

# Adaptable resource management for WSNs

*Pedro Javier del Cid,  
Sam Michiels,  
Wouter Joosen*

*Report CW 561, August 2009*



Katholieke Universiteit Leuven  
Department of Computer Science

Celestijnenlaan 200A – B-3001 Heverlee (Belgium)

# Adaptable resource management for WSNs

*Pedro Javier del Cid,  
Sam Michiels,  
Wouter Joosen*

*Report CW 561, August 2009*

Department of Computer Science, K.U.Leuven

## **Abstract**

Wireless sensor networks (WSNs) are currently being deployed in an application specific manner. We believe that with the purpose of exploiting the full potential of these networks and to maximize the return on investment they can represent for the business industry, we must now think in terms of a lightweight highly distributed and mobile service platform. Although application specific solutions that address resource management are effectively being used, we believe current work in resource management is lacking the appropriate mechanisms to support “the service platform” approach. More resource management support is needed to ease application development for this more dynamic approach to deploy WSN applications. In this context: multiple concurrent applications and dynamically changing business requirements will require a more dynamic way to allocate services. Resource management solutions need to be able to manage requests for service from local or external concurrent applications and adapt to changing business requirements and system conditions. In this paper we contribute by outlining the key requirements resource management solutions need to address in order to provide support for these approaches. We discuss and review to what extent current state of the art addresses them and finally we present a conceptual architecture that will allow application developers to more easily realize these dynamic approaches

# Adaptable Resource Management for WSNs

Pedro Javier del Cid, Sam Michiels, Wouter Joosen

IBBT-Distrinet Research Group

Katholieke Universiteit Leuven

Leuven, Belgium

{javier.delcid, sam.michiels}@cs.kuleuven.be

**Abstract**—Wireless sensor networks (WSNs) are currently being deployed in an application specific manner. We believe that with the purpose of exploiting the full potential of these networks and to maximize the return on investment they can represent for the business industry, we must now think in terms of a lightweight highly distributed and mobile service platform. Although application specific solutions that address resource management are effectively being used, we believe current work in resource management is lacking the appropriate mechanisms to support “the service platform” approach. More resource management support is needed to ease application development for this more dynamic approach to deploy WSN applications. In this context: multiple concurrent applications and dynamically changing business requirements will require a more dynamic way to allocate services. Resource management solutions need to be able to manage requests for service from local or external concurrent applications and adapt to changing business requirements and system conditions. In this paper we contribute by outlining the key requirements resource management solutions need to address in order to provide support for these approaches. We discuss and review to what extent current state of the art addresses them and finally we present a conceptual architecture that will allow application developers to more easily realize these dynamic approaches.

**Keywords**—component; wireless sensor networks, resource management, middleware, dynamic adaptation, service to task allocation, context aware.

## I. INTRODUCTION

Wireless sensor networks (WSNs) have emerged as one of the key technologies to enable pervasive computing environments [1]. These systems will span from the physical environment, personal devices to traditional backend business infrastructure. In this context the WSNs will represent a mobile, resource constraint service platform with the capacity to sense the physical environment and provide a variety of services to its users. In this vision we have to acknowledge that it is not possible to know in advance exactly what services each node will provide and to what applications it will provide them to.

Conventional approaches utilize WSNs in a rather static pattern [2], usually involving: data collection, performing minimal on board processing and forwarding events to the central system. In these conventional approaches it was assumed that a WSN would be used exclusively by one application for example sense ambient temperature for an air conditioning application [3]. While only being used by one application, they were designed to be able to optimize a single

system wide global goal, for example prolonging sensor network lifetime [4], maximizing system lifetime [5] or maximizing a measure of total information extracted from the system [6].

In our new perspective, as previously mentioned, we consider deploying WSNs that will provide services to a variable number of applications. The WSNs will no longer have global system wide goals to optimize. Each node on the WSN can run multiple applications and provide services transparently to neighboring nodes or back end applications. This is not an entirely new concept, the idea that multiple applications would be using a WSN has been proposed in many papers like [7][8][9][10]. We believe a separation of concerns between application logic and WSN resource management is essential to complying with non functional requirements in such a dynamic environment.

In order to help simplify the development of applications designed to be used in the presented context, resource management middleware (RMM) needs to be able to allocate services to applications in a dynamic way for concurrently running applications. Allocated services can be local or external and dynamic adaptation of allocations based on context should be supported. The re allocation of services, that have become unavailable due to malfunction, unreliable network conditions or high node mobility should also be supported. A binding model that allows separation of concerns between application logic and resource management is needed.

The main purpose of middleware in WSNs is to support the development, deployment, maintenance and execution of applications [11]. In this paper we contribute by outlining the key requirements RMM needs to address in order to provide support for these approaches. We discuss how these requirements are not supported in existing approaches to resource management for WSNs and present a conceptual architecture that provides the first steps toward a solution.

In section II we define useful terminology and present a case study. In section III we outline and describe the key requirements RMM needs to address. In section IV we present current state of the art. In section V we present the conceptual architecture of our RMM, and then we present how this architecture addresses the issues presented section III. In section VI we present our ideas on future work and conclude the paper.

## II. TERMINOLOGY AND CASE STUDY

### A. Terminology

For the sake of clarity we first define some terminology used in this paper as to avoid unnecessary confusion. *Resource management*: the allocation of services to applications and the dynamic reallocation of those allocations based on context. *Node or mote*: a sensor node in a WSN that is capable of performing on board processing, gathering sensory information and communicating to other nodes in the network via radio. *Sensor*: the hardware device on a node that performs the sensing action. *Services*: in the WSN are for example the temperature sensing or light detection provided by a node. *Application in the WSN*: A program installed on a node that processes information locally. For example the program that receives information from a body temperature sensor and blood pressure sensor and processes the raw data and can derive possible emerging health conditions that will require emergency assistance. *Local services or applications*: services or applications running on the same node as the resource management middleware. *Remote services or applications*: services or applications running on a different node as the resource management middleware.

### B. The Meds case study

This case study illustrates how a WSN can be deployed as a lightweight service platform, i.e. in a way that its nodes will provide multiple services to concurrent applications. A WSN is deployed at the “Meds” hospital, on the monitored patients and in all the ambulances. The WSN in the hospital is to be used initially for an HVAC [3] application, inventory management application [12], an emergency response application [14] and a health monitoring application [13]. In the ambulance it will be used by the inventory management, emergency response and health monitoring service. To illustrate the requirements that will be demanded from a node you can think of a wheel chair in that hospital. It is tracked by the inventory application at all time inside the hospital and on the ambulances. When transporting a monitored patient, the sensors on the patient can use the positioning service on the wheel chair and outsource the processing of data to save energy. The temperature sensor on the wheelchair is used by HVAC application to gather ambient temperature. When the wheel chair is in an ambulance it can turn off its positioning service and use the one in the ambulance. When a dangerous gas leak is detected in a hospital wing, the emergency response application can know if the wheel chair is available to be used to relocate patients to a safe area. It is important to acknowledge that not all services should be acquired externally even if available in a neighboring node. Imagine two monitored patients which both have heart rate and body temperature monitors, you would not think to accept those services from the other patient’s sensors. It is also logical to assume that since the range of the radio transmitters on nodes is relatively short, we can assume that using the positioning service or temperature service from a neighbor one hop away would be acceptable. This refers to spatial accuracy of the service.

## III. PROBLEM OUTLINE

When you plan to use the WSN as a service platform it is necessary to take into account the business applications that will be using the services provided by the WSN. In doing so we found it advantageous to classify the different types of application approaches we expect to have to support.

### A. Application approaches that need to be supported.

In [15] the authors propose the following classification: (1) event driven approach, the application stays passive until an event of interest occurs, for example the fire monitoring application in the hospital, that only requires to know if a smoke alarm has been triggered. (2) Data driven approach, the sensor information is actively requested by the application, for example the health monitoring application that requires constant monitoring of a patient’s vital signs. (3) Application driven approach, the integration of business functionalities or outsourcing to the WSNs, for example: In the health monitoring application, it is required that the data obtained by the body sensors be processed locally. This processing is meant to help detect medical conditions that could require emergency response, e.g. when the combination of high blood pressure and an elevated heart rate could indicate the existence of a serious condition. Due to the nature of the emergency it is not ideal to transmit the sensor’s raw data to a central processing system in order to detect these medical conditions. You are interested in having that data processed locally to provide a much faster response to an emerging medical condition. These 3 approaches do not have to be considered separately, many applications will demand a combination or all of them, as clearly illustrated in our case study.

### B. Key Requirements

**Support for multiple allocation types:** From the previous classification we can conclude that the RMM will need to support the allocation of services requested in the following manners: (1) event based: that are requested in an event based manner. (2) Continuous: The use of a service in a continuous manner, i.e. given a specified sampling frequency. (3) For applications: The allocation of local and external services to concurrent applications.

**Supporting requests from multiple applications for the same service:** In order to be able to allocate the same service to multiple applications, services must be decoupled from any application.

**Service discovery, matching and binding:** Since applications and services are developed independently, there is a need for service descriptions. Service descriptions should be lightweight and semantically correct to ensure that the behavior is predictable. They also need to contain QoS attributes that enable the application to specify spatial and temporal accuracy for each service they request. In order to separate application logic from resource management a binding model is needed that enables this separation. The matching process between service requests and available services must be lightweight and not impose significant overhead.

**Supporting concurrent requests and wireless code update mechanisms:** due to dynamic environment there is a

need to support concurrent service requests and runtime code updates. Concurrency models that support multiple applications running in parallel, dynamic loading of programs and runtime software reconfiguration have all been thoroughly examined in [16] and for the sake of brevity are not discussed further on this paper.

**RM has an important role in supporting non functional requirements:** (1) Energy and resource constraints: The utilized mechanisms in WSNs have to be as energy efficient as possible to maximize system lifetime [17]. (2) Adaptability: Due to the dynamic and resource constraint nature of the WSNs, the need for proactive adaptation of resources has been repeatedly advocated [18][19]. For these reasons RMM needs to be able to re allocate services based on the systems changing conditions and contextual information. (3) Scalability: Due to the distributed nature of WSN applications, middleware solutions need to be scalable [20]. (4) Heterogeneity: we expect to have to support two types. (i) Varying device capabilities and (ii) varying run time environments.

#### IV. RELATED WORK

The term resource management in WSNs has been loosely used to refer to a wide variety of different functionalities. There is no widely accepted definition or set of functionalities that accurately describe resource management. Very different approaches have been used, so we consider advantageous to classify the work based on the taxonomy presented in [21]. We first briefly describe the taxonomy and then classify some of the previous work done in resource management for WSNs and review how they support the requirements in section III.

##### A. A Taxonomy for programming approaches in WSNs.

The taxonomy classifies issues related to programming sensor networks in two broad classes: programming support and programming abstractions. The first one is concerned with providing systems, services and runtime mechanisms. The second one is concerned with the way we view sensor networks and provides abstractions of sensor nodes and data. In the programming support class they identified five main subclasses: virtual machine based, modular (agent) based, database based, application driven and message oriented middleware. In the programming abstractions class they identified two main subclasses: the first focuses on global behavior of the network, macro programming based and the second deals with local behavior of the nodes in distributed computation.

##### B. Programming support class.

In the programming support class many middleware solutions that address resource management issues have been developed, we will mention just a few of the more popular ones: Servilla [22] which offers support for the discovery and binding to local and remote services using a service oriented computing approach. Servilla has provided the basis for our service base and selection components, which are described in section V. Impala [23] which enables application adaption on the basis of different scenarios like energy efficiency and can identify needed parameter settings, Milan[24] can adjust

network characteristics to maximize network lifetime and respect application specified quality needs, Autosec [25] which provides support for dynamic service brokering that allows better resource use and application quality of services requirements can be translated into the underlying system level resources and finally Davim [16] which allows to customize and extend sensor behavior while also allowing to execute multiple applications in parallel. By decoupling of services and applications they allow for multiple applications to use the same set of services.

The solutions presented have efficient allocation of services in an event based manner but none of them provides specific support for continuous and application based requests. None of them have support for dynamic re allocation of specific services if they were to become unavailable. Furthermore they do not support external and local context sources for dynamic adaptation, service provisioning for external applications or provide service descriptions.

##### C. Programming abstraction class.

In the programming abstraction class there is a lot of interesting work available, for the sake of brevity we will only comment on three approaches. The first one abstract regions [26] focuses on local behavior by providing feedback on the quality of collective operations and exposing an interface for tuning local resource consumption. The second solution, COIN [27] focuses on both local and global perspectives by using distributed independent reinforced learning to adjust local behavior and collective intelligence which optimizes a utility function to adjust global behavior. The third one role based [9] also focuses on both global and local behavior but focusing on application requirements not a predefined optimization problem, the authors present an abstraction which lets the applications specify desired services and expected performance requirements and to list detailed protocol behavior desired from a node executing a role.

These approaches are higher level programming abstractions that provide efficient higher level mappings of network related resources but do not provide the fine grain control needed to solve the problems we are trying to solve.

Previous approaches in abstractions give coarse grained control over WSN resources, generally to the node level. Programming support approaches provide very fine grain control over specific resources on nodes; however we believe that there is a gap between these two approaches. Our work addresses some of the issues un tackled in this gap.

#### V. A CONCEPTUAL RM ARCHITECTURE

##### A. Key concepts

The four key concepts in our approach are the service broker, a job, a service description and the action group. All of these are explained as we present the following two components, after which we describe components that provide supporting functionality. Our architecture (Fig.1) was inspired by the one presented in [28]. It uses indirect loose couple bindings, publish subscribe abstractions and event based communication. The architecture presented here is the full

implementation, only to be installed on gateways. Nodes only required a partial implementation so they can be managed.

(1) **A service broker**, Inspired by the work done in resource management for grids [29][30][31], manages the allocation of services to jobs. In order to optimize the communicational channel and to improve local processing the **job** concept is introduced. A job is a way for RMs to communicate with other RMs. All service requests grouped in a Job are from the same application and target the same location. This location could be a node or location i.e. a warehouse. **Service descriptions** are used to specify key parameters of individual services and provide a flexible way to bind services to applications. To avoid the need of large amounts of metadata we use an approach like the proposed in [22] which divides the specification in functional and non functional properties. They include spatial and temporal accuracy parameters. This indirect binding through the service broker allows for the separation of concerns between application logic and resource management to provide better support for non functional requirements.

(2) Action group control, which keeps track of the action groups. **An action group** can be represented in a k-neighborhood. Action groups are defined to cover two main purposes: (i) one for each service available in the network, which allows you to define the usefulness of external services, for example spatial accuracy: a temperature service 5 hops away may not be representative of the temperature at our location, while a service 2 hops away may be more accurate. These values should be defined network wide for each application specifically. (ii) Limits how many hops away the node is allowed to transmit, when requesting external services, at a given point in time. This value is dynamically adapted with context, for example you can initially request for services from nodes 5 hops away and as your battery level lowers you can reduce the action group to only neighboring nodes 3 hops away. The capability of being able to adjust this setting dynamically based on local network context is very useful because you can very effectively limit the energy spent on transmissions based on current system conditions.

### B. Supporting functionality components

(3) Service base, was inspired by the framework in [22], takes a service specification and finds a matching local service. This component keeps a local service directory and allocations. The service selection does the same for external services. (4) Job control which keeps track of allocated jobs until they have completed successfully. In this way our RMM can detect when a running job requires a new service to be allocated due to a previously allocated service becoming unavailable. (5) Adaptation manager which keeps track of the available context sources. In this component you can register valid context sources local or external. For example: battery level and memory availability. (6) Comms component which handles all communication with local and external applications and other RMMs. (7) Managed resource which monitors the hardware resource, adapts its behavior, provides support to query, keeps track of allowed functioning parameters and provides the resource commit function. Temporal accuracy will be managed

by this component; which describe the sampling frequencies required for a service to be useful for a specific application.

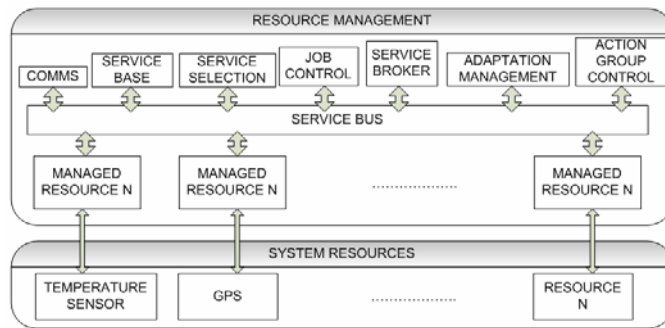


Figure 1. RMM architecture

### C. Description of the support for each key requirement.

**Support for the allocation of services:** (1) event based manner: we consider event based requests to be of two possible types: (i) one shot, e.g. a request for a temperature reading. (ii) Threshold based, e.g. a request for a notification if the temperature exceeds 10 degrees. Both these requirements are supported by the introduction of the job concept, which has been presented in the previous paragraph. (2) Continuous manner, e.g. an application requests temperature readings every five minutes. As explained in the previous paragraph a job provides a way of requesting a set of temperature events without having to re send requests. (3) Application based manner: local or external applications are handled transparently with the use of the service broker which can allocate services dynamically. Each application can specify their own set of requirements for the same service and the service broker manages those allocations.

**Support for multiple applications requesting a service:** The decoupling of a service from an application is achieved with our managed resource component which essentially provides a way of allowing multiple applications access to a service. This managed resource also provides support for external applications requesting a service. Complete service descriptions would be managed by the service broker which will provide a standard way of requesting services and customizing spatial and temporal requirements for each application.

**Concurrency and wireless code update mechanisms:** Our RMM is by no means a substitute or replacement for the runtime support offered by operating systems or virtual machines. We require runtime support and we mention some of the more relevant characteristics: dynamic memory allocation, a safe execution environment to avoid malfunctioning applications from crashing the device, task scheduling control, a concurrency model that supports multiple applications running in parallel, dynamic loading of programs, runtime software reconfiguration. There are solutions that provide these functionalities, we can mention [16] [32].

**Support for non functional requirements:** We provide mechanisms to support these requirements but they cannot be considered as completely solved.

(1) Energy saving mechanisms: (i) dynamically adjusting the amount of hops we can transmit for a service request, given the current system conditions these settings are dynamically adjusted. (ii) Adapting sensor settings: given system conditions, changing the sampling frequency of a temperature sensor can be achieved through the managed resource component. (iv) The use of equivalent external services: given contextual information regarding the existence of for example a positioning service on a neighbor node the system can dynamically allocate that service for a local application.

(2) Adaptability: We provide RM mechanisms that allow the system to adapt: (i) job control which monitors every job and ensures that the service allocated to that job is available until it is completed; otherwise it starts the adaptation process so that an equivalent service can be allocated to be able to accomplish the job. (ii) Action group control: which can be adjusted based on context to adapt the WSN to changing conditions. (iii) Adaptation manager which manages any required change based on system conditions. Previous allocations of local or external services can be changed based on context.

(3) Scalability: Our RMM does not require any global information to operate efficiently and the neighborhood it operates with can be dynamically adjusted through the action group. This allows the WSN to scale without additional communication overhead.

(4) Heterogeneity: we consider two types of heterogeneity: (i) Available resources: nodes in the WSN will range in the resources they have and so we designed this RMM to be component based so that full, partial or no implementations can be made. In full our RMM will handle all the functionalities presented, in partial the node will have the capacity of being managed by a RMM and no implementation will give the node access to network services but none of the resource management functionalities. (ii) Type of node: Our RMM can be implemented to run in different runtime environments and since all the communication is event based this should make the resource management functionalities available for a broad range of devices.

#### D. Further illustration with a use case.

The architecture is further explained by describing how it supports a representative use case that illustrates the above explained complexities. Consider the wheel chair presented in the case study; as it is transporting a patient with health monitoring sensors while on board an ambulance. The positioning service on the wheel chair receives a request for service from the inventory application which is running locally and at the same time it receives a request for service from the patient's sensors:

(1) *Accepting requests for service.* The comms component receives both requests for service. The first task for the comms component is to validate if the requesting applications are trusted. The comms component verifies if the application is already listed in its local lookup table, finding only the local application listed, for the health monitoring application a request of trust is sent to the security middleware (not addressed in this paper) which is in charge of handling security

aspects. The request of trust comes back positive then the requesting application is added to the lookup table and the request for service is inspected to ensure all the required parameters for allocation are included in the message. Both requests for service are validated and sent to the service base component.

(2) *Establishing if there are services available that match the request and allocate the available service.* Once the service base receives the requests for service it checks if the positioning service is available. The service base will check with the positioning service managed resource to confirm availability. Since the service is available, the service base marks the service request as processed and sends the request and selected service to the service broker. The service broker allocates the request and creates a job for the provisioning of the service and notifies job control, which keep track of the job until it is completed successfully. The managed resource is notified so it can provide the resource commit which guarantees the availability of the actual physical resource to provide the mentioned service. Both requests have been successfully allocated both applications can use the service without any interruptions from the other application.

(3) *Once a service is selected and allocated contextual information needs to be monitored in order to identify any needs for reallocation of services.* In order to adapt based on contextual information we use the adaptation management component. The Battery level has been registered as a context source and thresholds have been set. Suppose that after an hour of use the battery level drops under threshold one which indicates that all services provided to external applications must be terminated. The adaptation manager then notifies the service broker that the positioning service is no longer available for external applications. The service broker then notifies the service base and the comms component to send an interruption of service notification to the patient's application which will have to start using its internal service or request service from another node. The local inventory application keeps working uninterrupted.

(4) *Adjusting the sampling frequency of a local service.* Suppose now that the battery level keeps dropping and now it has dropped under threshold 2 which states that all high demand sensors are to be adjusted and only provide readings every 30 minutes. The adaptation manager registers the breach of the battery threshold notifying the managed resource that the sampling frequency must be adjusted and notifying the service broker of the modification. The service broker notifies the local inventory application of the change in sampling frequency for the positioning service. The inventory application keeps running uninterrupted.

(5) *Actions to be taken after a malfunction of the positioning service.* The GPS on the wheelchair malfunctions and the positioning service is no longer available. The managed resource notifies the adaptation management manager about the malfunction. The adaptation manager notifies the service base and the service broker. The service broker de allocates the job and starts a request for external service so a suitable positioning service can be allocated for the local inventory application.

(6) A local application requires a service not available locally. The request for external service is sent to the service selection component. The service selection component checks with the action group control component to see from whom he can request external services. The action group control informs resource selection and then resource selection sends a message to the comms component to send out the request for service to the ambulance's RMM. The procedure for incoming service requests is repeated, trust and validity are established and the comms component forwards all replies to service selection which matches the request with the specified service and sends it to the service base. The service base marks the service request as processed and sends the request and selected service to the service broker. The service broker allocates the task and notifies job control. The inventory application on the wheelchair now has access to the positioning service on the ambulance's sensor.

## VI. CONCLUSIONS AND FUTURE WORK

We presented the key requirements resource management solutions need to address in order to enable applications to use the WSN as a service based platform that can manage and provide services transparently to local or external applications. Additionally we discussed and reviewed to what extent current state of the art addresses them. Finally we presented a conceptual architecture that provides the first steps toward a solution. We are now starting implementation of a proof of concept demonstrator and we are interested in receiving feedback on the presented RMM architecture. Future work will include the support for federated WSN scenarios.

## ACKNOWLEDGMENT

We are very grateful to Christophe Huygens, Nelson Matthys, Klaas Thoelen, Danny Hughes, Rehan Afzal and Wouter Horr  for the interesting discussions and guiding comments on our paper. Research for this paper was partially funded by IMEC and IWT, Instiuit voor de Aanmoeding van innovatie door wetenschap en technologie in Vlaanderen. This research is conducted in the context of the IWT-STADIUM project No. 80037. [33]

## REFERENCES

[1] C. Becker, "Pervasive computing: key technologies and adoption", IEEE PerCom, 2009.

[2] L. Evers, M. Bijl, M. Marin-Perianu, R. Marin-Perianu, P. Havinga, "Wireless sensor networks and beyond: a case study on transport and logistics," University of Twente, Netherlands, May 2005.

[3] N. Ota, S. Ahrens, A. Redfern, X. Yang and P. Wright, "An application driven architecture for residential energy management with wireless sensor networks," University of California, Berkeley, 2006.

[4] C. Fu, B. Wang and H. Beng Lim, "Energy efficient resource management in distributed wireless imaging sensor networks," in proceedings of the ninth International PDCAT, 2008, pp.449-454

[5] H. Zhang and J. Hou, "Maximizing lifetime for wireless sensor networks," University of Illinois at Urbana-Champaign, technical report.

[6] A. Talukder, A. Panangada, T. Herrington, "Autonomous adaptive resource management in sensor network systems for environmental monitoring," IEEEAC paper IEEEAC paper #1348, January 2008.

[7] E. Lupu, N. Dulay, J. Sventek and M. Sloman, "Autonomous pervasive systems and policy challenges of a small world!," in POLICY07,2007.

[8] A. Kupcu, "Secmece: optimizing lifetime of federated sensor networks by exploiting data and model redundancy," Brown University, 2007.

[9] M. Kochhal, L.Schwiebert, S. Gupta, "Role based middleware for sensor networks," technical report, Wayne State University, may 2004.

[10] S. gou, C. Fan and T. Little, "Supporting concurrent task deployment wireless sensor networks," Boston University, USA, 2008.

[11] K. Romer, O. Kasten and F. Mattern, "Middleware challenges for wireless sensor networks," Mobile computing and communications review, volume 6, number 2.

[12] M. McKelvin, M. Williams and N. Berry, "Integrated radio frequency identification and wireless sensor network architecture for automated inventory management and tracking applications," in session systems, ACM, 2005, pp.44-47.

[13] G.Z. Yang, "Body sensor networks," Springer verlag, March 2006.

[14] K. Lorincz, D. Malan, T. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland and M. Welsh, "Sensor networks for emergency response: challenges and opportunities," Pervasive computing, 2004.

[15] L. Gomez, A. Laube, A. Sorniotti, "Design guidelines for integration of wireless sensor networks with Enterprise systems," Mobilware'08, 2008.

[16] S. Michiels, W. Horre, W. Joosen, P. Verbaeten, "DAVIM: A Dynamically Adaptable Virtual Machine for Sensor Networks," Proc. Of MidSens 06, ACM Press, 2006, pp. 7-12.

[17] R. Gemesi, N. Meratnia and P. Havinga, "Quality aware resource management for wireless sensor networks," University of Twente, Enschede, The Netherlands, 2006.

[18] P. Marron, A. Lachenmann, D. Minder, J. Hahner, R. Sauter and K. Rothermel, "Tincubus: a flexible and adaptive framework for sensor networks," In European workshop on wireless sensor networks, 2005.

[19] Y. Yu, B. Krishnamachari and V. Prasanna, "Issues in designing middleware for wireless sensor networks." IEEE network magazine, January 2004.

[20] A. Pandya, H. Luo and G. Pottie, "Scalability of sensor networks," University of California, Los Angeles, United States of America.

[21] S. Hadim, N. Mohamed, "Middleware challenges and approaches for wireless sensor networks," IEEE ds online, v.7, i.3, March 2006.

[22] C. Fok, G. Roman and C. Lu, "Enhanced coordination in sensor networks through flexible service provisioning," Springer Verlag, Coordination 2009, LCNCS 5521, 2009, pp. 66-85

[23] T. Liu and M. Martonosi, "Impala: a middleware system for managing autonomic, parallel sensor systems," Proc. ACM SIGPLAN, 2003.

[24] W. Heinzelman, A. Murphy, H. Carvalho and M. Perillo, "Middleware to support sensor network applications," IEEE network, 18/1,2004.

[25] Q. Han and N. Venkatasubramanian, "Autosec: an integrated middleware framework for dynamic service brokering," IEEE distributed systems online, vol.2 no.7, 2001.

[26] M. Welsh, "Exposing resource tradeoffs in region-based communication abstractions for sensor networks programming sensor networks using abstract regions," ACM SIGCOMM, V.34, N.1, 2004, pp.119-124.

[27] K. Shah, M. Kumar, "Resource management in wireless sensor networks using collective intelligence," In ISSNIP, December 2008, pp.423-428.

[28] N. Matthys and W. Joosen, "Policy based management of sensor networks," Midsense08, ACM 978-1-60558-366-2, 2008.

[29] R. Buyya, S. Chapin and D. DiNuccu, "Architectural models for resource management in the grid," Monash University, Australia.

[30] M. El-Darieby and D. Krishnamurthy, "A scalable wide area grid resource management frameworks," IEEE xplore, 2006.

[31] K. Krauter, R. Buyya and M. Maheswaran, "A taxonomy and survey of grid resource management systems for distributed computing," John Wiley and Sons limited, 2001.

[32] Sun SPOTs. <http://www.sunspotworld.com/>

[33] IWT Stadium project 80037, software technology for adaptable distributed middleware. <http://distrinet.cs.kuleuven.be/projects/stadium/>