodes within the system.

In the simulations we used the WSN simulator TOSSIM (LEVIS *et al.* 2003) since it is an accurate and scalable simulator of TinyOS applications. TOSSIM simulator presents a limitation of only working with a single deployment code image at a time, i.e., all the simulated sensor nodes use the same code during the simulation. To circumvent this restriction, as an alternative approach, we have created a single code containing the implementations of all the required components. All the simulated experiments sustain at least 1 hour and each one is repeated 30 times.

5.4.2 Application Scenario

We have chosen as application scenario, the transmission tower presented at the beginning of section 5.2, since it illustrates the need of both applications (Battery Monitoring and Overhead Power Line Monitoring) to share the same communication and sensing infrastructure (wireless sensor and actuator networks). An important point is that both applications are correlated, i. e., the behavior of an application can affect the behavior of another application. So, at 75°C there could be damages to the Overhead Power Line; if the line is damaged there is no energy supply through the overhead power lines. Consequently, there is no power to be stored by the battery even though 75°C represents a normal temperature for the BM application. Additionally, both applications share the same kind of information (temperature for instance), but with different data ranges. In our motivation scenario, the relevance of the temperature parameter changes according to the requirements of each specific application, which will use it. So, 75°C represents a normality situation to the BM application (the usual battery temperature is around 40°C – 144°C) but, the same temperature represents that the line could become damaged to the OLPM application (the usual and safe Overhead Power line temperature is around 40°C - 65°C). Finally,

in our motivation scenario the applications have different priorities. For the Smart Grid, it is more important the continuous energy supply through the overhead power line than the energy storage in the battery. So, the OLPM application must have a higher priority over the BM application. For these reasons, we have chosen this scenario to test our enhanced MDFs.

In order to evaluate the impact of dealing with multiple applications simultaneously, we have set time slots called T1, T2, T3 and T4. Each time slot represents a particular state of the applications along the time, a particular event to be detected. T1 represents ideal conditions for both applications (a safe condition, where there is no need to perform preventive actions to avoid damage). T2 represents an increase in the overhead power line temperature. T3 represents an increase in the battery temperature. T4 represents the occurrence of failures in both applications. During T1 the temperatures vary from 44°C to 65°C both in the Overhead power and in the battery. During T2, it varies from 65°C to 85°C in the Overhead power line. During T3, it varies between from 44°C to 65°C in the overhead power line and over 140 °C in the battery. Finally in T4 the temperature in the Battery is over 144°C and over 80 °C in the overhead power line.

We performed all experiments during 60 minutes. The fusion window and the interval to collect samples are set according to the most demanding application, in this case OPLM application. So, the Collector Nodes collected temperature samples at every 15 seconds and transmitted them to the Fusion Node; our fusion window is 4 messages. The Fusion Node performed the MDFs once per minute and sent the resulting fused data to the Sink Node as well as a control message to the Actuator Nodes. We have set that the initial air temperature, battery temperature and the overhead power line temperature respectively are 25°C, 44°C and 44°C, according to the thermal models presented in (GAL *ET AL.*, 2011), (HUNG *ET AL.*, 2010) and (SCHLAPFER & MANCARELLA, 2011). Both applications start at the same time.

5.4.3 Metrics

The following fundamental metrics were used in the performed experiments for evaluating accuracy of our enhanced MDFs compared to traditional ones were: False Positives (FP), False Negatives (FN), True Positive (TP), and True Negatives (TN). FP indicates when an application changes to a state and it should not have changed (an OPLM application that detects a transmission failure when there is no transmission failure) and FN indicates when an application does not change state and it should have changed (an OPLM application that does not detect a

transmission failure when the temperature reaches a given threshold). TP indicates that an application has changed states correctly (a transmission failure is detected) and TN indicates that an application has maintained its state correctly (an OPLM does not detect a transmission failure fire when there is no transmission failure). True occurrence is defined as the sum of TP and TN.

The metrics used in the performed experiments for evaluating resource consumption were: the memory consumption and the energy consumption. The memory consumption is defined as the amount of memory used by our enhanced MDFs installed in the nodes (RAM and ROM) and the energy consumption is defined as the amount of energy consumed by the network, i.e., the total amount of energy consumed by each node when executing our enhanced MDFs.

To evaluate the energy consumed by the execution of the MDFs we devised a simple energy model (WEI *et al.*, 2103). Sensor nodes in general have three functional modules (SILVA, VAZ DE MELO, ALMEIDA, & LOUREIRO, 2012): communication, sensing and processing. The state of a sensor node can be described by the combined states of all individual modules. We assume that each functional module has only two states, namely active and inactive. Each module can only stay in one of two states and its status does not affect other functional modules. In this work, the sensing and communication modules are the major functional parts we are concerned with. Thus, the energy consumption of a sensor node is a function of the node state and the time during which the node remains in that state. When the states are determined, a sensor node consumes a fixed energy rate in that time period. When the state of the corresponding functional module of a sensor node is active, the energy consumption is much greater than when it is inactive. Assuming that an insignificant amount of energy is consumed when the corresponding function module of a sensor node stays inactive, we consider that the energy consumption in that state is 0. Overall, the energy consumption of a sensor node during time t is calculated by $E(T) = E_c + E_s$, where E_c represents the energy consumption of communication module, and E_s represents the energy consumption of sensing module.

For the ease of analysis, we assume that the data exchange between two neighboring sensor nodes (within one-hop communication range) belonging to the same WSN is done through direct communication. The energy consumption of transmitting 1-bit data over distance d is defined as $E_{tx}(l,d)$:

$$E_{tx}(l, d) = E_{elec} \times l + \varepsilon_{amp} \times l \times d^2 \quad (11)$$

where E_{elec} and ε_{amp} are hardware-related parameters (MAHAJAN & MALHOTRA, 2011). We also assume that the receiver does not consume energy in the data exchange process. For any two-distance sensors (outside one-hop communication range, still belong to the same WSN), the data communication is transferred by using shortest path based multihop routing protocol (please note that the routing process is out of scope of our work). The energy consumption of transmitting l-bit data from source to the destination is defined as $E_{tx}(l, src, des)$:

$$E_{tx}(l, src, des) = \sum_{i=1}^k E_{tx}(l, d) \quad (12)$$

where k is the minimum hop count the data travels from source to destination and $E_{tx}(l, d)$ remains the same meaning as shown in equation (11). The energy consumption of sensing module is calculated by a linear equation:

$$E_{s_i} = ER_{s_i} \times t_{s_i} \quad (13)$$

where ER_{s_i} represents energy consumption of service i in one time unit and t_{s_i} represents the time period for performing service i.

5.4.4 Evaluating the Accuracy of the enhanced MDFs

This section describes the experiments conducted with the enhanced MDFs in our motivating scenarios for evaluating the accuracy of the enhanced data fusion methods compared to traditional ones in terms of the ability of the MDF (the enhanced or the traditional) to correctly identify a potential damage in the structures of the applications in our motivating scenarios. Three experiments with four time slots (T1, T2, T3 e T4) were made. In the first, we evaluate the enhanced MDFs Behaviour, when we evaluate the enhanced MDFs, we apply such MDFs to both applications simultaneously. In the second, we evaluate the traditional MDFs also to both applications simultaneously. Finally in the third we evaluated each application alone using traditional MDF, we apply such MDFs to each application individually. So, for each time slot,

each one of the proposed enhanced MDFs and each one of the traditional MDFs were tested in order to compare the enhanced and traditional MDFs results in terms of true occurrence (TP + TN).

In these experiments, we expect that: (i) in T1, no application alerts a FP or FN; (ii) in T2, only the overhead power line monitoring application becomes prone to generating FP; (iii) in T3, the battery monitoring application alerts a FN; and (iv) T4, both applications alert FP and FN.

A - EVALUATING THE ENHANCED MDFS BEHAVIOUR IN T1

Figures 11, 12 and 13, show the achieved results for T1. For each MDFs (the Bayesian Inference (BI), the Dempster-Shafer Inference (DS), the Fault Tolerant Interval (FTI) and the Moving Average Filter (MAF)), the columns in Figures (11, 12 and 13) represent the following metrics: True Positive (TP), True Negative (TN), False Positives (FP), False Negatives (FN). The lines in Figures (11, 12 and 13) represent BM and OLPM applications. The word “Both” in these Figures indicate that MDFs are applied to the both applications simultaneously.

Figure 11 represents how accurate are OLPM and BM applications in detecting the occurrence of events using traditional MDFs (applied to each application individually) in terms of true occurrences (TP + TN). In this Figure, the traditional MDFs are applied to a single application at a time (first BM and then OPLM application). Figure 12 represents how accurate are OLPM and BM applications using traditional MDFs in terms of true occurrences. In this Figure, the traditional MDFs are applied to the applications simultaneously. Figure 13 represents how effective in detecting the true occurrence of events are OLPM and BM application using enhanced MDFs in terms of true occurrences. In this Figure, the enhanced MDFs are applied to the applications simultaneously.

In the case where both applications are in an ideal condition (T1), we observe, in Figure 13, that the results obtained by our Enhanced MDFs were better than those obtained by the traditional MDFs (Figure 12) in terms of true occurrences (TP + TN). The amount of true occurrence for the traditional BI, DS, FT and MA techniques were 72,3%, 75,6%, 72,1% and 72,1% respectively. The amount of true occurrence obtained by the enhanced BI, DS, FT and MA techniques were 88,1%, 90,1%, 89,3% and 89,2% respectively. The amount of true occurrence obtained by the

enhanced MDFs techniques were much higher than those obtained by the traditional MDFs, (as presented in Figure 12).

The made comparison between the enhanced and the traditional MDFs showed interesting results. In state T1 (Figures 11, 12 and 13), we can observe that the enhanced MDFs do not reach 100% of true occurrence. The reason is that the data range of the battery monitoring application is larger than the one of the OPLM application. A temperature of 100°C, which could be considered normal to the BM application, is considered an indicative of failure to the OPLM application. If the network gathers a large number of samples in a data range specific to a single application, this could lead to a misinterpretation of the real situation of the environment. When we apply a traditional MDF, it manipulates all the data equally, what leads to an error. The error result occurs when an application cannot detect a failure (FN), or it detects a failure when such failure does not exist (FP). We concluded that Traditional MDFs are more susceptible to FPs than Enhanced MDFs, since the number of FP occurrences is greater than FN.

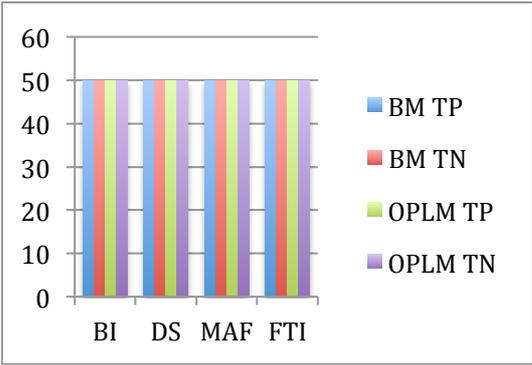


Figure 11 MDFs Behaviour in T1 for each application

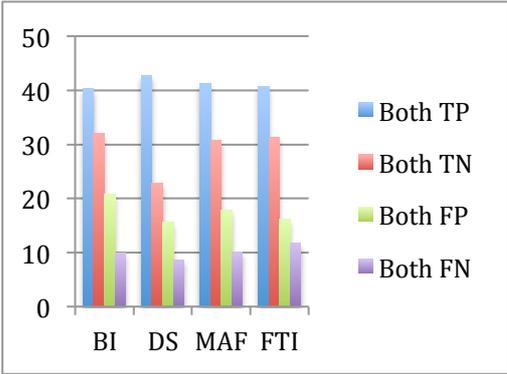


Figure 12 Traditional MDFs Behaviour in T1 applied to both applications simultaneously

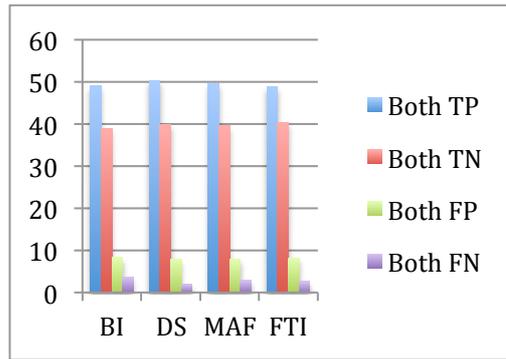


Figure 13 Enhanced MDFs Behaviour in T1 applied to both applications simultaneously

B -EVALUATING THE ENHANCED MDFS BEHAVIOUR IN T2

Figures 14, 15 and 16 show the achieved results for T2. For each MDFs (the Bayesian Inference (BI), the Dempster-Shafer Inference (DS), the Fault Tolerant Interval (FTI) and the Moving Average Filter (MAF)), the columns in Figures (14, 15 and 16) represent the following metrics: True Positive (TP), True Negative (TN), False Positives (FP), False Negatives (FN). The lines in Figures (14, 15 and 16) represent BM and OLPM applications. The word “Both” in these Figures indicate that MDFs are applied to the both applications simultaneously.

Figure 14 represents how accurate are OLPM and BM applications in detecting the occurrence of events using traditional MDFs (applied to each application individually) in terms of true occurrences (TP + TN). In this Figure, the traditional MDFs are applied to a single application at a time (first BM and then OPLM applications). Figure 15 represents how effective are OLPM and BM applications using traditional MDFs in terms of true occurrences. In this Figure, the traditional MDFs are applied to the applications simultaneously. Figure 16 represents how effective are OLPM and BM applications using enhanced MDFs in terms of true occurrences. In this Figure, the enhanced MDFs are applied to the applications simultaneously.

In the case where only the OPLM application alerts a failure (T2), it was observed that the results obtained by our enhanced MDFs were better than those obtained by the traditional MDFs (Figure 15) in terms of true occurrences (TP + TN). The amount of true occurrences obtained by the enhanced BI, DS, MA and FT techniques were 89,3%, 90,2%, 90,4% and 88,6%, respectively and the amount of true occurrences obtained by the traditional BI, DS, MA and FT techniques were 78,3%, 79,9%, 77,4,1% and 76,4%., respectively. Then, we can note that the amount of true

occurrences obtained by the enhanced techniques was much higher than those obtained by the traditional MDFs (Figure 15), because the data range of the OLPM application is within the data range of the BM application. So in these experiments we had excellent results.

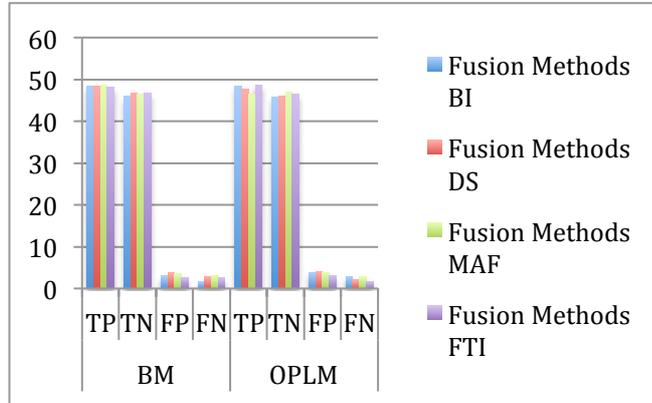


Figure 14 MDFs Behaviour in T2 for each application

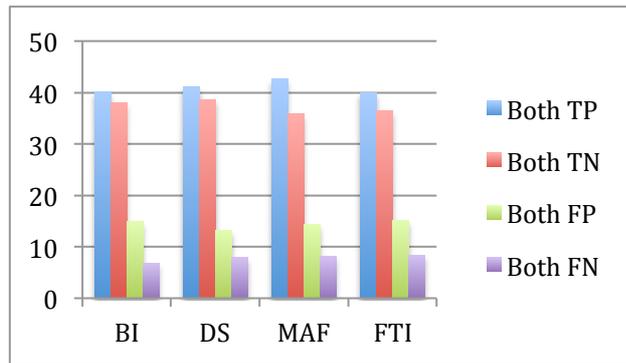


Figure 15 Traditional MDFs Behaviour in T2 applied to both applications simultaneously

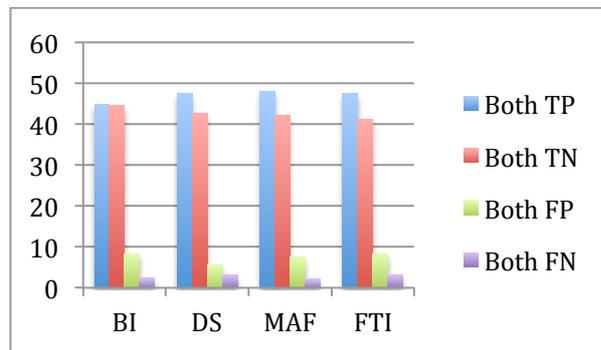


Figure 16 Enhanced MDFs Behaviour in T2 applied to both applications simultaneously

C - EVALUATING THE ENHANCED MDFS BEHAVIOUR IN T3

Figures 17, 18 and 19 show the achieved results for T3. For each MDFs (the Bayesian Inference (BI), the Dempster-Shafer Inference (DS), the Fault Tolerant Interval (FTI) and the Moving Average Filter (MAF)), the columns in Figures (17, 18 and 19) represent the following metrics: True Positive (TP), True Negative (TN), False Positives (FP), False Negatives (FN). The lines in Figures (17, 18 and 19) represent BM and OLPM applications. The word “Both” in these Figures indicate that MDFs are applied to the both applications simultaneously.

Figure 17 represents how accurate are OLPM and BM applications in detecting the occurrence of events using traditional MDFs (applied to each application individually) in terms of true occurrences (TP + TN). In this Figure, the traditional MDFs are applied to a single application at a time (first BM and then OPLM application). Figure 18 represents how effective are OLPM and BM application using traditional MDFs in terms of true occurrences. In this Figure, the traditional MDFs are applied to the applications simultaneously. Figure 19 represents how effective are OLPM and BM application using enhanced MDFs in terms of true occurrences. In this Figure, the enhanced MDFs are applied to the applications simultaneously.

In the case where only the battery monitoring application alerts a change (T3), we can observe in Figure 18 that the values achieved by the enhanced BI, DS, MA and FT techniques in terms of true occurrences (TP + TN) were 72%, 73,8%, 74,5% and 91,8%, respectively, and the values obtained by the traditional BI, DS, MA and FT techniques were 49%, 52,4%, 44% and 54%, respectively. Therefore, the amount of true occurrence obtained by our enhanced MDFs was better than those obtained by the traditional MDFs (Figure 18). However, our enhanced MDFs had results worse in T3 than those found in T1 and T2. This happens due to the fact that the data range of the OPLM application, which did not present change in T3, is not within the data range of the BM application.

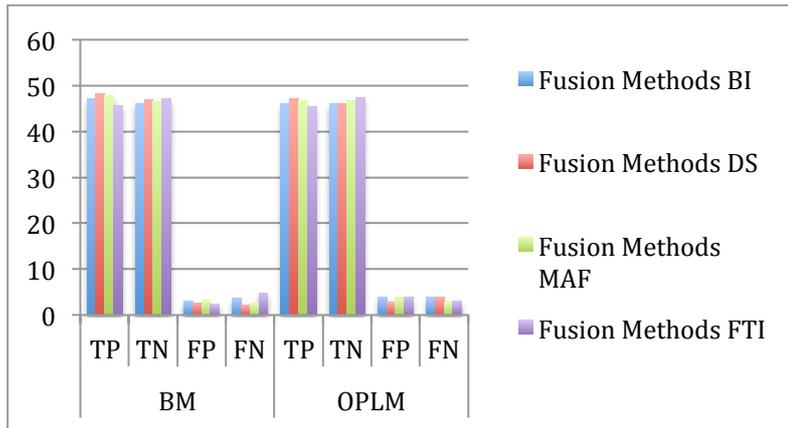


Figure 17 MDFs Behaviour in T3 for each application

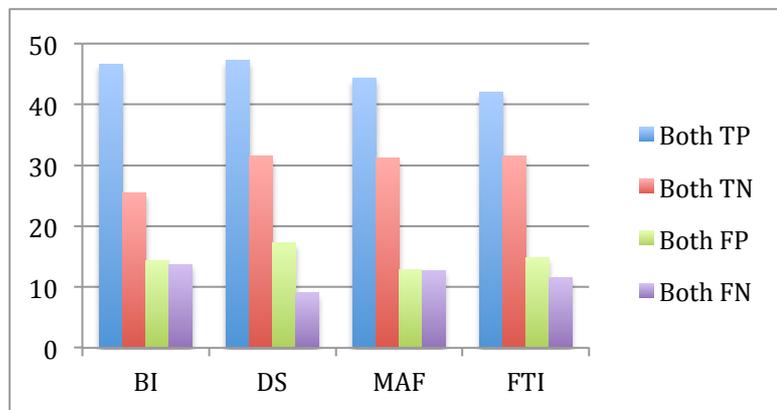


Figure 18 Traditional MDFs Behaviour in T3 applied to both applications simultaneously

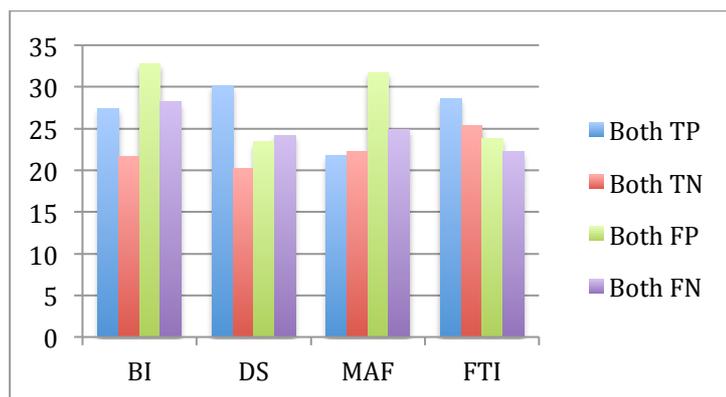


Figure 19 Enhanced MDFs Behaviour in T3 applied to both applications simultaneously

D - MDFS BEHAVIOUR IN T4

Figures 20, 21 and 22 present the achieved results for T4. For each MDFs (the Bayesian Inference (BI), the Dempster-Shafer Inference (DS), the Fault Tolerant Interval (FTI) and the Moving Average Filter (MAF)), the columns in Figures (20, 21 and 22) represent the following metrics: True Positive (TP), True Negative (TN), False Positives (FP), False Negatives (FN). The lines in Figures (20, 21 and 22) represent BM and OLPM applications. The word “Both” in these Figures indicate that MDFs are applied to the both applications simultaneously.

Figure 20 represents how accurate are OLPM and BM applications in detecting the occurrence of events using traditional MDFs (applied to each application individually) in terms of true occurrences (TP + TN). In this Figure, the traditional MDFs are applied to a single application at a time (first BM and then OPLM applications). Figure 21 represents how effective are OLPM and BM applications using traditional MDFs. In this Figure, the traditional MDFs are applied to the applications simultaneously. Figure 22 represents how effective are OLPM and BM applications using enhanced MDFs in terms of true occurrences. In this Figure, the enhanced MDFs are applied to the applications simultaneously.

In the case where both applications alert a change (T4), we observed in Figure 22 that the results obtained by our enhanced MDFs were better than those obtained by the traditional MDFs in terms of true occurrences (TP + TN). The amount of true occurrences obtained by our enhanced MDFs was higher than the ones obtained by the traditional MDFs because the states of both applications were representing changes. Due to this fact there was almost no misinterpretation of the data by the enhanced MDFs. On the other hand, it was observed that the traditional MDFs (Figure 21) did not achieve good results, because such methods cannot handle distinct data ranges properly. The amount of true occurrences obtained by the enhanced BI, DS, MA and FT methods were 92%, 93,4%, 90,8% and 91,8%, respectively.

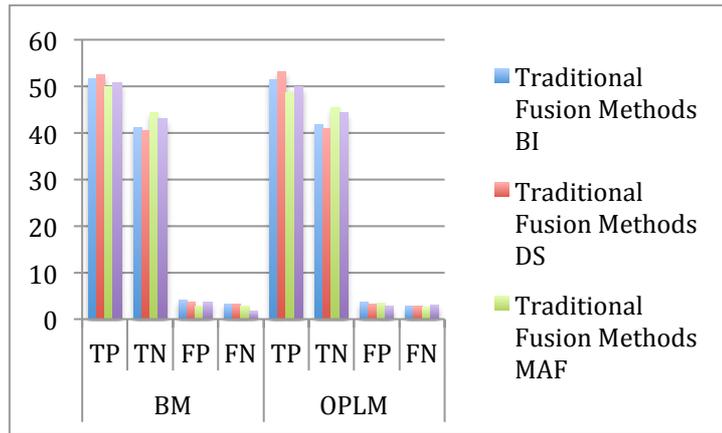


Figure 20 MDFs Behaviour in T4

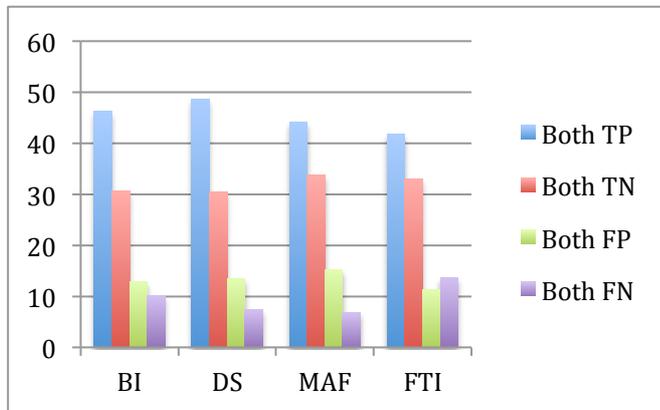


Figure 21 Traditional MDFs Behaviour in T4 applied to both applications simultaneously

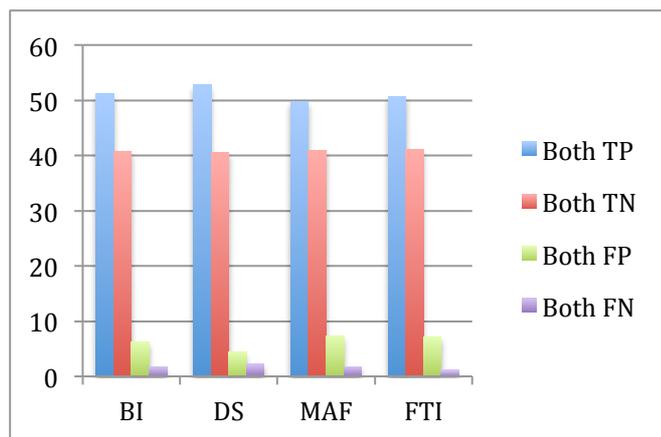


Figure 22 Enhanced MDFs Behaviour in T4 applied to both applications simultaneously

5.4.5 Experiment to Evaluate the Resource Consumption of WSN

This section describes the experiments conducted with the enhanced MDFs in our motivating scenario for investigating the impact of our enhanced fusion methods (its overhead) in terms of resource consumption (memory usage and energy consumption) on the WSN nodes.

Regarding to memory usage values (a total of 4 kB RAM and 128Kbytes of ROM were available in MicaZ nodes) for each role, the Collector Node uses 2.8% (116 bytes) of RAM and 0.8% (1054 bytes) of program memory and the Fusion Node uses 2.4% (99 bytes) of RAM and 2.3% (2956 bytes) of memory footprint. In short, in MicaZ sensor platform the proposed enhanced MDFs requires around 3% of total RAM memory (2% more than without the MDFs) and around 5% of total ROM memory. It may be noted that none of the Collector and Fusion nodes exceeds the 4 kB RAM available. Nevertheless, we can conclude that the proposed enhanced MDFs (Fusion System) is feasible to be employed in a real WSN deployment.

Regarding to energy consumption, each Collector Node consumes 1.614.938 mJ and each Fusion Node consumes 149.542 mJ in case of adopting the traditional MDFs. On the other hand, by adopting the proposed enhanced MDFs, each Collector Node consumes 801.792 mJ and each Fusion node consumes 743.155 mJ. This happens because the traditional MDFs (Fusion System) performs the MDF once for each application and sends the result to a Sink Node and the proposed enhanced MDFs performs the MDF only once and send the result to a Sink Node. So, the energy consumption using the traditional approach was 100% higher than using our enhanced MDFs.

5.4.6 Real node experiments

In this section, the same scenario simulated using TOSSIM was implemented on a real sensor node platform. We aimed to validate the results obtained from simulations by comparing them with the results obtained from a real WSN platform. Our goal was to confirm that the results obtained from simulations actually reproduce the results that would be returned if all experiments were performed on a real WSN platform.

This real experiment was performed in a controlled environment (our research laboratory at UFRJ). In this case, the nodes were kept stationary and disposed on the floor.

Regarding the energy consumption, the obtained average results in the experiments with a real sensor node for collector node (788964 mJ) and for the fusion node (719849 mJ) can be considered less than those in the simulated experiment for the collector node (801792 mJ) and for the fusion node (7431552 mJ). Considering the confidence interval of 95%, the observed differences between the obtained results in the real and simulated experiments were due to the noise simulation component of TOSSIM does not considerer of the same way the radio interference existent in the a real WSN platform. So, the presence of interference on the real experiment induced some packet loss that does not occur in the noise simulation component resulting in packet loss in real experiment. Therefore, these results validate the experiments performed in the simulated environment.

Considering the delay, the obtained average results in the experiments with a real sensor node can be considered similar to those in the simulated experiment. The obtained average delay result in the simulated experiment was slightly higher than the real one, which were 14,562 and 12,739 ms, respectively.

Considering the accuracy of event detection in terms of true occurrence (TP+ TN), the simulated experiment had achieved slightly better results than the real one, as shown in Figure 23. In Figure 23 each color represents one of the time slots and each column represents the difference between simulated and real experiments results of the accuracy. Again, the observed differences between the obtained results in the real and simulated experiments were due to the noise simulation component of TOSSIM does not considerer of the same way the radio interference existent in a real WSN platform. So, the presence of interference on the real experiment induced some packet loss that does not occur in the noise simulation component. Figure 16 shows that in the slot time T2, there is the greatest variance among the obtained results in the real and simulated experiments in relation the accuracy. This happens due to the fact that the interval of disturbed application (OPLM) is included in the interval of the non-disturbed application (Battery monitoring) as shown in figures 7, 8, and 9. In other words, an increase in the overhead power line temperature would cause problems to the overhead power line application represented by an increasing number of FP.

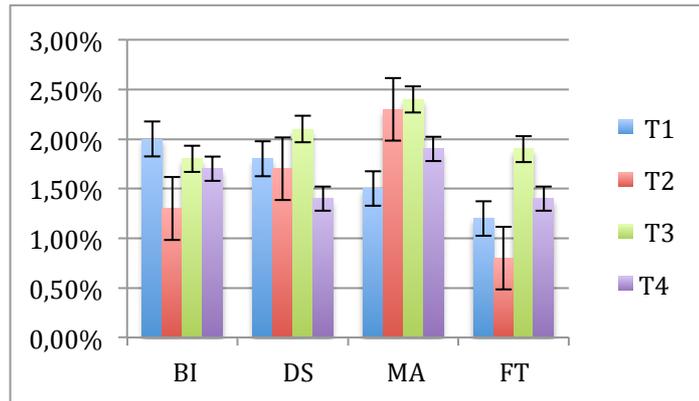


Figure 23 Difference among the results in the real and simulated experiments in relation the accuracy.

5.5 Conclusions

This chapter presented the enhanced MDFs for SSANs. Such MDFs augment traditional MDFs to work in the SSAN environment by exploring application similarities on data thresholds. Several experiments were conducted to prove the accuracy of the proposed MDFs in terms of true occurrence. The results of such experiments, the enhanced MDFs, in comparison to the traditional MDFs, reduced the energy consumption and were more accurated. The memory resources consumed by the different roles played by the nodes in our architecture were also assessed.

We can conclude that the data range and the weights assigned to applications (representing the relative priority assigned of each application) are extremely important in the fusion process. Since applications have different degrees of importance, it is intuitive to assume that their data have different degrees of importance. If an application less relevant but with a large data range has the same degree of importance of the other existing applications, it will lead to an error, i.e., a result which does not mirror the current situation of the environment. If we consider the data semantics, and weight it according to its importance, the MDF will return a result closer to reality.

The combination of different MDFs has the potential to lead to better results. Using different MDFs can lead to better results (in terms of energy consumption, FP, FN, TP, TN). One example is the adoption of the Dempster-Shafer Inference method after applying a Fault Tolerant Interval technique. Since the Fault Tolerant Interval technique shapes a certain amount of data, thus providing a better set of samples to be used by the Dempster-Shafer Inference.

As a future work, we intend to investigate how to combine different MDFs, looking for combinations that show better results (in terms of energy consumption, FP, FN, TP, TN, sensibility and specificity). Also, designing a framework capable of deciding what MDF to apply, given a certain amount of data available is definitely a future direction of great importance. Through some strategies, such as computational intelligence, finding the most appropriate MDF is useful in environments where applications change quickly, such as the SSAN scenario.

6 A Scheduling algorithm for shared sensor and actuator networks

The content of this Chapter presents a SSAN scheduling algorithm tailored for a single network. The algorithm combines the DAGs (Directed acyclic graphs that represent applications) with common tasks into a new DAG without damaging the integrity of each task graph. Once the DAGs are combined, the applications are then allocated to the SSAN based on DAG's composition (XIONG *et al* 2011). The composited DAG could contain periodic applications or aperiodic applications during its execution. This algorithm was presented at International Conference on Information Networking -ICOIN 2013 and it is entitled “A scheduling Algorithm for Shared Sensor and Actuator Networks” .

The rest of this Chapter is organized as follows. Section 6.1 presents an introduction. Section 6.2 introduces the models used in our work. In section 6.3, we present our designed scheduling algorithm for SSANs. In Section 6.4, we describe the experiments to evaluate the proposal. Section 6.5 concludes this paper and outlines future work.

6.1 Introduction

As Wireless Sensor and Actuator Networks (WSANs) design starts shifting from application-specific platforms to shared system infrastructures, a new but pressing research challenge is how to allocate limited node resources to contending applications running on such system. This issue has received notable attention from both academic and industry sides. One possible solution is to employ task scheduling algorithms, which are responsible for energy conservation as well as application managements. By tasks, we mean the non-dividable units of execution that consist of an application submitted by users. Although several efforts have been made, most of existing works concern about the case of running an application in single WSN with simple objective to achieve energy efficiency. Task scheduling algorithms (XIONG *et al.* 2011) designed for SSANs should not only pay close attention to the energy saving by altering functional parts of sensor nodes in their lifetime, but should also fully utilize the common tasks from different applications so as to further improve the use of limited precious node resources. This is practical due to the fact that tasks from multiple applications could run only once and the collected result is able to be shared by all of them. And the common tasks, if not properly addressed by the scheduling

algorithm, will consume system energy in a less efficient way by repeatedly performing them. This situation is normally to be appeared in periodic applications, e.g. a fire detection application and a heating application could share not only a temperature sensing task but also a temperature threshold evaluation task. Besides that, aperiodic applications with short execution length in the SSAN also have chance to experience the similar situation, e.g. meeting detection application in (WANG *et al.* 2004). The meeting detection application requires a presence sensor and a temperature sensor to analyze if the attendees are comfortable during a scheduled meeting (if the room's temperature is satisfactory in relation to the attendance). A fire detection application and a meeting application can share a temperature sensing task and a temperature threshold evaluation task. Moreover, the aperiodic application could share common tasks with periodic applications in the system; energy efficiency can be further improved by exploiting commonality of the application interests. For instance, the meeting detection application and the fire detection application have the same interest to collect data from the temperature sensor for their completion.

To address the aforementioned issues, this chapter presents an energy-efficient scheduling algorithm to perform multiple applications in SSANs. We modeled applications as DAGs (Directed Acyclic Graphs) where each node of a DAG is a task and the edges represent the dependencies between tasks. To maintain task dependencies, we combine the DAGs with common tasks into a new DAG without damage the integrity of each task graph. Once the DAGs are combined, the applications are then allocated to the SSAN based on DAG's composition (XIONG *et al.* 2011). The composited DAG could contain periodic applications or aperiodic applications during its execution.

6.2 Models and problem definition

6.2.1 System Model

Our approach is built upon the design of SenShare (LEONTIADIS *et al.* 2012), an architecture for SSAN that allows the creation of overlay WSNs on top of a physical shared infrastructure. In an overlay network each sensor node allows multiple co-located applications to use its hardware resource (we could have used another Virtual Network approach but Senshare was the only one specific to the SSAN scenario). An application can be distributed over the whole infrastructure or a selected sub-set of the physical network. SenShare provides a clear separation between the system infrastructure and the individual applications, allowing ownership and management of the

infrastructure and applications by different authorities. SenShare's goal is to map applications into physical nodes to operate in the virtual environment built on the real sensor network. From the application point of view, it is assigned to run in an overlay WSN, while in the context of the SSAN, the application is allocated to a subset of selected nodes. The overlay WSNs may share real nodes in the SSAN. In this way, an application can span across the whole infrastructure or a selected sub-set of the physical network.

By adopting Senshare as the infrastructure for our study, the sink node is responsible for the formation of the overlay wireless sensor networks. Each application in our model represents an overlay WSN in SenShare. By using SenShare, we ensure that all the intermediate results of application can freely travel the nodes that form the overlay network and reach the sink node. The sink node uses the *Network I/O* component presented in (LEONTIADIS *et al.* 2012) and our system encompasses one SSAN. The SSAN considered in this work includes a sink node and a set of heterogeneous sensor and actuator nodes that can belong to multiple physical networks. The sink node is responsible for the overlay network formation as well as executing the proposed scheduling algorithm. The sink acts as a façade to the entire system responsible for handling the dynamic arrival of applications, decomposing an application into tasks, determining the specific nodes required for processing the application, deciding how to dispatch the tasks to the selected nodes and gathering the latest information about the network execution context and sensor nodes status. The information includes, for each sensor node, its sensing capabilities, current residual energy and operation mode and geographical location. Such information is acquired by messages sent by sensor nodes whenever required from the sink (normally at the time of a successful completion of the allocated tasks of an application).

Any given SSAN is modeled as an undirected graph $G = (V, E)$, where $V = (v_1, v_2, \dots, v_n)$ represents the set of sensor nodes and $E = (e_1, e_2, \dots, e_m)$ represents the set of all possible communication links among the sensor nodes in the same WSN. We assume that there is a SSAN G , formed by a group of overlay WSNs $= \{G_1, \dots, G_n\}$, deployed into the monitored area W so that all interested events of the applications can be detected by at least one sensor. All overlay networks in G have a routing path to the sink node SN .

Any sensor node can overlap the monitoring region of a set of other sensor nodes. This indicates that some regions of the monitored area W are mutually covered by multiple sensors

that can perform a certain task. For any given sensor node V_i in V , i denotes the index of the sensor nodes that belongs to the SSAN. A sensor node V_i is capable of providing one or more tasks depending on their capabilities to collect/sample different types of data, for instance, temperature, light, smoke, and movement. All sensors are endowed with the same radio interface as well as communication and sensing ranges. Moreover, sensors can detect all events of interest occurred within their sensing range. We assume that all the sensors in the SSAN have a valid communication path to reach the SN . This is guaranteed by the SenShare infrastructure that manages the formation of overlay networks as presented in (LEONTIADIS *et al.* 2012). Communication interference between sensors is not considered in this section. In order to examine the availability of a sensor, we introduce an availability function that indicates whether a sensor can provide the required service at the specified area. The function is shown below:

$$A(t, x, y, i) = \begin{cases} 1, & \text{if a sensor is available} \\ 0, & \text{if a sensor is unavailable} \end{cases} \quad (14)$$

where t is the task that an application requires, x and y are the geographical location for the monitoring event and i is the index of the sensor. In order to evaluate the node energy consumption, we adopted the energy consumption model used in (TIAN, EKICI and OZGUNER 2005).

6.2.2 Application Model

We represent applications as Directed Acyclic Graphs (DAG) (XIONG *et al.* 2011). A DAG $T = (N, A)$ consists of a set of vertices N representing the tasks to be executed and a set of directed edges A representing dependencies among tasks. The edge set A contains directed edges a_{ij} for each task n_i which is the parent node of the task n_j . In DAG, a task without parent tasks is called entry-task and a task without child tasks is called exit-task. We stipulate all entry tasks of a DAG are sensing tasks, meaning the tasks that represent the different sensing functions requested from the users. The communication cost is only counted when two adjacent non-communication tasks are allocated to different sensor nodes, otherwise the cost is ignored.

6.3 The Proposed algorithm

In this section, we present a task scheduling algorithm tailored for SSAN. In the SSAN scenario, multiple applications could have common tasks. Our algorithm exploits the commonality in tasks to achieve energy efficiency.

As we mentioned before, there are multiple applications running in SSAN, an obvious way to complete these applications is to schedule them one by one. The immediate issue with this approach is that it might leave the nodes in idle status for a long time and increase the overall end-to-end delay of applications, since each task is non-preemptive, which means when the task starts execution and it cannot be terminated until it is completed (PASCHOALINO and MADEIRA 2012). Another potential issue with this means is to produce more intermediate result exchange between nodes, thus reducing the network lifetime. In other words, such approach does not take advantage of the fact that applications may share common tasks. An alternative approach (XIONG *et al.* 2011) is to schedule several DAGs at the same time, transforming the selected DAGs into a single DAG. The approach proposed in (XIONG *et al.* 2011) creates a composite graph by making: (i) the entry nodes of all DAGs as immediate successors nodes of a single dummy entry node and (ii) the exit nodes of all DAGs immediate as ancestors' nodes of a single dummy exit node. These two extra dummy nodes are associated with zero computation and communication cost. Besides that, we also need to combine the DAGs with common tasks into a single DAG and distribute the composite DAG to appropriate sensor nodes. The pseudo code of the algorithm is given in Figure 24:

<p>Input: A set of applications (DAGs), system topology, application requirements</p> <p>Output: The task scheduling scheme</p>
<ol style="list-style-type: none"> 1. //Formation of a new DAG 2. Define a Final DAG (FD) composed of an Entry Node (EN) and a Final Node (FN). 3. When a new DAG A_i arrives, decomposes the application A_i into n tasks, where $\mathbf{N} = \{n_1, n_2, \dots, n_n\}$. 4. For each task v_i in V do 5. For a given task $n_i \in \mathbf{N}$, verify if this task is in FD. 6. If v_i is in FD, the ancestor of n_i in A_i will point to n_i in FD and n_i in FD will point to the successor of n_i in A_i 7. If the entry node of A_i has no ancestor, it becomes EN and if the final node of A_i has no successor it becomes FN. 8. Put all the tasks of FD into a queue Q //Sensor availability 9. While size(Q) > 0 do 10. For a given task $n_i \in \mathbf{N}$, check the sensor availability of sensors by using equation (1) 11. Create a candidate sensors list CS_i and put the sensors with value 1 into the list 12. Remove the task n_i from queue Q

```

13. End While
14. //Scheduling
15. For each candidate sensors list  $CS_i$ 
16. For each  $v_j \in CS_i$  do
17. select the sensor with the closest value to the result generated
    by equation (2)
18. End If
19. End For
20. Return the optimal sensor  $v_j$  for  $n_i$ 
21. End For
22. The latest information of nodes which perform the allocated
    tasks of application  $A_i$  is collected by their sink nodes and
    send back to application planner for next arrival application

```

Figure 24 pseudo code of proposed algorithm

In the following subsections we will provide a detailed explanation of our algorithm. It encompasses 3 steps: (i) new DAG formation, (ii) sensor availability evaluation and (iii) scheduling. In the first step, the algorithm identifies and eliminates common tasks. Secondly, the nodes that can perform the required tasks are selected and in the last step tasks are assigned to the chosen sensors.

6.3.1 New DAG Formation

When a new application arrives, the sink node will analyze the composition of the application. First of all, the proposed algorithm will add the new arrival application into the existing DAG. The new arrival application is placed between the dummy entry node and dummy exit node. Then, it will search in the new arrival application for tasks are common to the applications that are currently running in the system. Using the precedence, our algorithm excludes the common tasks (in this case represented as nodes of the DAG). The ancestor of the common node (task presented in more than one application) on the arriving application will point to the node that is already in the DAG and the node in the tree will point to the successor in the arriving application.

6.3.2 Sensor Availability

For each task, all possible sensor nodes capable of executing the required task are identified. Due to the hardware limitation of sensor node, each sensor node is constrained by its sensing range and only the physical phenomenon within this range can be detected. As a result, those sensors

are filtered; by considering locations of those available sensors, their sensing ranges and the location of requested task, and filtered sensors are put into a candidate sensor list.

6.3.3 Scheduling

As the performance of the proposed algorithm relies on task sharing to reduce energy consumption, we derive a formula to represent energy consumption on a node, and quantitatively select the best candidate from a candidate sensor list. The normalization technique is employed to generate equation (15).

$$E_{candidate} = \frac{E_{residual} - E_{s_i}}{E_{residual}} \quad (15)$$

For a given task, $E_{residual}$ represents the residual energy of the SSAN. E_{s_i} denotes the energy consumption value for any sensor in the primary sensor list and it ranges from E_{best} to ∞ . We use normalization because sensor may have different batteries, and we are interested in how a sensor is positioned in relation to the network.

6.4 Experiments

To evaluate the proposed algorithm performance under a realistic scenario, we implemented the algorithm in real sensor nodes and run a set of tests under the context tailored for SSANs. The goal of the tests is to evaluate the concerned system performance metrics: system lifetime and scheduling success rate.

The first experiment aimed at assessing the system lifetime, evaluating the energy consumption of our algorithm compared to a **naive approach**, a scheme for task scheduling that was created as a baseline. In such scheme, each task is allocated in a random way that considers if the node is able to execute the selected task (in terms of geographical position, required sensing interfaces and amount of energy to conclude the task). If the node randomly selected is not able to perform the task, another node will be randomly chosen. In the second experiment, we evaluated the influence of packet loss on the system lifetime. Packet loss is important since it allows us to analyze how resilient the proposed algorithm is. In the third experiment, we evaluated the scheduling success rate to determine if our algorithm is able to, besides saving energy, properly allocate tasks to meet the application requirements. Finally, we compare the performance of our algorithm with the algorithm proposed in (XU *et al.* 2010).

6.4.1 Experimental Settings

The tests were conducted in the SUN SPOT platform (WILDE, GUINARD and TRIFA 2010), a sensor platform particularly suitable for rapid prototyping of WSNs applications. The SUN SPOT SDK environment includes Solarium, that contains a SPOT emulator that is useful for testing SPOT software and/or to create scenarios with a large number of nodes whenever the real hardware is not available. In the performed tests, we created a mix of real nodes with emulated SPOTS by using Solarium so that we can assess the system scalability.

6.4.2 Parameter Setup

In our tests, there are several parameters that need to be properly specified considering the application, system and energy models. We assume that applications arrive to the SSAN following a Poisson distribution with a mean of 10 time units. At each arrival time, it is likely that more than one application arrive, and this probability is controlled by a parameter called *multiple application arrival rate* (LI, DELICATO and ZOMAYA 2013) that ranges from 0 to 1 with the default value set to 0.1. The larger the value of multiple application arrival rates is the higher is the possibility of multiple applications to occur at each arrival time. The maximum number of applications arriving to the SSAN at the same time is set to 3. Each arriving application contains 2 to 6 independent tasks, similarly to the procedure used in (XIONG *et al.* 2011). This procedure of random graphs generation was further explained in (XIONG *et al.* 2011). It is discussed in the literature that random graphs may not always represent real applications; however, the diversity they provide is sufficient for this group of experiments as explained in (XIONG *et al.* 2011). A sensor field was created with 200 nodes randomly distributed in a square area with 200m x 200 m. A single sink node was placed in the top right corner of the field. We used LQRP (PASCHOALINO and MADEIRA 2007) as routing protocol to transmit data through nodes. Unless specifically stated, all the aforementioned parameter values will be used as default setting in the experiments. All our tests compare the results of the proposed algorithm with a benchmark, the **naïve approach** previously described. The work of (XU *et al.* 2010) will be used in some tests and also compared with the **naïve approach**. We initially assumed that all transmissions were carried out without data loss. Further we introduced a data loss model in the simulations, in order to evaluate the impact of packet loss on the performance of the proposed algorithm. Each simulation runs for 900s at the end of which the

network residual energy was obtained. All the presented results were averaged over at least 300 times; consequently, a 95% confidence interval with a 5% (or better) precision was achieved.

6.4.3 Performance Metrics

In our experiments, the **system lifetime** and **SSR** are used as the concerned performance metrics for the evaluation. System lifetime is probably the most important metric for the SSANs performance evaluation. In this algorithm, the main goal of the SSAN is to appropriately schedule tasks to save energy so as to extend the system lifetime. However, the energy consumption metric does not fully demonstrate the superior performance of our algorithm. We also adopt two other metrics used in (ZHU *et al.* 2012): (i) *number of executed tasks* and (ii) *number of missed tasks*. The number of executed tasks is used to measure the total number of tasks is successfully executed in the experiment. The number of missed tasks is used to measure the total number of tasks is failed to execute due to various issues occurred in the system, such as packet loss, sensor malfunctioning, loss of connectivity and node's energy depletion (XIONG *et al.* 2011). For example, due to insufficient energy, a given task n_i may only be scheduled or executed for a reduced number of times within the scheduling interval. Based on these two metrics, we derive **Scheduling Success Rate (SSR)**, which is known as *number of executed tasks / (number of executed tasks + number of missed tasks)*.

6.4.4 System lifetime

The main goal of this test is to assess how long the SSAN would last under the given number of applications. Results shown in Figure 25 indicate that, with the increment in the number of applications running in the SSAN, the system lifetime is dropped. More importantly, as expected, the relationship between the number of applications and the system lifetime is not linear. This is because that more applications are used as input naturally increase the possibility of finding common tasks between them, thus the proposed algorithm is able to fully utilize the common tasks to reduce the system energy consumption by executing those tasks only once and sharing the result among the required applications.

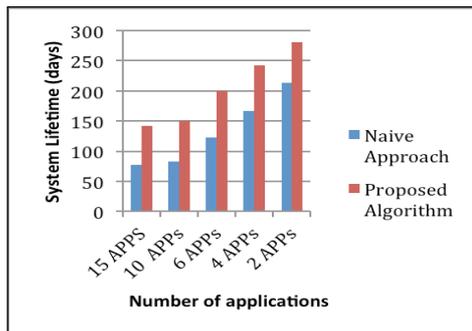


Figure 25 System lifetime with different number of applications

6.4.5 Scheduling Success Rate

As previously mentioned, SSR refers to percentage of tasks that are deployed in the system and successfully performed. If a task is missed, it might compromise all child tasks from the same application execution and force the sink node to redeploy these tasks in the system. Otherwise, the data sent back to the user can be flawed and further leads to wrong decision/action, or even an application could be unattended. We verified the SSR by varying the number of applications in the system, as shown in Figure 26. We can observe when the number of applications increases, the SSR for both algorithms are decreased. Compared to the benchmark, the proposed algorithm can guarantee more than 80% SSR under all test conditions, but the naive approach is failed to deliver such performance because it does not employ any technique to utilize the task sharing.

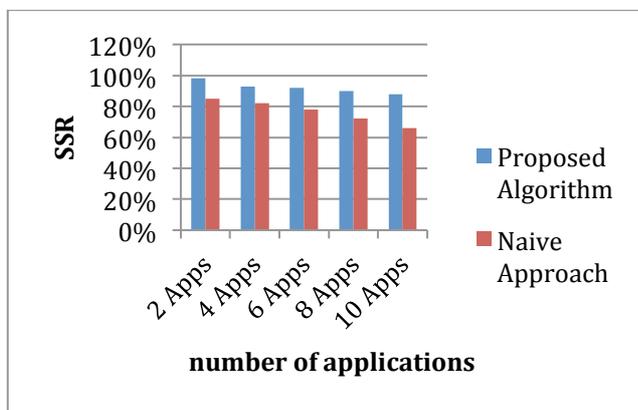


Figure 26 SSR with a different number of applications

6.4.6 - Comparison with Greedy and Random Task Allocation Schemes

We compare the performance of our algorithm with two different approaches: a greedy algorithm proposed in (XU *et al.* 2010) and the random, naive approach previously described. In this evaluation, we assess the capability of the algorithms to adjust to the changes in SSAN, as the

system workload is continually increased over time. The strategy adopted in the tests is shown below. After the system starts running, in each interval two new applications are added into the system. A time interval lasts for 30 seconds. Therefore, the following intervals were established: in T1, 2 applications were running; in T2 (30 sec after the system initialization), 4 applications were running; in T3 (60 sec after the system initialization), 6 applications were running; and so on, until T5 (with 10 applications are simultaneously running in the system). Figure 27 shows that as the number of applications increases, the system lifetime of all algorithms are decreased, but in our algorithm as the number of applications increase the decrease in system lifetime is slighter than in the Random algorithm, due to the fact that we are sharing tasks among applications. We also have better results compared to the Greedy algorithm. The differences between this figure and Figure 25 were due to the adaptation cost that the system had with the arriving applications.

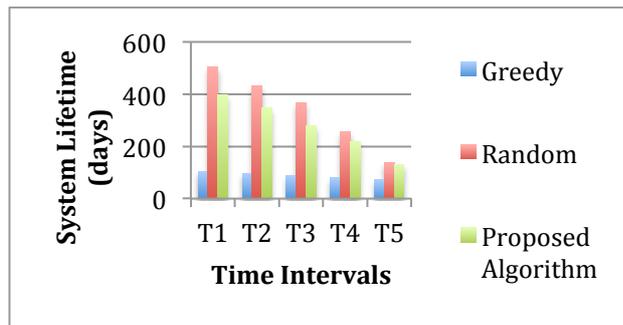


Figure 27 Comparison of system lifetime

In Figure 28, we analyze the differences among the SSR values for the considered intervals and algorithms. The resultant difference denotes the adaptation cost, which means the cost that the algorithms have to schedule the arriving applications. Our algorithm has the best results as the number of applications increase (as we can see in T5, and the same results in T3 and T4), followed by the greedy task allocation scheme. As the number of applications increase the SSR of the Greedy algorithm decays faster than the SSR of our algorithm. The proposed algorithm advantage is sharing common tasks, since it will have to schedule just the tasks that were not in the system instead of the complete set of tasks of new applications. At the beginning the greedy approach performs better for having no restriction to choose the nodes.

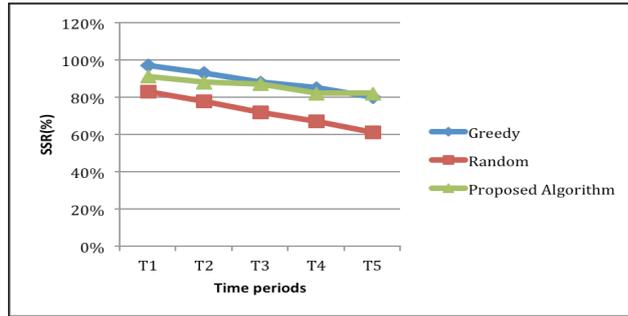


Figure 28 SSR for different time intervals

Finally, we have performed a test to analyze how the packet loss would affect each algorithm performance, since it is well known that wireless communication is very prone to errors and packet loss in real world settings. This was performed changing signal strength using Sun SPOT API. Such test provided very important results (Figure 29): as the packet loss rate increases the SSR of our algorithm decreases. This happens because the proposed algorithm expects useful information to perform the allocation scheme while the random algorithm uses any information at hand and the greedy scheme uses as much as possible. If our algorithm loses information they lose SSR while the greedy algorithm is less sensitive to packet loss. Such results demonstrate that our algorithm has, overall considering energy and SSR, the best performance. The experiments prove that it can effectively adapt to dynamic changes in the number of running applications.

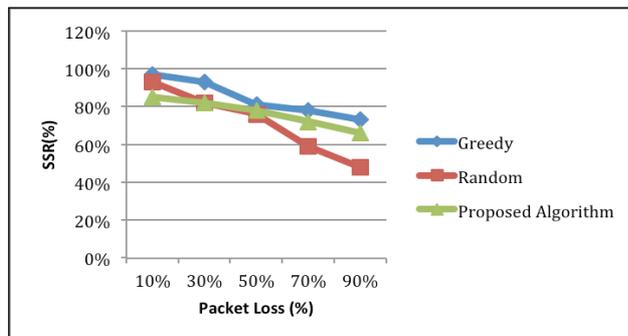


Figure 29 SSR with different levels of packet loss

6.5 Conclusions

In this Chapter, we address task allocation of multiple applications running on top of a SSAN. We developed a new scheduling algorithm that exploits common tasks to make effective task-

sensor assignments explicitly take full advantage of applications with common tasks and avoid repeating tasks unnecessarily. Experimental results show that our algorithm produces promising results in terms of SSR, energy consumption and system lifetime.

7 Adaptive QoS-Aware Service Selection and Allocation for Multiple Applications Execution in Heterogeneous Service-Oriented Wireless Sensor Networks

The content of this Chapter presents a service selection and allocation algorithm for Multiple SSANs called SERAPH (adaptive QoS-aware SERvice selection and allocation for multiple APplications execution in Heterogeneous service-oriented WSNs) that efficiently utilizes the underlying resources in service-oriented SSANs, and yet can provide the satisfied QoS level from sensor nodes within a dynamic environment regardless their hardware-level or OS-level differences to the end-users. The paper describing the algorithm was presented at IEEE International Conference on Sensing, Communication and Networking - SECON 2014 and it is entitled Adaptive QoS-Aware Service Selection and Allocation for Multiple Applications Execution in Heterogeneous Service-Oriented Wireless Sensor Networks”.

The remainder of the Chapter is organized as follows: Section 7.1 presents an introduction and Section 7.2 introduces the models used in this work. Our proposed approach is detailed in Section 7.3. Experimental results in Section 7.4 demonstrate the effectiveness of our approach. Finally, Section 7.5 draws some conclusions and illustrates future work.

7.1 Introduction

Wireless sensor networks (WSNs) generally consist of a large number of battery-operated devices equipped with one or more sensing units, processor, memory, and wireless radio. The main goal of WSNs is to collect raw measurement data about the interest physical phenomena, transform the data through a series of actions into more value added information, and forward these values to base stations for further processing by end-users. With a new noticeable trend of running different applications on multiple nodes belonging to different WSNs in order to better exploit the expensive physical network infrastructure, the resource allocations of nodes are highly dynamic with frequent changes according to different users' requirements and the statuses of nodes. The new WSN design trend is well represented by some existing industrial paradigms, such as: Internet of Things, Web of Things and cyber-physical systems. An obvious but key challenge faced by such kinds of systems is how to allow users to fully utilize the resources of heterogeneous sensor nodes to complete the WSN applications successfully without knowing the

details of underlying system infrastructure while still remaining energy efficiency and satisfying the various user requirements.

The task scheduling approach tries to model the WSN applications as task graphs and allocate their composite tasks to the ideal nodes whose offer the corresponding basic functionalities in the system according to different needs, has been well studied in several our works (LI *et al.* 2012), (LI *et al.* 2013), (FARIAS *et al.* 2012). However, this kind of approach may require the developers have the necessary knowledge to understand the specific programming interface offered in such approach due to data exchange between the running application and the underlying platform is OS specific, e.g. SenShare (LEONTIADIS *et al.* 2012). For providing a better solution in this context, we argue that service oriented architecture (SOA (BELL 2010)) is a well-suited approach, allowing to abstract the underlying heterogeneous sensor networks as a set of services that offer a certain functionalities (GEYIK, SZYMANSKI AND ZERFOS 2012) in a relatively standard and common way. We consider that services are a suitable abstraction for developing SSAN applications and the companion Web standards and protocols provide a proper format for data representation and exchange among the network and its users. Besides, the loosely-coupled property of service-oriented architecture enables dynamic service finding, selection and allocation within the system. By adopting the SOA in the SSAN domain, sensor nodes act as data providers and the network as a whole plays the role of service provider for client applications. Due to the well-known issues of limited hardware capabilities and energy resources in WSNs, each sensor node typically provides one or few non-divisible functionalities in its monitored area. In this paper, we call each kind of functionality as a primitive service. However, single nodes are often technically impossible to complete a SSAN application solely, we aim at using a group of nodes to collectively provide a composite service (e.g. a service that is formed through a suitable combination of basic functionalities) for performing SSAN applications.

Service composition and service-oriented mechanisms have been well studied in the context of traditional Web services (RAO and SU 2005). Compared to the Web service context where the availability of service providers and desirable working conditions are assured and formally defined by means of service level agreements (SLA), WSNs are highly dynamic as nodes often vary their status for conserving energy and spatially restricted as nodes can only provide the

desired functionality in a certain area. The always-on service paradigms of traditional web service composition approaches are inadequate to be used in SSAN since they are less sensitive to single-point-failure and utilize the limited resources less efficiently. Consequently, an adaptive service composition method is indeed required to make the provided services to adapt to the dynamic underlying infrastructure without explicit intervention from end-users at runtime. In SSANs, the quality of service (QoS) promised by the service provider is built on the top of the nodes that could be manufactured by various vendors and implemented using different programming language, OS and models, whereas the implementation details should be hidden from the end-users to substantially reduce the complexity of application development. More importantly, it is the service provider responsibility to maintain the quality of the provided services at the negotiated service level and prevent the unexpected violent quality change during the service time. In other words, the service composition method used in the service provider side should comprise a QoS-aware mechanism which guarantees the quality of provided service drop to below SLA at almost no time. Sometimes, In SSANs, the QoS requirements of applications could be vaguely or imprecisely defined or, yet, specified as intervals. This particular situation implies that the QoS-aware mechanism needs to not only address the given QoS requirements with clear numerical numbers in a satisfactory way, but also properly handle the QoS requirements presented in an indistinct way.

Given these premises, we developed a service selection and allocation algorithm called SERAPH (adaptive QoS-aware **SER**vice selection and allocation for multiple **AP**plications execution in **H**eterogeneous service-oriented WSNs) that efficiently utilize the underlying resources in service-oriented SSANs, and yet can provide the satisfied level of services from sensor nodes within a dynamic environment regardless their hardware-level or OS-level differences to the end-users. In summary, this section makes the following contributions:

- 1) An adaptive service-based approach is proposed to dynamically allocate applications to heterogeneous sensor nodes across multiple WSNs,
- 2) The allocation scheme is endowed with capability to handle both precise and fuzzy QoS requirements while keeping SLAs between providers and end-users,
- 3) Dynamically assign varied roles to sensor nodes and take the assignments into consideration when selecting and allocating the services for different applications.

7.2 Modeling and Problem Definition

In this section, we provide a detailed description of the models adopted in our study.

7.2.1 Service Modeling

A service is an abstract representation that defines a class of primitive service instances offering the same functionalities. It consists of functional and non-functional (QoS) parameters. Service parameters are defined as follows: $S = (F, \text{QoS}, \text{ID})$, where F is the non-divisible function provided by the sensor node to transforming a given input into desired output. QoS denotes the parameters that reflect the perceived quality of such transformation in general sense. ID is an indexing element for offering a unique identification to a specific service. From the above definition, a service-oriented WSN can be described as a group of services, abstracted from the sensor nodes and/or their sink nodes as following:

$$S = \{S_1, S_2, S_3, \dots, S_n\} \quad (16)$$

A non-divisible function of a service can be described as $F(S_i)$, which can be further described as a tuple:

$$F(S_i) = (I, O, KW, ID) \quad (17)$$

where I and O represents the input and output of a specific function respectively. KW is the keywords and/or functional description of a specific function for service registration and/or advertising purpose, and lastly ID is the unique identification of the function. For example, a primitive service S_1 called CamVideo merely captures and sends images of a certain area in sequence. Therefore, the input data of this service has to be image; no other types of data are allowable. The data type of the output is strongly related to the function of the service, in this example, the output has to be image as well. KW for the service could be named as image, video or something else, which lies on the will of system developers or service providers.

As mentioned early, QoS parameters can be considered as the information on services characteristics with the related quantitative measurements of the services. Compared to the traditional service-oriented paradigm, the QoS in WSNs vary more frequently over time due to the dynamic conditions of the underlying environment where WSNs exist. Besides that, QoS

parameters could be represented in a quantitative way with the given linguistic terms and their associated fuzzy sets. Please note that, some other approaches in the field, for instance, ontology, for encoding QoS parameters might also be applicable, however, their exact application details are beyond the scope of this work.

7.2.2 Application Modeling

In our service-based approach we model client applications as a composition of services. Let the application be defined as:

$$A = \{S_i \mid i = 1, 2, 3, \dots, m\} \quad (18)$$

where m is the number of services required to meet the application requirements. The same application arrived at different time could be performed by dynamic binding between primitive service instances as long as they cover the required geographic area. Therefore, a specific service-based application A at a given time can be further described as:

$$\begin{aligned} A(t) &= \{F_j(S_i) \in F(S_i) \mid S_i \in A \text{ and } i \\ &= 1, \dots, m, j \in [1, \dots, k_i]\} \end{aligned} \quad (19)$$

where k_i is the number of the primitive service S_i .

In this algorithm, we assume that all services are independent from each other. We also assume that once a service is initiated it cannot be interrupted until its completion. It is apparent that some services implemented in WSNs may be only entry services, which means the services do not receive any input and only produce data. Furthermore, we can also find out an exit service, a service that does not generate output and only receive input. In an application, the response to the end-user requesting is usually represented as an exit service.

7.2.3 System Modeling

We assume that there are k WSNs in the system denoted by the set $W = (W_1, W_2, \dots, W_k)$, deployed into the target (monitored) area G to ensure that any event can be detected by at least one sensor node. Any given WSN is modeled as an undirected graph $G = (V, E)$, where $V = (v_1, v_2, \dots, v_n)$ represents the set of sensor nodes and $E = (e_1, e_2, \dots, e_m)$ represents the set of all possible communication links among the sensor nodes in the same WSN. Each WSN W_i has a single sink node SN_i and a number of sensor nodes $|V|$. Sink SN_i is used as a gateway between the WSNs and external networks (such as the Internet) via a server. It dispatches user requests to the

selected sensor nodes which are able to provide the corresponding services in that WSN. Sink maintains a data repository that describes the capabilities (services), the residual energy and the location information of each sensor node (we are assuming that each node is capable of announcing its location either by a GPS or a trilateration algorithm). Upon the completion of services, the sink is in charge of communicating the results to the client applications. In addition, each sink SN_i is also responsible for the communication with other sinks to exchange data/intermediate result whenever this is required.

Any sensor node can overlap the monitoring region of a set of other sensor nodes. This indicates that some regions of the monitored area W are mutually covered by multiple sensors that can perform a certain task. For any given sensor node v_i in V , i denotes the index of the sensor node that belongs to the WSN. A sensor node v_i is capable of providing one or more Services depending on their capabilities to collect/sample different types of data, for instance, temperature, light, smoke, and movement. All sensor nodes are endowed with the same radio interface as well as communication and sensing ranges. Moreover, sensors can detect all events of interested occurred within their sensing range. According to the various services embedded on the nodes, we assign different roles to sensor nodes, which can be further described as: (i) *relay nodes* will receive only routing service; (ii) *sensor nodes*: can receive sensing and routing service; (iii) *decision nodes*: can receive decision, sensing and routing service and (iv) *actuators*: actuating, decision, sensing and routing service. Please keep in mind, when the incoming user requests are varied, the roles of the sensor nodes will be changed correspondingly. In addition, we assume that all the sensors in the WSN have a valid communication path to reach its sink. Communication interference between sensor nodes is not considered in this algorithm.

7.3 SERAPH: An Approach for service composition in heterogeneous WSNs

The algorithm we present in this section aims at efficiently performing multiple applications, constructed by service composition across WSNs, while reacting to changing network conditions by dynamically recomposing the service. In general, service allocation algorithms can be implemented in two ways, the centralized approach that proceeds with composition down the service hierarchy when finding the appropriate service providers within the system; and the distributed approach that proceeds in the opposite direction. Although the distributed approach outperforms the centralized approach in terms of scalability and is robust to single-point-failure,

this approach is somehow confined to sensor nodes communication range and only makes decision based on their immediate neighbors due to the information exchange between nodes. It is difficult for nodes far from each other to exchange reliable information without a high energy consumption cost (GEYIK, SZYMANSKI AND ZERFOS 2012). In other words, sensor nodes tend to make local decisions without knowledge of the overall availability of services in the system. Our proposed solution, SERAPH is therefore designed as a centralized approach that contains three phases: ***node filtering, service matching and service allocation.***

First of all, databases are created and populated in the sink node with data required to operate the system; the applications to be active in the environment are configured, as well as their parameters are set. Information on the applications and their parameters are deployed in the sensors flash memory. The services to be activated depend on which applications are running in the WSN. In the *node filtering* phase, we create a list of candidate WSN, meaning WSN that have nodes able to execute the desired application according to the availability function described in equation 20:

$$A(s, x, y, i, c) = \begin{cases} 1, & \text{if a sensor is available} \\ 0, & \text{if a sensor is unavailable} \end{cases} \quad (20)$$

where s is one of the service that an application requires, x and y are the geographical location for the monitoring event, i is the index of the sensor and c represents the conditions of the sensor (a set of rules to decide which kind of task the sensor is able to perform). The parameter c can be expressed by the following rules: If the required service s is an actuation service, the node must comply with the conditions about s , x , y and i and to have actuation capabilities. If the node has sensing capabilities and the node's distance to the monitored event is smaller than the distance to the sink node, then it could accept a sensing service. If it is a decision service the node must be neighbor of a node able to perform the sensing service used as input to the decision. All nodes should be able to accept a routing task.

In the *service matching* phase, when a new application arrives, the sink node will analyze the composition of the application. First, SERAPH puts all newly arrived applications into a queue according to their arrival time. Then, it will search in the newcomer applications for services that are common to the applications that are already currently running in the system. Our algorithm excludes the common services from the list of services required by the new applications. In order

to avoid losing data due to the reuse of a service, we have used the idea of session persistence presented in (WEI and VESILO 2006) and used in (LI *et al.* 2014). The rationale behind the solution is to maximize the intersection of execution time of the same service from different applications. For example, suppose that two sensor nodes p_1 and p_2 are able to provide the same service S from a monitored area. A service S requested from application A_1 is assigned to the sensor p_1 . When a new application A_2 arrives, it contains the same service request for the same area. At the same time, the service S of application A_1 is still running on S_1 . Instead of initializing a new sensor p_2 to perform the task, we allocate this task to the sensor p_1 which is already providing the same service to application A_1 . Before the service S of application A_1 stops running, the collected data can be used for both application A_1 and A_2 . In the meantime, only one measurement of energy is drawn from the system, and the longer intersection time for the applications, the more energy the system saves.

<p>Input: Service based applications, topology of WSNs, application requirements</p> <p>Output: The service-sensor assignment</p>
<p>// Node Filtering</p> <p>Consider Q as a queue for all the arrival applications</p> <p>When a new application A_i arrives, decomposes the application A_i into n services, where $S_{A_i} = \{S_1, S_2, \dots, S_n\}$.</p> <p>For each service in S_{A_i} look in the group of sensors S, using equation (5) to form a list of candidate sensors CS_i</p> <p>//Service Matching</p> <p>FOR each service S_i in n DO</p> <p style="padding-left: 40px;">For a given service $S_i \in S_{A_i}$, verify this service can be found being used in the WSN.</p> <p style="padding-left: 40px;">IF S_i is in the WSN</p> <p style="padding-left: 80px;">take S_i out of N</p> <p style="padding-left: 40px;">END IF</p> <p style="padding-left: 40px;">Put all the services a list Q</p> <p>END FOR</p> <p>//Service Allocation – In the sink node</p> <p>FOR each candidate sensors list CS_i</p> <p style="padding-left: 40px;">FOR each $v_j \in CS_i$ DO</p> <p style="padding-left: 80px;">A fuzzy system will generate the parameters</p> <p style="padding-left: 120px;">select the sensor with the best value to the result generated by equation (6)</p> <p style="padding-left: 40px;">End FOR</p>

<p>End FOR</p> <p>Return the optimal sensor v_j for n_i</p> <p>The latest information of nodes which perform the allocated services of application A_i is collected by their sink nodes for next arrival application</p>
--

Figure 30 The pseudo code of SERAPH

In the *service allocation* phase, we evaluate, in the sink node, each service in face of the incoming application. We set the parameters of the following equation based in a fuzzy system. For a given service, we derive the following equation:

$$SV = O \times TP \quad (21)$$

where SV means the sensor value that will be used in SERAPH to choose the most suitable node, TP means the topological distance, where by topological distance we mean the number of hops between the node and the sink node (HERRERA and HERRERA-VIEDMA 2000). O is an objective function that relates QoS parameters as it can be seen in equation 22:

$$O = \alpha \times D + \beta \times L + \gamma \times E \quad (22)$$

where α , γ and β are fuzzy-defined coefficients for the normalized delay, data loss and energy consumption, respectively. These three coefficients have a relationship $\alpha + \gamma + \beta = 1$. L represents the loss in percentage (%), the parameter $E = \frac{E_{residual} - E_{S_i}}{E_{residual}}$ represents the residual energy of the WSN. E_{S_i} denotes the energy consumption value for a given service. We use normalization because sensors may have different batteries. D represents the delay of the service completion.

Three QoS parameters (D , L and E) are considered: delay, data loss and average dissipated energy (directly related to the network lifetime). Delay is defined as the elapsed time from the sending of a packet by a sensor until its reception by the sink, while the dissipated energy is computed by dividing the total energy spent in the network by the amount of packets correctly received by the sink. The percentage of data loss is calculated by the ratio of the total data packets sent by the sensor nodes to the total data packets received by the sink node.

In order to dynamically adapt to the network conditions, we change the values of α , β and γ through a fuzzy system. By dynamically changing the coefficients, we are able to find the best node to execute a given service based on the current network conditions, reducing the negative impact on the desired requirements (delay, packet loss and energy consumption).

To build the fuzzy system used in this work we have to define its variables. The first step for building the semantic base is the definition of all the linguistic variables used in the algorithm. We assumed that the linguistic variables are defined through a quintuple (X, L, U, G, M) , where X is the variable symbolic name; L is the set of labels assumed by X ; U is the universe of discourse that contains all possible values assumed by X ; G is the syntactic rule, usually defined in the form of a grammar; and M are the semantic rules that define the meaning of each label L (also known as membership function). To simplify the G -grammars definition we adopted the approach based on the use of an ordered structure of linguistic terms, presented in (DELICATO, PROTTI, PIRMEZ and REZENDE 2006). Therefore, according to such approach, we supply directly the sets of primary terms (also known as fuzzy sets or sets of labels), distributed over a scale on which a complete (total) order is established.

For building the fuzzy systems, three linguistic variables were created, named *delay*, *energy* and *loss*, representing the respective QoS parameters. For the definition of these variables, we established that the second element of the quintuple (set of labels) is a function of the observed behavior of the services. The universe of discourse and its shape should be defined for each QoS variable. The threshold values of each fuzzy set, for each linguistic variable, should also be defined. The definition grammar G is also based on the approach used in (DELICATO, PROTTI, PIRMEZ and REZENDE 2006). In the adopted methodology, each fuzzy set (term set) of the fuzzy systems represents the behavior of one QoS parameter (or more than one, if they show similar behaviors) and its membership function is graphically represented by a trapezoidal shape.

The boundaries of the polygonal are defined as follows: the value representing the adopted confidence interval (95%) for each parameter on the left side (lower boundary) and the smallest simulated value whose membership degree in the next fuzzy set is equal to 1, on the right side (upper boundary). For a given variable, each fuzzy set must have an intersection with its next fuzzy set. Consequently, a given value belonging to its universe of discourse will be contained in, at least, one of its fuzzy sets.

The next stage in generating the knowledge base consists of building the inference rules that relate the linguistic values of the fuzzy variables. Fuzzy sets of the linguistic variables are related through logic operators, as in the statement “if delay is VL *and* Loss is S then α is small, β is high and γ is very small”. Once the fuzzy variables were defined, the inference rules for each fuzzy set are built taking into account the correlation between the fuzzy sets and the behaviors of

the simulated applications.

- Delay – Entrance variable that represents the time to execute the whole application. The universe of this variable is [0,40] measured in ms. The set of labels of these variable is (very small [0-10], small [10-20], acceptable [20-30], high [30-40], very high [40-50]).
- Loss - Entrance variable that represents the number of packets lost between nodes. The universe of this variable is [0,1] representing %. The set of labels of these variable is (very small [0-0.2], small [0.2-0.4], acceptable [0.4-0.6], high [0.6-0.8], very high [0.8-1.0]).
- Energy – Entrance variable representing the remaining energy of the node in a given time. The universe of this variable is [0,1] representing %.The set of labels of these variable is (very small [0-0.2], small [0.2-0.4], acceptable [0.4-0.6], high [0.6-0.8], very high [0.8-1.0]).

The outputs variables are the values of α , β and γ . The universe of this variable is [0,1]. The set of labels of these variable is (very small [0-0.2], small [0.2-0.4], acceptable [0.4-0.6], high [0.6-0.8], very high [0.8-1.0]).

The fuzzy rules used are as follow:

If Loss is high and Energy is low then α is small, β is high and γ is very small

If Loss is small and Energy is high then α is high, β is small and γ is very small

If Delay is small and Energy is high then α is acceptable, β small is and γ is small

If Delay is high and Energy is small then α is small, β is small and γ is acceptable

Once the best node is the latest information of nodes which perform the allocated Services of application A_i is collected by their sink nodes for updating the fuzzy system about the network conditions until the next arrival application.

7.4 Experiments

To evaluate the performance of SERAPH under a realistic scenario, we implemented it in real sensor nodes and designed a number of experiments under the context tailored to previous WSNs. The goal of the experiments is to further reveal the system performance from three aspects, namely: (i) the adaptation of service allocation to the underlying node states;

(ii) user QoS requirements satisfaction and (iii) the efficient use of resources. In order to do that we have used two selected metrics: system lifetime and allocation success rate (ASR).

The experiments were conducted in the SUN SPOT platform (GEIHS *et al* 2009). In all the following experiments, we used a mix of sensor nodes (real nodes and virtual nodes emulated by Solarium) to ensure we can fully study how algorithm performs in large scale WSNs.

7.4.1 Experimental Settings

In our experiments, each application contains 2 to 60 independent services. We also assumed that there are totally 2000 sensor nodes randomly distributed in a 2000 x 2000 m² field, organized as 10 independent WSNs with a single sink node placed in the top right corner for each one. We used LQRP (PASCHOALINO and MADEIRA 2007) as routing protocol to transmit data among sensor nodes. Each simulation runs for 900s. We adopted the same energy consumption model used in (WEI *et al.* 2014)

The first set of experiments (Section 7.4.3) aimed at assessing the system lifetime by comparing the energy consumption of our approach to a naive approach, which is a task allocation scheme used as a baseline, under the same conditions. In the naive approach, each user request is randomly allocated to a currently available node in the system. If the selected node is not able to perform the task due to certain reasons, another one will be randomly selected until the user request can be properly handled. In the second set of experiments (Section 7.4.4), we evaluated the allocation success rate to determine if our algorithm is able to, besides saving energy, properly allocate services to meet the application requirements. Starting from the third set of experiments (Section 7.4.5), we compared SERAPH with other known algorithms. Besides the naive approach previously described, we carefully selected two different proposed approaches with the similar design purposes: one is a greedy resource allocation algorithm, called FRG proposed in (TING *et al.* 2012), and the other one is a QoS-aware task allocation algorithm, called SACHSEN presented in (WEI *et al.* 2014) to conduct a series of experiments. The naive approach would also be used in the experiments. In the fourth set of experiments (Section 7.4.6), we evaluate SERAPH capacity to adapt to changing network conditions, such as data loss, available nodes variation and service delay. In this experiment, we compared SERAPH (the

version that has the capacity to adapt to changing network conditions) with the same selected algorithms what we used in the previous one.

7.4.2 Performance Metrics

In our experiments, the system lifetime and allocation successful rate (ASR) are used as the performance metrics for the evaluation. System lifetime is probably the most important metric for the WSNs performance evaluation. For evaluating this algorithm, we adopt the same definition of system lifetime used in (LI *et al.* 2014), which is equivalent to the time of the first sensor node has its energy completely depleted. As mentioned before, one of the goals of SERAPH is to appropriately allocate services to save energy so as to extend the system lifetime. However, the system lifetime metric does not fully reveal the performance of our proposed approach. We therefore used other two metrics introduced in (SHUGUANG *et al.* 2011): (i) *number of executed services* and (ii) *number of missed services* to derive a new metric - ASR. The number of executed services is used to measure the total number of services that have been successfully executed in the experiment. The number of missed services is used to measure the total number of services that fails to be performed due to various issues occurred in the system, such as data loss, sensor malfunctioning, loss of connectivity and nodes' energy depletion (LI *et al.* 2014). For example, a given service S_i could be allocated to a node with insufficient energy and therefore to be performed for a shorter period of time. Using these two metrics, we derive ASR as following:

$$ASR = \frac{\text{number of executed service}}{(\text{number of executed service} + \text{number of missed services})}$$

7.4.3 Experiments for evaluating System lifetime

The main goal of this set of experiments is to assess how long the WSNs would last under a given number of client applications. We have performed 5 separated experiments by using 2, 4, 6, 8, and 10 applications as system input, respectively. As shown in Fig. 31, with the increment of the number of applications simultaneously running in WSNs, the system lifetime is reduced accordingly.

More importantly, as expected, the relationship between the number of applications and the system lifetime is not linear. This is because that when more applications are run within the system, there is naturally an increase in the possibility of finding common services among them, thus leveraging the fully utilization of the common services. SERAPH approach well utilizes this idea to reduce energy consumption of nodes by executing common services only once and shared the result among all required applications in a certain time period.

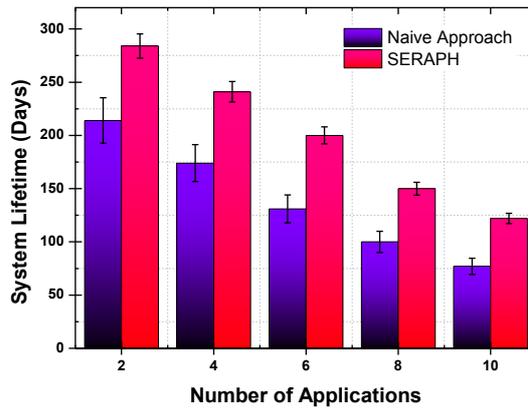


Figure 31 System lifetime with different number of applications

7.4.4 Experiments for evaluating the Allocation Success Rate

In this set of experiments, we evaluate the ASR by varying the number of applications in the system. We have performed 5 separated experiments using total 2, 4, 6, 8, and 10 applications as system inputs, respectively. As shown in Fig. 32, we can easily observe that ASR (percentage of services that are deployed in the system and successfully performed) for both algorithms is decreased as long as the number of applications increases. Compared to the naive approach, SERAPH can guarantee over 90% ASR under all experiments, but the naive approach fails to deliver such performance because it does not employ any technique to utilize the service sharing.

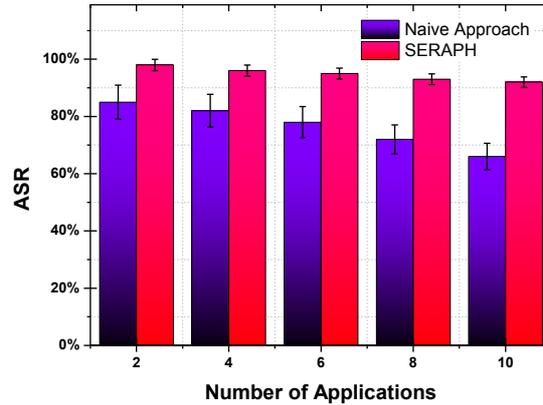


Figure 32 ASR with a different number of applications

7.4.5 Comparison with Selected Service Allocation Schemes

In these experiments, we firstly aimed at evaluating how the selected algorithms (SERAPH, the naive approach, the FRP algorithm (SHUGUANG *et al.* 2011), and SACHSEN algorithm (LI *et al.* 2014)) react under the entrance of new applications in the network in terms of system lifetime and ASR. The applications arrival plan used in this experiment is as below. After the system starts running, in each pre-determined equivalent time interval, which is set to 30s, two new applications are put into the system until the maximum number of applications (that is 10 applications in this experiment) is reached. Fig. 33 shows that as the number of applications increases along with time, the system lifetime of all selected algorithms are decreased in different extents. However, SERAPH consistently and substantially outperforms other algorithms, on all kinds of system workloads. SACHSEN showed the similar performance to SERAPH at all runs due to both designs implemented the idea of service sharing, which efficiently utilize the hardware resources to complete the WSN applications by directly reducing the chance of running redundant services from the same area at the same time. The more common service requests are simultaneously being performed within the system, the better performance of service-shared based algorithms will have. However, the non-prominent performance difference between these two algorithms are explained because SERAPH is relatively more robust to the changing network conditions from its embedded fuzzy system mechanism, and has more chance to select a correspondingly suitable node to perform the specific service from its embedded role-assignment mechanism. In addition, as we can easily observe from Fig. 33, FRG has the worst system lifetime performance among all algorithms. This is mostly due to the greedy design principle

makes it to always serve the node with the best QoS to the incoming user request regardless of the overall system performance.

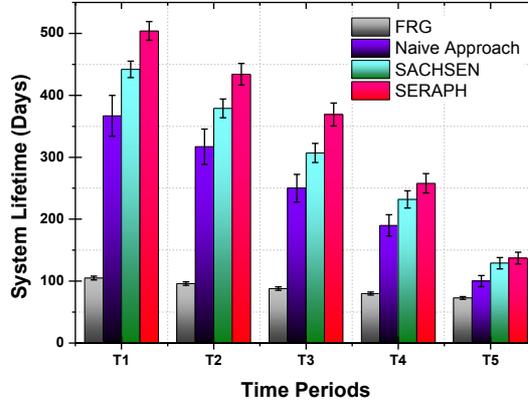


Figure 33 System lifetime comparison among the selected algorithms

Next, we study the ASR performance of all selected algorithms under the same conditions as used in the previous experiment. The ASR performance of selected algorithms has visible distinctions from two aspects in Fig. 34.

Firstly, we noticed that the fluctuation in performance was closely related to the algorithm itself, rather than the system workload. The naive approach showed the highest performance variation among all algorithms, while the FRG showed the slightest performance variation. The naive approach distributes incoming requests to the available nodes in a random way without considering the QoS requirements. Consequently, the naive approach should complete the least user requests successfully in total during the system lifetime. As opposed to the naive approach, FRG showed pretty stable during multiple tests running under the same conditions. Secondly, the performance of naive approach dropped sharply along with the increment of applications as expected. Surprisingly, the FRG approach owned the second degradation rate in performance during our tests. Although the FRG algorithm is still superior to all other approaches in each single test time period, this outcome is obtained in a very costly way of inevitably sacrificing system lifetime to offer best, even over-provisioning node to the requests. SERAPH did not show remarkable performance when the number of applications remains low in our tests. However, our approach performed better and better when the number of applications starts increasing, since it has more shared tasks. Especially, when the number of applications reaches 10, SERAPH obtains the almost same level performance as FRG but with a much longer system lifetime. This

experiment makes us believe SERAPH is particularly suitable for using in the multiple SSANs handling medium to heavy system load.

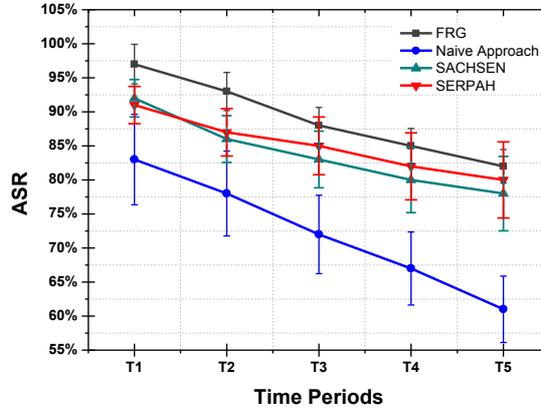


Figure 34 ASR for different time intervals

Finally, we performed an experiment to further analyze the algorithm performance in a data loss environment, since it is well known that wireless communication is very prone to errors and data loss in real world scenarios. We simulated the environment by manipulating the real-time signal strength from a built-in Sun SPOT function from RadioInputStream interface without affecting the connectivity of WSNs. The results are shown in Fig. 34. We can easily perceive that the ASRs of all selected algorithms are continuously dropped when the data loss rate increases. The performance of FRG approach is least affected by the data loss because it always tries to deliver the user request to one of available sensor nodes unless all of them are out of reach. Compared to the FRG, the naive approach adopted the simplest mechanism to deliver the task, which means once the sensor node is randomly selected; the user request is sent out straightaway without considering the result. This makes the random algorithm very prone to the wireless communication environment.

As we can observe from the Fig. 35, especially when the data loss rate reaches 50% and onwards, the performance of the naive approach is dropped dramatically compared to the other approaches. The performance of SERAPH appears to be relatively stable when the data loss rate is consecutively increased, since the best sensor candidate is chosen in advance. This mechanism requires that up-to-date sensor node information is collected regularly. If any eligible sensor nodes information is lost on the way to the sink, those nodes are automatically chosen to be

service providers for performing the awaiting requests. Data loss impact is therefore reduced to a certain extent.

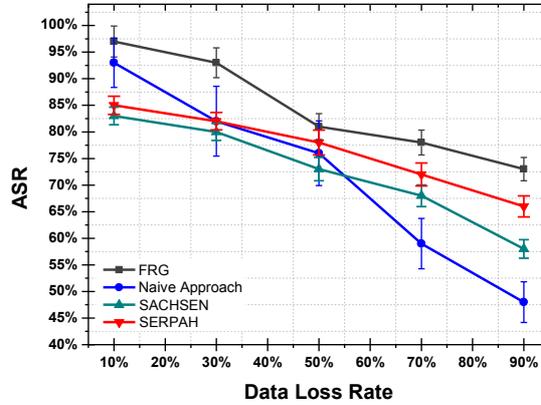


Figure 35 The ASR performance of selected approaches under different levels of data loss

7.4.6 Adaptability Experiment

This experiment aimed at evaluating how the selected algorithms (naive approach, FRG, SACHSEN and SERAPH) perform in terms of ASR if there were modifications in the network conditions. In order to simulate modifications in the network conditions, we have set time slots called S1, S2, S3 and S4 respectively. Each time slot represents a particular state of the application along the time. All time slots are equally lasting 900s and repeatedly occurred in sequence: (i) Time Slot 1 (S1) represents a great data loss (70%) network environment; (ii) Time Slot 2 represents a stage with no data losses (iii) Time Slot 3 represents a small-scale heterogeneous WSNs which reduces the number of nodes from 2000 to 200 and (iv) Time slot 4 represents a high delay network, which the service time is increased to twice than usual). Unlike the previous experiments, there are in total 15 applications constantly running within the SSANs.

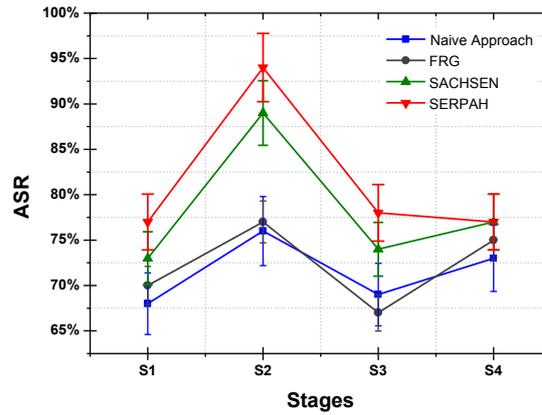


Figure 36 ASR in different stages

As we can observe from Fig. 36, SERAPH performed better than all other algorithms in terms of ASR since it is robust to all changing network conditions with its embedded fuzzy system rules while other algorithms were failed to do so.

7.5 Conclusions

In this Chapter, we presented a service selection and allocation algorithm called SERAPH for multiple applications execution in heterogeneous service-oriented WSNs. SERAPH exploits common services to make energy-efficient service-sensor assignments. In addition, it explicitly takes full advantage of fuzzy system and sensor roles to adapt the given network condition changings. Experimental results show that our algorithm produces promising results in terms of ASR and system lifetime. As future work, we intend to extend SERAPH to work in as a decentralized algorithm and introduce new QoS metrics to the fuzzy system.

8. Conclusions

The goal of this Chapter is to summarize the results obtained through this Thesis development, and to point out future works. In Section 8.1, the answers to some fundamental questions established according to the hypotheses raised in this Thesis will be given. In Section 8.2, we present the main contributions and results obtained in this thesis. Section 8.3 will outline future research directions.

8.1 Answers to the Fundamental Research Questions

In this thesis, a framework for developing Smart Spaces Applications using SSANs was presented. In order to guide the investigation of this thesis, various hypotheses were raised and presented in Chapter 1:

Hypothesis 1: "A framework for control and decision for smart environments using SSAN should: (I) be able to deal with the different requirements of multiple applications simultaneously; (II) schedule multiple applications; (III) execute data fusion for multiple applications; (IV) make decisions about their behavior; (V) be scalable; and (VI) maintain the decision making process as much as possible within the network."

Hypothesis 2: "Data fusion techniques applied to SSAN must maintain data semantics and accuracy."

Hypothesis 3: "Techniques for SSAN scheduling should take into consideration the applications priority and the dependencies between tasks of an application, as well as common tasks among various different applications."

Hypothesis 4: "Decision Systems for SSAN should take application integration into consideration, as well as trying to maintain the decision in-network."

Based on the assumed hypotheses, the following fundamental research questions were elaborated and evaluated throughout this work:

- What are the requirements of a framework able to manage the decisions made in a SSAN?

- How to execute semantic data fusion involving multiple applications, so that the result of the data fusion is accurate, has the least possible delay and a high fusion degree?
- How to schedule multiple applications so as to avoid activities to be repeated unnecessarily?
- How to build a decision system so as to integrate decisions and maintain the decision as much as possible in-network?

All the results obtained arise from these fundamental questions. The main objective when defining the fundamental questions was to help defining the main research points that would be analyzed during the investigation of the hypotheses that were presented and to establish different ways to reach the contributions made by this thesis.

In order to find the presented answers, a prior detailed study of SSAN was made. Since every facet of the SSAN was studied and research opportunities were found, a framework was defined. This framework had its components specified and implemented. Therefore, the fundamental question related to the SSAN framework, as it was proposed in Chapter 1, can be answered as following:

- What are the requirements of a framework able to manage the decisions made in a SSAN?

Answer: As presented in Chapter 3, the design and development of a successful SSAN is not trivial. It is necessary to deal with different kinds of challenges imposed by the nature of a WSN on one hand and the requirements of multiple applications on the other hand. In the following, we explore some important requirements necessary for constructing a SSAN from the perspective of application and network requirements.

(i) Energy efficiency: This is perhaps the most stringent requirement for almost all WSN applications, since sensor nodes are generally powered by non-rechargeable batteries, which are inevitably limited in energy capacity. This was accomplished through the specification of new MDFs and new task allocation algorithms presented in Chapters 5, 6 and 7.

(ii) Dynamic resource allocation: The far-end users can submit diverse applications to the SSAN at any time. Information sharing further extends the benefits of collaboration to SSANs since some intermediate results can be directly reused among applications and the associated

repeated operations can be mostly avoided. The task allocation algorithms presented in Chapters 6 and 7 met this requirement.

(iii) Collaboration and information sharing: This is an increasingly important requirement for many sensor network applications, especially in SSANs. The MDFs proposed in Chapter 5 met this requirement.

(iv) Dedicated platform construction: SSAN design is fully compatible with the traditional application-specific WSN by constructing a virtual sensor network over the same physical network as a dedicated platform for each application. To accomplish such design is not trivial, especially when the underlying network is heterogeneous (composed of different types of devices). Within the heterogeneous sensor network, different communication technologies, such as Zigbee, Bluetooth, UWB and WIFI are used and even different operating systems could be installed on the sensor nodes. This requires appropriate bridging mechanisms to hide the hardware-specific details from end-users and present each constructed virtual sensor network as a unique and dedicated platform to the application. This requirement was out of the scope of this thesis and will be dealt in a future work.

- How to execute semantic data fusion involving multiple applications, so that the result of the data fusion is accurate, has the least possible delay and a high fusion degree?

Answer: As described in Chapter 5, this thesis presented enhanced MDFs for SSANs. Such MDFs augment traditional MDFs to work in the SSAN environment by exploring application similarities on data thresholds. We can conclude that the data range and the weights assigned to applications (representing the relative priority assigned to each application) are extremely important in the fusion process. Since applications have different degrees of importance, it is intuitive to assume that their data have different degrees of importance. If an application less relevant but with a large data range has the same degree of importance of other existing applications, this will lead to an error, i.e., a result which does not mirror the current situation of the environment. If we consider the data semantics, and weight it according to its importance, the MDF will return a result closer to reality.

As the number of applications in a SSAN grows, so does the amount of tasks they have in common. Therefore, the fundamental question related to scheduling in SSAN, as proposed in Chapter 1, can be stated as such:

- How to schedule multiple applications so as to avoid activities to be repeated unnecessarily?

Answer: As presented in Chapters 6 and 7, in this thesis we address task allocation of multiple applications running on top of a SSAN. In Chapter 6, we presented a new scheduling algorithm that exploits common tasks to make effective task-sensor assignments explicitly take full advantage of applications with common tasks and avoid repeating tasks unnecessarily. In Chapter 7, we present SERAPH that also exploits common services to make energy-efficient service-sensor assignments. In addition, it explicitly takes full advantage of fuzzy system and sensor roles to adapt the given network condition changings. The capacity of avoiding common tasks and adapting to the network conditions is essential to the SSAN scenarios.

- How to build a decision system so as to integrate decisions and maintain the decision as much as possible in-network?

Answer: Although not a primary target of our thesis, we have found (in the following papers that we have published “*A control and decision System for Smart Buildings using Wireless Sensor and Actuator Networks*”, “*A control and decision System for Smart Buildings*” and “*Sistema de Controle e Decisão para Edifícios Inteligentes usando Redes de sensores e atuadores sem fio*”) that a control and decision-making system for smart space applications should be decentralized. Therefore, there is no need of transmitting raw data to a centralized entity (the Base station) on a hop-by-hop basis, neither the final decisions back to the WSN, thus reducing the need of radio transmissions and saving energy for the SSAN. Moreover, since in our solution the decision-making process is incorporated into the SSAN, thus getting closer to the actuators, we achieve: (i) faster execution of decisions, (ii) better performance of the smart building, and, consequently, (iii) enhanced energy efficiency. Also, integrating applications on Smart Spaces has as purpose to save energy by reducing worthless task repetitions and fostering collaboration between applications to achieve common goals. Such integration further minimizes energy waste by avoiding undesirable states.

8.2 Major contributions and published articles

The main contributions of this thesis can be divided in conceptual and specific contributions. The conceptual contributions were found due to the investigations in the published works and the experience gained during the development of this thesis. The specific contributions are as follows.

Contributions related to our Literature Review (presented in Chapter 3):

- A taxonomy for SSAN;
- A Systematic Literature Review of SSAN;

Contributions related to our proposed framework:

- The characterization of a decentralized framework for Smart Spaces using SSAN (presented in Chapter 4);
- The specification of a framework for SSAN and an implementation of an instance of it, called ASGARD (presented in Chapter 4);
- The specification, implementation and testing for new fusion techniques regarding the new emerging SSAN scenario (presented in Chapter 5);
- The specification, implementation and testing for task scheduling algorithms in SSAN's (presented in Chapters 6 and 7);

From these contributions, several articles were published. The following are those directly related to the thesis.

- SBRC 2012 – Henrique Ribeiro, Luci Pirmez, Flávia Delicato, Claudio Miceli, Conde: Um sistema de controle e decisão para Edifícios Inteligentes usando Redes de Sensores e Atuadores sem Fio.

Abstract: A way of improving energy efficiency in Smart Buildings is by applying control and decision-making mechanisms to the devices in the building in order to automate their operations. Another way is to distribute these mechanisms between nodes in a wireless sensor and actuator network (WSAN). This research proposes a decentralized control and decision-making system for applications in Smart Buildings that use a WSAN known as CONDE. The tests demonstrate

that, since monitored magnitudes are not transmitted to a centralizing entity, there is gain in response-time, in a way that the decision is applied quicker, saving time and energy, in both the network and the building.

- LCN 2012 - Claudio M. de Farias, Luci Pirmez, Flávia Coimbra Delicato, Igor L. Dos Santos, Albert Y. Zomaya, Information Fusion Techniques Applied to Shared Sensor and Actuator Networks.

Abstract: This work presents an adaptation (and enhancement) of a well-known fusion technique in order to deal with multiple applications simultaneously in a context of Shared Sensor and Actuator Networks (SSAN). SSAN allow the sensing infrastructure to be shared among multiple applications that can potentially belong to different users instead of assuming an application-specific network design. We also present simulations and tests conducted with the proposed solution on real nodes to validate our proposal.

- ICOIN 2013 - Claudio M. de Farias, Luci Pirmez, Flávia C. Delicato, Wei Li, Albert Y. Zomaya, José N de Souza, A Scheduling Algorithm for Shared Sensor and Actuator Networks.

Abstract: Shared Sensor and Actuator Networks (SSAN) represents a new design trend of Wireless Sensor Network (WSNs) that allows the system infrastructure to be shared among multiple applications submitted by different users instead of the original application-specific network design. In this context, we propose a task scheduling algorithm that well utilizes the characteristics of applications with common tasks to improve the system energy efficiency, so as to extend the most concerned performance metric system lifetime. Our proposal is validated by both computer simulations and real sensor nodes tests.

- UIC 2013 Claudio Farias, Luci Pirmez, Flavia C. Delicato, Henrique Soares, Igor L. dos Santos and Luiz. F. R. C. Carmo; A Control and Decision System for Smart Buildings.

Abstract: The employment of a control and decision process supported by wireless sensor and actuator networks (WSANs) is a promising way to improve energy efficiency of Smart Buildings. We present and evaluate CONDE, a decentralized system for decision and control for Smart Building applications based on WSANs. Performed experiments shown that since data is processed within the network instead of transmitted to a central location, there is a gain in terms

of the system response time and resource consumption. Therefore, CONDE improves the energy efficiency of both the monitored building and the WSN infrastructure, when compared to centralized approaches. Moreover, it exploits integration of applications at the decision level to further improve the system efficiency.

- JPDC 2014 - Wei Li, Flavia C. Delicato, Paulo F. Pires, Young Choon Lee, Albert Y. Zomaya, Claudio Miceli, Luci Pirmez; Efficient Allocation of Resources in Multiple Heterogeneous Wireless Sensor Networks

Abstract: Wireless Sensor Networks (WSNs) are useful for a wide range of applications, from different domains. Recently, new features and design trends have emerged in the WSN field, making those networks appealing not only to the scientific community but also to the industry. One trend is towards running different applications on heterogeneous sensor nodes deployed in multiple WSNs in order to better exploit the expensive physical network infrastructure. Another trend regards the capability of accessing sensor generated data from the Web, fitting WSNs in novel paradigms of Internet of Things (IoT) and Web of Things (WoT). Using well-known and broadly accepted Web standards and protocols enables the interoperation of heterogeneous WSNs and the integration of their data with other Web resources, in order to provide the final user with value-added information and applications. Such emergent scenarios where multiple networks and applications interoperate to meet high level requirements of the user will pose several changes in the design and execution of WSN systems. One of these challenges regards the fact that applications will probably compete for the resources offered by the underlying sensor nodes through the Web. Thus, it is crucial to design mechanisms that effectively and dynamically coordinate the sharing of the available resources to optimize resource utilization while meeting application requirements. However, it is likely that Quality of Service (QoS) requirements of different applications cannot be simultaneously met, while efficiently sharing the scarce networks resources, thus bringing the need of managing an inherent tradeoff. In this paper, we argue that a middleware platform is required to manage heterogeneous WSNs and efficiently share their resources while satisfying user needs in the emergent scenarios of WoT. Such middleware should provide several services to control running application as well as to distribute and coordinate nodes in the execution of submitted sensing tasks in an energy-efficient and QoS-enabled way. As part of the middleware provided services we present the Resource Allocation in

Heterogeneous WSNs (RA-HWSN) algorithm. RA-HWSN is a new resource allocation heuristic for systems composed of heterogeneous WSNs that heuristic effectively deals with the tradeoff between possibly conflicting QoS requirements and exploiting heterogeneity of multiple WSNs.

- ETT 2014 - Claudio Miceli de Farias, Henrique Soares, Luci Pirmez, Flávia Delicato, Igor dos Santos, Luiz Fernando Carmo, José Neuman, Albert Zomaya.; A Control and Decision System for Smart Buildings using Wireless Sensor and Actuator Networks

Abstract: A research field that makes use of Information and Communication Technologies (ICTs) to provide solutions to contemporaneous environmental challenges such as greenhouse gas emissions and global warming is the “smart building” field. The use of Wireless Sensor and Actuator Networks (WSANs) emerges as an alternative for the use of ICTs in the Smart Buildings. However, most of smart building applications make use of centralized architectures with sensing nodes transmitting messages to a base station wherein, effectively, the control and decision processes happen. In this context we present CONDE, a decentralized control and decision-making system for smart building applications using WSANs. CONDE main contributions are: (i) the decentralization of the control and decision-making processes among WSAN nodes, saving energy of both the WSAN and the building; (ii) the integration of applications through sharing the sensed data and chaining decisions between applications within the WSAN, also saving energy of both the WSAN and the building, and (iii) the provision of a consensual multilevel decision that takes into account the cooperation among nodes to have a broader view of the monitored building. Performed experiments have shown CONDE gains in terms of: (i) response time; (ii) system efficiency and (iii) energy savings from the network and the building

- ACM Survey (in submission) - Claudio Miceli de Farias, Wei Li, Flávia C Delicato, Luci Pirmez, Albert Y Zomaya, Paulo F. Pires, José N. De Souza; Shared Sensor Network Design to Support Multiple Applications Deployment: A Survey

Abstract: While wireless sensor networks (WSNs) have been traditionally tasked with single applications, in recent years we have witnessed the emergence of Shared Sensor Networks (SSN) as integrated cyber-physical system infrastructure for a multitude of applications. Instead of assuming an application-specific network design, SSNs allow the underlying infrastructure to be

shared among multiple applications that can potentially belong to different users. On one hand, a potential benefit of such design approach is to increase the utilization of sensing and communication resources, whenever the underlying network infrastructure covers the same geographic area and the sensor nodes monitor the same physical variables of common interest for different applications. On the other hand, compared with the existing application-specific design, the shared sensor network approach poses several research challenges at different aspects of WSNs. In this article, we present a systematic literature survey on shared sensor networks. The main objective of the paper is to provide the reader with the opportunity of understanding what has been done and what still remains open in this field, as well as which are the pivotal factors of this evolutionary design and how this kind of design can be exploited by a wide range of WSN applications.

- SECON 2014 (in submission) - Claudio; Miceli de Farias, Wei Li, Flávia C Delicato, Luci Pirmez, Albert Y Zomaya; Adaptive QoS-Aware Service Selection and Allocation for Multiple Applications Execution in Heterogeneous Service-Oriented Wireless Sensor Networks

Abstract: A noticeable Wireless Sensor Networks (WSNs) design trend, running multiple applications over the same infrastructure, has been well represented by several existing industrial paradigms, e.g. Internet of Things, Web of Things and Cyber-Physical Systems. These multiple applications will have to share the network sensing, processing and communication infrastructure. In this paper, with the goal of fully utilising the network infrastructure and inspired by a service-oriented architecture, we modelled applications as sets of primitive and independent services to be deployed into sensor nodes. By using such approach, sensor nodes can be easily divided into roles according to their offered services and we can further identify common services among applications so that to leverage service sharing and optimizing the use of the WSN resources. With these premises, we propose an adaptive service selection and allocation algorithm called SERAPH that can efficiently utilise the underlying hardware resources, and yet provide the satisfied level of QoS from heterogeneous sensor nodes without involving complicated operations from end-users. Sometimes, QoS requirements of WSN applications could be represented as fuzzy values, SERAPH is intentionally designed with the capability of handling such cases. The experimental results show that SERAPH provides good performance to the end-users and

achieves high energy efficiency, making it more attractive for use in more WSNs implementations adopted the same design trend.

The following papers are indirectly related to the thesis:

- SBCARS 2010 - Taniro Rodrigues, Priscilla Dantas, Flávia Delicato, Paulo Pires, Claudio Miceli, Luci Pirmez. Using MDA for Building Wireless Sensor Network Applications

Abstract: Research on Wireless Sensor Networks (WSN) has evolved with potential applications in several domains. However, the building of WSN applications is hampered by the need of programming in low-level abstractions provided by sensor OS and of specific knowledge about each application domain and each sensor platform. We propose a MDA approach to develop WSN applications. This approach allows domain experts to directly contribute in the development of applications without needing low level knowledge on WSN platforms and, at the same time, it allows network experts to program WSN nodes to met application requirements without specific knowledge on the application domain. Our approach also promotes the reuse of the developed software artefacts, allowing an application model to be reused across different sensor platforms and a platform model to be reused for different applications.

- SBSEG 2010 – Hélio Salmon, Claudio Miceli de Farias, Luci Pirmez, Silvana Rossetto, Rodrigo Pirmez, Flávia Delicato, Luiz Carmo. Sistema de Detecção de Intrusão Imuno-inspirado customizado para Redes de Sensores Sem Fio.

Abstract: In this work we propose an IDS framework inspired in the Human Immune System and a decentralized and customized version of the dendritic cell algorithm to be applied in the wireless sensor network context. Its basic feature is the nodes neighborhood monitoring and collaboration to identify an intruder. The work was experimentally evaluated in order to demonstrate its efficiency in detecting a denial-of-sleep attack.

- IEEE/IFIP 2011 – Taniro Rodrigues, Priscila Dantas, Flávia Delicato, Paulo Pires, Luci Pirmez, Thaís Batista, Claudio Miceli de Farias, Albert Zomaya. Model-Driven Development of Wireless Sensor Network Applications.

Abstract: Research on Wireless Sensor Networks (WSN) has evolved with potential applications in several domains. However, the building of WSN applications is hampered by the need of programming in low-level abstractions provided by sensor OS and of specific knowledge about each application domain and each sensor platform. We propose a MDA approach to develop WSN applications. This approach allows domain experts to directly contribute in the development of applications without needing low level knowledge on WSN platforms and, at the same time, it allows network experts to program WSN nodes to meet application requirements without specific knowledge on the application domain. Our approach also promotes the reuse of the developed software artifacts, allowing an application model to be reused across different sensor platforms and a platform model to be reused for different applications.

- IJCSNS 2011 - Luci Pirmez, Nilson Rocha Vianna, Reinaldo de Barros Correia, Luiz Fernando Rust da Costa Carmo, Claudio Miceli de Farias and Helio Mendes Salmon, EWIDS: An Extended Wireless IDS for Metropolitan Wireless Networks Based on Kinematical Analysis.

Abstract: Wireless metropolitan area networks (WMANs) are well known to subject users or applications and to a vast gamma of security risks, hindering security critical distributed applications from employing this type of network as a communication infrastructure. Most existing approaches for addressing WMAN security issues use cryptography-based mechanisms or ad-hoc adapted versions of traditional Intrusion Detection Systems (IDS) for wired networks. While the first approach may lead to unfeasible computation costs for mobile hand-held devices, the second exhibits a high dependency on the freshness of their attack-signature databases, besides not considering any inherent characteristic of wireless networks, such as mobility. Thus, we present EWIDS (Extended Wireless IDS); a lightweight IDS specially designed for WMANs, which detects anomalous wireless device transmissions by employing kinematical analysis on the motion of users' mobile devices. EWIDS also takes into account the decision information generated by transmitter fingerprint mechanisms used to identify wireless device. Both information is integrated through a fuzzy logic engine in order to increase the system performance. Realistic simulations based on WMAN scenarios revealed that our approach is very promising, since worst-case results have shown high correct alarm rates associated with low false positive rates.

- SBRC 2011– Tiago França, Paulo Pires, Flávia Delicato, Luci Pirmez, Claudio Miceli, Web das Coisas: Como conectar de dispositivos físicos ao mundo digital e desenvolver aplicações que usem tais dispositivos.

Abstract: In the near future the number of computing devices connected to the internet will be massive. Those devices are smart objects that are becoming part of people's life and will be quickly found in any place. The Web of Things (WoT) proposal is to integrate smart devices as first-class citizens into the World Wide Web. So, users will be able to access those physical objects via URLs, to browse them, and to compose the resources provided by such devices into physical mashups. The goal of this short course is to present the state of the art in WoT application development. First, we will provide an overall view of the Internet of Things (IoT) concept and how such concept evolved to forge the Web of Things paradigm. Next, we will present the underlying software architecture currently employed in the WoT projects. Following, we will detail the current architecture employed in WoT projects. Finally, we will present how the resources available in smart devices can be composed in Web applications, called mashups. During the course, practical examples of WoT application development will be shown using the Sun SPOT platform.

- IJWIN 2012 – Hélio Salmon, Claudio Miceli, Luci Pirmez, Silvana Rossetto, Flávia Delicato, Paulo Aguiar, Luiz Fernando Rust, Paula Loureiro, Intrusion Detection System for Wireless Sensor Networks using Danger Theory immune-inspired techniques.

Abstract: An IDS framework inspired in the Human Immune System to be applied in the wireless sensor network context is proposed. It uses an improved decentralized and customized version of the Dendritic Cell Algorithm, which allows nodes to monitor their neighborhood and collaborate to identify an intruder. The work was implemented and tested both in simulation and in real sensor platform scenarios, comparing them to each other and was also compared to a Negative Selection Theory implementation in order to demonstrate its efficiency in detecting a denial-of-sleep attack and in energy consumption. Results demonstrated the success of the proposal.

- SEMISH 2013– Taniro Rodrigues, Claudio Miceli, Flávia Delicato, Luci Pirmez, Paulo Pires, Priscilla Victor, Thais Batista, DSL e Metodologia para Construção de Aplicações Ubíquas.

Abstract: Wireless Sensor Networks and Actuators (WSANs) are a major component of Ubiquitous systems. However, the complexity of programming such networks requires domain experts to know the specifics of each sensor platforms available, thus increasing the learning curve for developing applications for these networks. In this paper we report on the integration of two existing works that aim to facilitate WSAN application building and increase the effectiveness of the development process for such environments. This paper integrates: (i) a domain specific language, (ii) a method and infrastructure to develop ubiquitous applications. We evaluate the proposed integration through a comparative analysis and a proof of concept.

- Book Chapter in Large Scale Network- Centric Distributed Networks 2013 – Flávia Delicatio, Igor Leão, Luci Pirmez, Paulo Pires and Claudio Miceli de Farias, Challenges In The Use of Wireless Sensor Networks for Monitoring The Health of Civil Structures, in Large Scale Network-Centric Distributed Systems.

Abstract: This chapter provides an overview of the challenges faced in the design of new techniques for enabling new decentralized solutions of large-scale wireless sensor networks (WSNs) in the structural health monitoring (SHM) domain. It introduces the definition of SHM along with the concept of WSNs. The chapter discusses the concepts of SHM and WSNs apart from one another. The chapter explains existent solutions employing WSNs in the context of SHM. It focuses on SHM techniques based on the use of accelerometers; but in further investigations, this classification can be expanded to works that use other kinds of sensing devices, for example, strain gauges, following the same logic of higher degrees of decentralization and in-network processing. The concept of “generations of sensor networks for SHM” was used for such classification. Each generation is presented by describing respective examples of works found in the current literature.

The contributions previously listed do not show all the results obtained by the thesis up to date. Other results obtained in this thesis are the implementation and the knowledge acquired for the evaluation of the ASGARD. Through the ASGARD, the investigation of future works on this thesis, as it is discussed in the next section, will continue.

8.3 Future Work

The investigation conducted in this thesis leads to the identification of other research opportunities. These opportunities are described as follows:

- Scheduling algorithms for SSAN: Creating distributed scheduling algorithms to SSAN that deal with application in an unified way, not only prioritizing some applications over others but also preventing redundant tasks in multiple applications from being executed, saving time, processing, and consequently, energy; Also working with multiple sink nodes in the SSAN. A centralized approach is not desirable for SSANs due to its limitations in scalability and fault tolerance. First, a SSAN is usually of large scale in terms of the number of nodes and hop counts. Hence, it is inefficient or even impossible to achieve global information sharing that is required by centralized optimization algorithms. Second, in a centralized approach, much of the computation and communication happens on a single point resulting in a single point of failure.

- Extending data fusion techniques that use computer intelligence techniques to deal with multiple applications. Using computer intelligence techniques, such as fuzzy systems or Neural Networks, we are able to dynamically adjust the applications weights in the MDFs. We could also use one of the computer intelligence techniques to infer about some state of the monitored Space.

- Developing new security frameworks able to deal with multiple applications simultaneously. SSANs are subject to vulnerabilities associated with wireless communication and ad-hoc organization, both inherent characteristics of this type of network. Furthermore, in scenarios involving unprotected hostile outdoor areas, SSANs are prone to different types of attack which can compromise reliability integrity and availability of the sensor data traffic and sensor lifetime as well. We propose a dynamic resilient security component to be added to ASGARD that deals with security requirements of multiple applications.

- Work in new routing algorithms able to deal with multiple sinks and multiple applications that enter dynamically in the network. Routing is one of the most challenging problems in WSNs due to its infrastructure-less nature and unreliability of nodes. Although many routing protocols were proposed for WSNs, most of them still rely on unique node identifiers that applications are

expected to provide. For applications to use such routing protocols, they need another resource discovery protocol that allows senders to query the network for the identifiers of the nodes they want to communicate with. Such separation between resource or service discovery and route discovery results in unnecessary control traffic in the network that could drain the sensor batteries faster. So in the SSAN context the routing algorithms must consider the multiple applications design.

- Designing a mechanism capable of deciding what MDF to apply, given a certain amount of data available is definitely a future direction of great importance. Through some strategies, such as computational intelligence, finding the most appropriate MDF is useful in environments where applications change quickly, such as the SSAN scenario.

- An investigation of how to combine different MDFs, looking for combinations that show better results (in terms of energy consumption, FP, FN, TP, TN, sensibility and specificity). The combination of different MDFs has the potential to lead to better results. Using different MDFs such as the Dempster-Shafer Inference after applying a Fault Tolerant Interval technique can lead to a better results (in terms of energy consumption, FP, FN, TP and TN), since the Fault Tolerant Interval technique shapes a certain amount of data, thus providing a better set of samples to be used by the Dempster-Shafer Inference.

- An investigation of approaches for applying the ideas of shared sensor network design to Cloud of Things. A comprehensive design that incorporates both integrating heterogeneous WSN resources to clouds and a shared sensor network design would be more beneficial for the future applications along with the further development of IoT.

- An investigation about dedicated platform constructions: Since this SSAN requirement was not covered by this Thesis, a future work is to build appropriate bridging mechanisms to hide the hardware-specific details from end-users and present each constructed virtual sensor network as a unique and dedicated platform to the application. A REST architecture to be used by these mechanisms will be further investigated.

The list of future works previously discussed presents the main research opportunities that can be directly derived from the work presented in this thesis. It is important to highlight that the opportunities could only be listed due to knowledge acquired during the thesis.

References

- A. Mandal, K. Kennedy, C. Koelbel, G. Marin, J. Mellor-Crummey, B. Liu and L. Johnsson. Scheduling Strategies for Mapping Application Workflows onto the Grid. In IEEE International Symposium on High Performance Distributed Computing (HPDC 2005), 2005.
- Akyildiz, I. F., SU, W., Sankarasubramaniam, Y., AND Cyirci, E. 2002. Wireless sensor networks: A survey. *Comput. Netw.* 38, 4 (March), 393–422.
- Alex, Hitha; Kumar, Mohan; Shirazi, Behrooz; MidFusion: An adaptive middleware for information fusion in sensor network applications, *Information Fusion*, Volume 9, Issue 3, July 2008, Pages 332-343, ISSN 1566-2535, <http://dx.doi.org/10.1016/j.inffus.2005.05.007>.
- Andre Kaustell, M. Mohsin Saleemi, Thomas Rosqvist, Juuso Jokiniemi, Johan Lilius, Ivan Porres, Framework for Smart Space Application Development. In: Salvatore Flavio Pileggi (Ed.), *International Workshop*
- Atzori, L., Iera, A. and Morabito, G. The Internet of Things: A survey. *Computer Networks* 54, 15 (2010), 2787-2805.
- Augustin, I., Yamin, A., Geyer, C.F.R. (2004). Isam, joining context-awareness and mobility to building pervasive applications. *Mobile Computing Handbook*, capítulo 2, p. 73-94. Boca Raton (CRC Press). Artigo convidado.
- Avilés-López, E. and García-Macías, J. TinySOA: a service-oriented architecture for wireless sensor networks. *SOCA* 3, 2 (2009/06/01 2009), 99-108.
- Azizian, M., & Patel, R. (2011). Data fusion for catheter tracking using Kalman filtering: applications in robot-assisted catheter insertion. In K. H. Wong & D. R. Holmes III (Eds.), *SPIE Medical Imaging* (pp. 796413–796413–11). International Society for Optics and Photonics. doi:10.1117/12.878148
- Barbara Hohlt, Lance Doherty, and Eric Brewer. 2004. Flexible power scheduling for sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks (IPSN '04)*. ACM, New York, NY, USA, 205-214

- Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., PRATT, I. and WARFIELD, A. Xen and the art of virtualization, ACM, Bolton Landing, NY, USA,(2003).
- Bayes, Mr.; Price, M. (1763). An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S.: Bayes, M.: Free Download & Streaming: Internet Archive. Philosophical Transactions. Retrieved May 2, 2013, from <http://archive.org/details/philtrans09948070>
- Bell, M.: ‘SOA Modeling Patterns for Service Oriented Discovery and Analysis’ (Wiley Publishing, 2010).
- Bhattacharya, S., Saifullah, A., Lu, C., & Roman, G.-C. (2010). Multi-Application Deployment in Shared Sensor Networks Based on Quality of Monitoring. 2010 16th IEEE Real-Time and Embedded Technology and Applications Symposium, 259-268. Ieee. doi:10.1109/RTAS.2010.20
- Biyabani, A. (2009). Infrastructure Monitoring Smart Wireless Sensor Network with Solar Energy Harvesting. Science.
- Blasch, E.P.; Lambert, D.A.; Valin, P.; Kokar, M.M.; Llinas, J.; Das, S.; Chee Chong; Shahbazian, E., "High Level Information Fusion (HLIF): Survey of models, issues, and grand challenges," Aerospace and Electronic Systems Magazine, IEEE , vol.27, no.9, pp.4,20, September 2012, doi: 10.1109/MAES.2012.6366088
- Brouwers, N., Langendoen, K. and Corke, P. Darjeeling, a feature-rich VM for the resource poor, ACM, Berkeley, California,(2009).
- Brown, C., Durrant-Whyte, H., Leonard, J., Rao, B., and Steer, B. 1992. Distributed data fusion using Kalman filtering: A robotics application. In Data Fusion in Robotics and Machine Intelligence, M. A. Abidi and R. C. Gonzalez, Eds. Academic Press, Inc., San Diego, CA, Chapter 7, 267–309.
- Buckl, C., *et al.* (2009) Services to the Field: An Approach for Resource Constrained Sensor/Actor Networks. International Conference on Advanced Information Networking and Applications Workshops, p. 476–481.

- Bui, V. T., Verhoeven, R. and Lukkien, J. J. A Body Sensor Platform for concurrent applications. (3-5 Sept. 2012 2012), 38-42.
- Caracas, A., Kramp, T., Baentsch, M., Oestreicher, M., Eirich, T. and Romanov, I. Mote Runner: A Multi-language Virtual Machine for Small Embedded Devices. (18-23 June 2009 2009), 117-125.
- Cheng, B.C., Lemos, R., Giese, H., Inverardi, P., Magee, J., Andersson, J., Becker, B., Bencomo, N., Brun, Y., Cukic, B., Marzo Serugendo, G., Dustdar, S., Finkelstein, A., Gacek, C., Geihs, K., Grassi, V., Karsai, G., Kienle, H., Kramer, J., Litoiu, M., Malek, S., Mirandola, R., Müller, H., Park, S., Shaw, M., Tichy, M., Tivoli, M., Weyns, D., and Whittle, J.: ‘Software Engineering for Self-Adaptive Systems: A Research Roadmap’, in Cheng, B.C., Lemos, R., Giese, H., Inverardi, P., and Magee, J. (Eds.): ‘Software Engineering for Self-Adaptive Systems’ (Springer Berlin Heidelberg, 2009), pp. 1-26.
- Chowdhury, N. M. M. K. and Boutaba, R. Network virtualization: state of the art and research challenges. *Communications Magazine, IEEE* 47, 7 (2009), 20-26.
- Christos Efstratiou, Ilias Leontiadis, Marco Picone, Kiran K. Rachuri, Cecilia Mascolo, and Jon Crowcroft. 2012. Sense and sensibility in a pervasive world. In *Proceedings of the 10th international conference on Pervasive Computing (Pervasive'12)*, Judy Kay, Paul Lukowicz, Hideyuki Tokuda, Patrick Olivier, and Antonio Krüger (Eds.). Springer-Verlag, Berlin, Heidelberg
- Crossbow (2010) “Crossbow Technology”, <http://www.xbow.com/>. Accessed in April, 2010.
- Curino, C., Giani, M., Giorgetta, M., Giusti, A., Murphy, A. L. and Picco, G. P. Mobile data collection in sensor networks: The TinyLime middleware. *Pervasive and Mobile Computing* 1, 4 (2005), 446-469.
- Dasarathy, B. V. (1991). Decision fusion strategies in multisensor environments. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(5), 1140–1154. doi:10.1109/21.120065
- de C.Paschoalino, R., and Madeira, E.R.M.: ‘A Scalable Link Quality Routing Protocol for Multi-radio Wireless Mesh Networks’. *Proc. Computer Communications and Networks, 2007. ICCCN 2007. Proceedings of 16th International Conference on*, 13-16 Aug. 2007 pp. 1053-1058.

- de Farias, C., Soares, H., Pirmez, L., Delicato, F., Santos, I., Carmo, L. F.,. A Control and Decision Framework for Smart Buildings. In: 2013 10th IEEE International Conference on Ubiquitous Intelligence and Computing (UIC), 2013, Vietri Sul Mare.
- Delicato, F. C. ; Protti, F. ; Rezende, J. F. ; Pirmez, L. . An Efficient Heuristic for Selecting Active Nodes in Wireless Sensor Networks. *Computer Networks*, v. 50, p. 3701-3720, 2006.
- Delicato, F. C., Pires, P. F., Pirmez, L. and Batista, T. *Wireless Sensor Networks as a Service*, IEEE Computer Society(2010).
- Delicato, F. C., Pires, P. F., Pirmez, L. and Carmo, L. F. A Service Approach for Architecting Application Independent Wireless Sensor Networks. *Cluster Computing* 8, 2-3 (2005), 211-221.
- Delicato, F. C., Pires, P. F., Pirmez, L. and Carmo, L. F. R. D. C. A Flexible Web Service Based Architecture for Wireless Sensor Networks, IEEE Computer Society(2003).
- Denholm, P., Kulcinski, G. L., & Holloway, T. (2005). Emissions and energy efficiency assessment of baseload wind energy systems. *Environmental science & technology*, 39(6), 1903-11. Retrieved from <http://www.ncbi.nlm.nih.gov/pubmed/15819254>
- Dietrich, I. and Dressler, F. (2009) "On the Lifetime of Wireless Sensor Networks", *ACM Transactions on Sensor Networks*, Volume 5, Issue 1 (February 2009).
- Distefano, S., Merlino, G. and Puliafito, A. Enabling the Cloud of Things. (4-6 July 2012 2012), 858-863.
- Dong, J., Zhuang, D., Huang, Y., & Fu, J. (2009). Advances in Multi-Sensor Data Fusion: Algorithms and Applications. *Sensors*, 9(10), 7771-7784. doi:10.3390/s91007771
- Efstratiou, C. Challenges in Supporting Federation of Sensor Networks. (2010).
- Efstratiou, C., Leontiadis, I., Mascolo, C., & Crowcroft, J. (2010). A shared sensor network infrastructure. *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10* (p. 367). New York, New York, USA: ACM Press. doi:10.1145/1869983.1870026
- Electric Power Research Institute (EPRI, 2008), *The Green Grid - Energy Savings and Carbon Emissions Reductions Enabled by a Smart Grid*, Technical Update, Palo Alto, CA.

- Eltarras, R. and Eltoweissy, M. Adaptive Multi-Criteria Routing for Shared Sensor-Actuator Networks. (6-10 Dec. 2010 2010), 1-6.
- EPRI The green grid: Energy savings and carbon emissions reductions enabled by a smart grid. (2008).
- Erol-Kantarci, M. and Mouftah, H. T. Wireless Sensor Networks for smart grid applications. (24-26 April 2011 2011), 1-6.
- Farias, C. M., Pirmez, L., Delicato, F. C., Dos Santos, I. L., & Zomaya, A. Y. (2012). Information fusion techniques applied to Shared Sensor and Actuator Networks. 37th Annual IEEE Conference on Local Computer Networks (pp. 188–191). IEEE. doi:10.1109/LCN.2012.6423603
- Farias, C.M., Pirmez, L., Delicato, F.C., Li, W., Zomaya, A.Y., and de Souza, J.N.: ‘A scheduling algorithm for shared sensor and actuator networks’, in Editor In Proceedings of ICOIN: ‘Book A scheduling algorithm for shared sensor and actuator networks’ (2013, edn.), pp. 648-653.
- Farias, C., Soares, H., Pirmez, L., Delicato, F., Santos, I., Carmo, L. F., de Souza, J., Zomaya, A. and Dohler, M. (2014), A control and decision system for smart buildings using wireless sensor and actuator networks. *Trans Emerging Tel Tech*, 25: 120–135. doi: 10.1002/ett.2791
- Floch, J., Hallsteinsen, S., Stav, E., Eliassen, F., Lund, K., and Gjørven, E.: ‘Using architecture models for runtime adaptability’, *Software, IEEE*, 2006, 23, (2), pp. 62-70.
- Flores-Cortés, C. A., Blair, G. S. and Grace, P. An Adaptive Middleware to Overcome Service Discovery Heterogeneity in Mobile Ad Hoc Environments. *IEEE Distributed Systems Online* 8, 7 (2007), 1.
- Fok, C.-L., Roman, G.-C. and Lu, C. Agilla: A mobile agent middleware for self-adaptive wireless sensor networks. *ACM Trans. Auton. Adapt. Syst.* 4, 3 (2009), 1-26.
- Fok, C.-L., Roman, G.-C., and Lu, C.: ‘Adaptive service provisioning for enhanced energy efficiency and flexibility in wireless sensor networks’, *Science of Computer Programming*, 2013, 78, (2), pp. 195-217.

- Gal, S. A., Oltean, M. N., Brabete, L., Rodean, I., & Opincaru, M. (2011). On-line monitoring of OHL conductor temperature; live-line installation. 2011 IEEE PES 12th International Conference on Transmission and Distribution Construction, Operation and Live-Line Maintenance (ESMO) (pp. 1–6). IEEE. doi:10.1109/TDCCLLM.2011.6042240
- Galli, S., Scaglione, A. and Zhifang, W. For the Grid and Through the Grid: The Role of Power Line Communications in the Smart Grid. *Proceedings of the IEEE* 99, 6 (2011), 998-1027.
- Garlan, D., Shang-Wen, C., An-Cheng, H., Schmerl, B., and Steenkiste, P.: ‘Rainbow: architecture-based self-adaptation with reusable infrastructure’, *Computer*, 2004, 37, (10), pp. 46-54
- Gay, D., Levis, P., Culler, D. and Brewer, E. nesC 1.3 Language Reference Manual, July 2009.
- Geihs, K., Barone, P., Eliassen, F., Floch, J., Fricke, R., Gjørven, E., Hallsteinsen, S., Horn, G., Khan, M.U., Mamelli, A., Papadopoulos, G.A., Paspallis, N., Reichle, R., and Stav, E.: ‘A comprehensive solution for application-level adaptation’, *Software: Practice and Experience*, 2009, 39, (4), pp. 385-422.
- Geyik, S., Szymanski, B., and Zerfos, P.: ‘Robust Dynamic Service Composition in Sensor Networks’, *Services Computing, IEEE Transactions on*, 2012, PP, (99), pp. 1-1.
- Gungor, V. C., Lu, B., & Hancke, G. P. (2009). Opportunities and Challenges of Wireless Sensor Networks in Smart Grid - A Case Study of Link Quality Assessments in Power Distribution Systems. *System*.
- Hassan, M. M., Song, B. and Huh, E.-N. A framework of sensor-cloud integration opportunities and challenges, ACM, Suwon, Korea,(2009).
- Helal, S. and Chen, C. The Gator Tech Smart House: enabling technologies and lessons learned, ACM, Singapore,(2009).
- Herrera, F., and Herrera-Viedma, E.: ‘Linguistic decision analysis: steps for solving decision problems under linguistic information’, *Fuzzy Sets and Systems*, 2000, 115, (1), pp. 67-82.
- Hill, J., Szewczyk, R., Woo, A., Hollar, S., Culler, D. and Pister, K. System architecture directions for networked sensors, ACM, Cambridge, Massachusetts, USA,(2000).

- Hung, K. S., Lee, W. K., Lui, K. S., Yang, G. H., & Zhong, J. (2010). On Wireless Sensors Communication for Overhead Transmission Line Monitoring in Power Delivery Systems, 309–314.
- I. S. Jacobs and C. P. Bean, “Fine particles, thin films and exchange anisotropy,” in *Magnetism*, vol. III, G. T. Rado and H. Suhl, Eds. New York: Academic, 1963, pp. 271–350.
- Ian F. Akyildiz, Tommaso Melodia, Kaushik R. Chowdhury, A survey on wireless multimedia sensor networks, *Computer Networks*, Volume 51, Issue 4, 14 March 2007, Pages 921-960, ISSN 1389-1286.
- Intille, S. S., Larson, K., Beaudin, J. S., Nawyn, J., Tapia, E. M. and Kaushik, P. A living laboratory for the design and evaluation of ubiquitous computing technologies, ACM, Portland, OR, USA,(2005).
- Ishmanov, F., Malik, A. S. and Kim, S. W. Energy consumption balancing (ECB) issues and mechanisms in wireless sensor networks (WSNs): a comprehensive overview. *European Transactions on Telecommunications* 22, 4 (2011), 151-167.
- Islam, M. M., Hassan, M. M., Lee, G.-W. and Huh, E.-N. A Survey on Virtualization of Wireless Sensor Networks. *Sensors* 12, 2 (2012), 2175-2207.
- J. Blythe, S. Jain, E. Deelman, Y. Gil, K. Vahi, A. Mandal, and K. Kennedy. Resource Allocation Strategies for Workflows in Grids In *IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005)*.
- JayasumanA, A. P., Qi, H. and Illangasekare, T. H. Virtual Sensor Networks - A Resource Efficient Approach for Concurrent Applications. (2-4 April 2007 2007), 111-115.
- Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L. S. and Rubenstein, D. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet, ACM, San Jose, California,(2002).
- Kabadayi, S., Pridgen, A. and Julien, C. Virtual Sensors: Abstracting Data from Physical Sensors, *IEEE Computer Society*2006).

- Khaleghi, B., Khamis, A., Karray, F. O., & Razavi, S. N. (2013). Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1), 28–44. doi:10.1016/j.inffus.2011.08.001
- Kientz, J. A., Patel, S. N., Jones, B., Price, E., Mynatt, E. D. and Abowd, G. D. The Georgia Tech aware home, ACM, Florence, Italy,(2008).
- KITCHENHAM, B., PEARL BRERETON, O., BUDGEN, D., TURNER, M., BAILEY, J. and LINKMAN, S. Systematic literature reviews in software engineering – A systematic literature review. *Information and Software Technology* 51, 1 (2009), 7-15.
- Kwon, Y., Sundresh, S., Mechitov, K. and Agha, G. ActorNet: an actor platform for wireless sensor networks, ACM, Hakodate, Japan,(2006).
- Lédeczi, Á., Nádas, A., Völgyesi, P., Balogh, G., Kusy, B., Sallai, J., Pap, G., Dóra, S., Molnár, K., Maróti, M. and Simon, G. Countersniper system for urban warfare. *ACM Trans. Sen. Netw.* 1, 2 (2005), 153-177.
- Leontiadis, I., Efstratiou, C., Mascolo, C. and Crowcroft, Jt. 2012. SenShare: transforming sensor networks into multi-application sensing infrastructures. In *Proceedings of the 9th European conference on Wireless Sensor Networks (EWSN'12)*, Gian Pietro Picco and Wendi Heinzelman (Eds.). Springer-Verlag, Berlin, Heidelberg, 65-81.
- Levis, P. and Culler, D. Maté: a tiny virtual machine for sensor networks, ACM, San Jose, California,(2002).
- Levis, P., Madden, S., Polastre, J., Szewczyk, R., Whitehouse, K., Woo, A., Gay, D., et al. (2005). TinyOS: An Operating System for Sensor Networks. (W. Weber, J. M. Rabaey, & E. Aarts, Eds.) *Ambient Intelligence*, 115–148. doi:10.1007/b138670
- Levis, Philip, Lee, N., Welsh, M., & Culler, D. (2003). TOSSIM. *Proceedings of the first international conference on Embedded networked sensor systems - SenSys '03* (p. 126). New York, New York, USA: ACM Press. doi:10.1145/958491.958506
- Li, W., Delicato, F. C. and Zomaya, A. Y. Adaptive energy-efficient scheduling for hierarchical wireless sensor networks. *ACM Trans. Sen. Netw.* 9, 3 (2013).

- Li, W., Delicato, F.C., Pires, P.F., and Zomaya, A.Y.: 'Energy-efficient three-phase service scheduling heuristic for supporting distributed applications in cyber-physical systems'. Proc. Proceedings of the 15th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems, Paphos, Cyprus, 2012 pp. 229-238.
- Li, W., Delicato, F.C., Pires, P.F., Lee, Y.C., Zomaya, A.Y., Miceli, C., and Pirmez, L.: 'Efficient allocation of resources in multiple heterogeneous Wireless Sensor Networks', Journal of Parallel and Distributed Computing, 2014, 74, (1), pp. 1775-1788.
- Li, W., F.C. Delicato, and A.Y. Zomaya, Adaptive energy-efficient scheduling for hierarchical wireless sensor networks. ACM Trans. Sen. Netw., 2013. 9(4).
- Lillegraven, T. N. and Wolden, A. C. Design of a Bayesian recommender system for tourists presenting a solution to the cold-start user problem, Norwegian University of Science and Technology(2010).
- Liu, M., et al. (2009) SmartCIS: Integrating Digital and Physical Environments. In SIGMOD '09: Proceedings of the 35th SIGMOD International Conference on Management of Data, p. 1111–1114. ACM Press.
- Liu, T. and Martonosi, M. Impala: a middleware system for managing autonomic, parallel sensor systems, ACM, San Diego, California, USA,(2003).
- M. Wiczorek, R. Prodan and T. Fahringer. Scheduling of Scientific Workflows in the ASKALON Grid Environment. In SIGMOD Record, volume 34(3), September 2005.
- Mahajan, S., & Malhotra, J. (2011). Energy Efficient Control Strategies in Heterogeneous Wireless Sensor Networks: A Survey. International Journal of Computer ..., 14(6), 31–37. Retrieved from http://www.researchgate.net/publication/50946239_Energy_Efficient_Control_Strategies_in_Heterogeneous_Wireless_Sensor_Networks_A_Survey/file/32bfe50ff5769d331c.pdf
- Mainwaring, A., Culler, D., Polastre, J., Szewczyk, R. and Anderson, J. Wireless sensor networks for habitat monitoring, ACM, Atlanta, Georgia, USA,(2002).
- Manjunatha, P.; Verma, A.K.; Srividya, A., "Multi-Sensor Data Fusion in Cluster based Wireless Sensor Networks Using Fuzzy Logic Method," Industrial and Information Systems, 2008.

- ICIIS 2008. IEEE Region 10 and the Third international Conference on , vol., no., pp.1,6, 8-10 Dec. 2008
- Martinez, K., Hart, J. K. and Ong, R. Environmental Sensor Networks. *Computer* 37, 8 (2004), 50-56.
- Marzullo, K. 1984. Maintaining the time in a distributed system: An example of a loosely-coupled distributed service. Ph.D. thesis, Stanford University, Department of Electrical Engineering, Stanford, CA.
- Melodia, T., & Akyildiz, I. (2010). Cross-layer QoS-aware communication for ultra wide band wireless multimedia sensor networks. *IEEE Journal on Selected Areas in Communications*, 28(5), 653–663. doi:10.1109/JSAC.2010.100604
- Meyer, S. and Rakotonirainy, A. A survey of research on context-aware homes, Australian Computer Society, Inc., Adelaide, Australia,(2003).
- Moher, D., Liberati, A., Tetzlaff, J. and Altman, D. G. Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. *Annals of Internal Medicine* 151, 4 (2009), 264-269.
- Mottola, L. and Picco, G. P. Programming wireless sensor networks: Fundamental concepts and state of the art. *ACM Comput. Surv.* 43, 3 (2011), 1-51.
- Mouftah, H. (2010). Wireless Sensor Networks for Smart Grid Applications. *Engineering*, 1–6. Retrieved from http://www.zu.edu.eg/AEAS/%D8%A7%D8%A8%D8%AD%D8%A7%D8%AB/WIRELESS_SENSOR_NETWORKS_FOR_SMART_GRID_APPLICATIONS.pdf
- Nakamura, E. F., Loureiro, A. a F., & Frery, A. C. (2007). Information fusion for wireless sensor networks. *ACM Computing Surveys*, 39(3), 9-es. doi:10.1145/1267070.1267073
- Nirmalya, R., Rajamani, V. and Julien, C. Supporting multi-fidelity-aware concurrent applications in dynamic sensor networks. (March 29 2010-April 2 2010), 43-49.
- Noh, A.S.-I.; Lee, W.J.; Ye, J.Y., "Comparison of the Mechanisms of the Zigbee's Indoor Localization Algorithm," *Software Engineering, Artificial Intelligence, Networking, and*

- Parallel/Distributed Computing, 2008. SNPD '08. Ninth ACIS International Conference on , vol., no., pp.13,18, 6-8 Aug. 2008.
- P. Rosakis, A.J. Rosakis, G. Ravichandran, J. Hodowany, A thermodynamic internal variable model for the partition of plastic work into heat and stored energy in metals, *Journal of the Mechanics and Physics of Solids*, Volume 48, Issue 3, March 2000, Pages 581-607, ISSN 0022-5096, 10.1016/S0022-5096(99)00048-4.
- Papazoglou, M. P., Traverso, P., Dustdar, S. and Leymann, F. Service-Oriented Computing: State of the Art and Research Challenges. *Computer* 40, 11 (2007), 38-45.
- Parwekar, P. From Internet of Things towards cloud of things. (15-17 Sept. 2011 2011), 329-333.
- Paschoalino, R.; Madeira, E.R.M., “A Scalable Link Quality Routing Protocol for Multiradio Wireless Mesh Networks”. *Proc. of WiMan 2007*, Honolulu, 2007.
- Provan, G. M. (1992). The validity of Dempster-Shafer belief functions. *International Journal of Approximate Reasoning*, 6(3), 389–399. doi:10.1016/0888-613X(92)90032-U
- Rao, J., and Su, X.: ‘A Survey of Automated Web Service Composition Methods’, in Cardoso, J., and Sheth, A. (Eds.): ‘Semantic Web Services and Web Process Composition’ (Springer Berlin Heidelberg, 2005), pp. 43-54.
- Rocha, A. R., Pirmez, L., Delicato, F. C., Lemos, É., Santos, I., Gomes, D. G. and De Souza, J. N. WSNs clustering based on semantic neighborhood relationships. *Computer Networks* 56, 5 (2012), 1627-1645.
- Rouvoy, R., Eliassen, F., Floch, J., Hallsteinsen, S., and Stav, E.: ‘Composing Components and Services Using a Planning-Based Adaptation Middleware’, in Pautasso, C., and Tanter, É. (Eds.): ‘Software Composition’ (Springer Berlin Heidelberg, 2008), pp. 52-67.
- Rowaihy, H., Johnson, M.P., Liu, O., Bar-Noy, A., Brown, T., and Porta, T.L.: ‘Sensor-mission assignment in wireless sensor networks’, *ACM Trans. Sen. Netw.*, 2010, 6, (4), pp. 1-33.
- S. Xiong, J. Li, M. Li, J. Wang, and Y. Liu, “Multiple Task Scheduling for Low-Duty-Cycled Wireless Sensor Networks,” in *INFOCOM '11*.

- Sakellariou, R. e Zhao, H. A Hybrid Heuristic for DAG Scheduling on Heterogeneous Systems. In Proceedings of 13th Heterogeneous Computing Workshop (HCW 2004), 26-30, April 2004, Santa Fe, New Mexico, USA.
- Schlapfer, M., & Mancarella, P. (2011). Probabilistic Modeling and Simulation of Transmission Line Temperatures Under Fluctuating Power Flows. *IEEE Transactions on Power Delivery*, 26(4), 2235–2243. doi:10.1109/TPWRD.2011.2145394
- Schor, L., Sommer, P. and Wattenhofer, R. (2009) Towards a Zero-Configuration Wireless Sensor Network Architecture for Smart Buildings. Proceedings of the First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, p. 31–36. ACM Press.
- Shah, S. Y. and Szymanski, B. K. Dynamic Multipath Routing of Multi-Priority Traffic in Wireless Sensor Networks. (2012).
- Sharma, N. K., Irwin, D. E. and Shenoy, P. J. VSense: Virtualizing Stateful Sensors With Actuators. (2009).
- Shuguang, X., Jianzhong, L., Zhenjiang, L., Jiliang, W., and Yunhao, L.: ‘Multiple task scheduling for low-duty-cycled wireless sensor networks’. Proc. INFOCOM, 2011 Proceedings IEEE, 10-15 April 2011 pp. 1323-1331.
- Silva, T. H., Vaz de Melo, P. O. S., Almeida, J. M. de, & Loureiro, A. A. F. (2012). Uncovering properties in participatory sensor networks. Proceedings of the 4th ACM international workshop on Hot topics in planet-scale measurement - HotPlanet '12 (p. 33). New York, New York, USA: ACM Press. doi:10.1145/2307836.2307847
- Simon, D., Cifuentes, C., Cleal, D., Daniels, J. and White, D. Java™ on the bare metal of wireless sensor devices: the squawk Java virtual machine, ACM, Ottawa, Ontario, Canada,(2006).
- Siu-Nam, C., and Chan, A.T.S.: ‘Dynamic QoS Adaptation for Mobile Middleware’, *Software Engineering*, *IEEE Transactions on*, 2008, 34, (6), pp. 738-752.
- Smith, S. W. 1999. *The Scientist and Engineer’s Guide to Digital Signal Processing*, 2nd ed. California Technical Publishing, San Diego, CA.
- Snoonian, D. Control systems: smart buildings. *IEEE Spectr.* 40, 8 (2003), 18-23.

- Steffan, J., Fiege, L., Cilia, M. and Buchmann, A. Towards Multi-Purpose Wireless Sensor Networks, IEEE Computer Society(2005).
- Soares, H. R. S. ; Pirmez L. ; Delicato, F. C. ; Farias, C. M. . CONDE: Um Sistema de Controle e Decisão para Edifícios Inteligentes usando Redes de Sensores e Atuadores Sem Fio. In: Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos, 2012, Ouro Preto. Anais do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre: SBC, 2012.
- Sugihara, R. and Gupta, R. K. Programming models for sensor networks: A survey. ACM Trans. Sen. Netw. 4, 2 (2008), 1-29.
- Tapia, D. I., Bajo, J., & Corchado, M. (2010). Wireless Sensor Networks for Data Acquisition and Information Fusion: A Case Study.
- Ting Zhu; Mohaisen, A.; Yi Ping; Towsley, D.; , "DEOS: Dynamic energy-oriented scheduling for sustainable wireless sensor networks," INFOCOM, 2012 Proceedings IEEE , vol., no., pp.2363-2371, 25-30 March 2012
- Tsai, K.-C., Sung, J.-T., & Jin, M.-H. (2008). An Environment Sensor Fusion Application on Smart Building Skins. 2008 IEEE International Conference on Sensor Networks Ubiquitous and Trustworthy Computing sutc 2008, 078(2), 291-295.
- Umesh Singh, Amarendra K. Singh, S. Parvez, and Anand Sivasubramaniam. 2010. CFD-based operational thermal efficiency improvement of a production data center. In Proceedings of the First USENIX conference on Sustainable information technology (SustainIT'10). USENIX Association, Berkeley, CA, USA, 6-6.
- Vijay, G., Ben Ali Bdira, E. and ibnkahla, M. Cognition in Wireless Sensor Networks: A Perspective. Sensors Journal, IEEE 11, 3 (2011), 582-592.
- Voinescu, A.; Tudose, D.S.; Tapus, N., "Task Scheduling in Wireless Sensor Networks," Networking and Services (ICNS), 2010 Sixth International Conference on , vol., no., pp.12-17, 7-13 March 2010
- Wagner, A., Speiser, S., & Harth, A. (2010). Semantic Web Technologies for a Smart Energy Grid: Requirements and Challenges. Communication, 1–4.

- Wang, J. et al. (2004) "A sensor-fusion approach for meeting detection" In: Workshop on Context Awareness at Int. Conf. on Mobile Systems, App., and Services (MobiSys), Boston, June 2004.
- WATER RESOURCES RESEARCH, VOL. 21, NO. 6, PP. 837-846, 1985
doi:10.1029/WR021i006p00837
- Wei, L. and R. VESILO. Modeling of Session Persistence in Web Server Systems. in Australian Telecommunications Networks and Application Conference. 2006. Melbourne.
- Wilde, E.; Guinard, D.; e Trifa, V. Architecting a Mashable Open World Wide Web of Things, Institute for Pervasive Computing, ETH Zürich, Zürich, Switzerland, No. 663, February 2010.
- Wu, C., Xu, Y., Chen, Y. and Lu, C. Submodular game for distributed application allocation in shared sensor networks. (25-30 March 2012 2012), 127-135.
- Wu, F., Kao, Y., Tseng, Y. From wireless sensor networks towards cyber physical systems, Pervasive and Mobile Computing, Volume 7, Issue 4, August 2011, Pages 397-413, ISSN 1574-11
- Wu, J., Yuan, S., Ji, S., Zhou, G., Wang, Y., & Wang, Z. (2010). Multi-agent system design and evaluation for collaborative wireless sensor network in large structure health monitoring. *Expert Systems with Applications*, 37(3), 2028–2036. doi:10.1016/j.eswa.2009.06.098
- XIE, T. and QIN, X. An Energy-Delay Tunable Task Allocation Strategy for Collaborative Applications in Networked Embedded Systems. *Computers, IEEE Transactions on* 57, 3 (2008), 329-343.
- Xiong, N. (2002). Multi-sensor management for information fusion: issues and approaches. *Information Fusion*, 3(2), 163-186. doi:10.1016/S1566-2535(02)00055-6
- Xu, N., Rangwala, S., chintalapudi, K. K., Ganesan, D., Broad, A., Govindan, R. and Estrin, D. A wireless sensor network For structural monitoring, ACM, Baltimore, MD, USA,(2004).
- Xu, Y., Chen, Y., & Lu, C. (2012). Submodular game for distributed application allocation in shared sensor networks. 2012 Proceedings IEEE INFOCOM (pp. 127–135). IEEE. doi:10.1109/INFCOM.2012.6195490

- Xu, Y., Saifullah, A., Chen, Y., Lu, C., and Bhattacharya, S.: 'Near optimal multi-application allocation in shared sensor networks'. Proc. Proceedings of the eleventh ACM international symposium on Mobile ad hoc networking and computing, Chicago, Illinois, USA, 2010 pp. 181-190.
- Yong, D., Neumann, M. A., Gordon, D., Riedel, T., Miyaki, T., Beigl, M., Wenzhu, Z. and Lin, Z. A Platform-as-a-Service for in-situ development of wireless sensor network applications. (11-14 June 2012 2012), 1-8.
- Yu, Y. and Prasanna, V. K. Energy-balanced task allocation for collaborative processing in wireless sensor networks. *Mob. Netw. Appl.* 10, 1-2 (2005), 115-131.
- Yu, Y., Rittle, L. J., Bhandari, V. and Lebrun, J. B. Supporting concurrent applications in wireless sensor networks, ACM, Boulder, Colorado, USA,(2006).
- Yuan Tian; Ekici, E.; Ozguner, F.; , "Energy-constrained task mapping and scheduling in wireless sensor networks," *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on* , vol., no., pp.8 pp.-218, 7-7 Nov. 2005
- Z. Henan and R. Sakellariou, "Scheduling multiple DAGs onto heterogeneous systems," in *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International, 2006*, p. 14 pp.
- Zhang, R. and Zhang, Y. "LR-Seluge: Loss-Resilient and Secure Code Dissemination in Wireless Sensor Networks," *icdcs*, pp.497-506, 2011 31st International Conference on Distributed Computing Systems, 2011
- Zomaya, A. Y. *Parallel and distributed computing handbook*, McGraw-Hill, Inc.1996.

