PhD Research Proposal

by

Roland Katona

Supervisors: Dr. Donna O'Shea and Dr. Dirk Pesch

March 20, 2015

Abstract

The explosion of technological innovations in the past decades has irreversibly changed our way of life. Cities are evolving and becoming smarter and more interactive as new technologies are added to the existing infrastructure. Small embedded devices with sensing and acting capabilities are contributing to the creation of an additional sensing layer to our habitats with the aim of making them easier and safer to live and work in. Wireless Sensor Networks (WSNs) play a pivotal role in this by providing valuable information to simplify the management and improve the utilization of scarce urban resources.

WSNs, since their inception, were typically designed and deployed as a vertically integrated homogeneous sensing infrastructure developed with a specific application or user in mind. Such systems often suffer from issues such as redundant deployment of similar sensing capabilities within the same sensing boundary resulting in issues such as: low return of investment and resource underutilization. Nonetheless, future WSNs in Smart Cities should be designed in a way where multiple users can exploit the same infrastructure. Heterogeneity of devices/platforms, standards and technologies, however, encumbers the extensive realization of shared WSNs. Therefore, to accomplish such a shared infrastructure, the ownership and management of the sensor network needs to be separated. Additionally, traditional WSN development methods impose numerous fundamental problems. These include difficulties in application programming for the highly resource constrained and heterogeneous nodes/platforms as well as limitations on (re)configure-ability of the nodes/applications and the lack of interoperability amongst the different WSN networks/platforms.

To address these issues and limitations, this PhD research proposes a framework that combines multiple virtualisation methods at the different layers (hardware, system, network, application/service) of the WSN infrastructure. Virtualisation of WSNs creates opportunities where resources and services from several WSNs can be efficiently used by multiple applications and users, bringing with it a host of new challenges and research opportunities. Applying virtualisation to the WSN is expected to simplify network management, accelerate application development and enhance security by abstracting and isolating the underlying resources. Virtualisation can also enable concurrency and facilitate interoperability amongst heterogeneous nodes/platforms by providing a generic execution environment. Therefore the ultimate goal of the proposed framework is to create a dynamic, adaptable and scalable virtualised WSN infrastructure that meets the business objectives of network providers by enabling them to quickly provision and deliver value-added services to various customers.

Contents

1 Research context and contribution to the research field **3**

2 Objectives **4**

2.1 Identified characteristics of the proposed framework 4

3 Review of the state of the art / current literature / current practice of research **7**

3.1 Introduction 7

3.2 Virtualisation 8

3.3 WSN programming 10

3.4 Virtual machines (VMs) 11

3.5 Virtual Sensor (VS) 14

3.6 Cloud computing aproaches 15

3.7 Data Virtualization (DV) 18

3.8 Mobile agent approach 18

3.9 Query Processing Approach 19

3.10 Sensor Network Virtualisation (SNV) 20

3.11 Conclusion 21

4 Expected progress with respect to the state of the art / current literature / current practice of research **23**

5 Research methodology **24**

6 Work plan **25**

7 Ethical issues **26**

1 Research context and contribution to the research field

This PhD research is being carried out as part of SFI project SURF: Service-centric networking for urban-scale feedback systems which aims to exploit information and communications technologies to improve the quality of life of city dwellers around the world. Urban environments have limited resources such as road networks, energy and water. These resources are under increasing strain as a result of population growth. Such resources could be managed in a better way if there was better access to real-time, city- wide information on how the resources are being used.

The SURF project will investigate the design of communications technologies which improve the accessibility of this information for urban services providers. This will allow improved management of city resources, resulting in less traffic, a reduction in water shortages, and fewer power-cuts.

In addition to the traditional communication networks, evolving technologies such as mobile phones, IP/wireless cameras and wireless sensor nodes are added to the existing infrastructure. And as a result, contributing to the creation of an additional sensing layer to human habitats.

In the aforementioned context, this PhD work will concentrate on developing methods that enable existing and new WSN infrastructures to accommodate multiple users and applications concurrently. The proposed research will investigate various virtualisation techniques that can potentially complement each. The resulting framework implements various virtualisation mechanisms to form a virtual ecosystem aiming to address and improve issues associated with the existing technologies. These issues include ease of (re)programmability, manageability, security, performance, resource and constraint management as well as interoperability, deployment/maintenance cost and complexity.

2 Objectives

- Analyse how various virtualisation techniques at the different levels of the WSN infrastructure complement each other and examine the synergy that occurs when combining them.
- Study the impact of virtualisation on the WSN node resources with regards to the potential gain in utilization and the inherent overhead associated with the technique.
- Investigate the benefits of virtualising a sensor network in relation to service provisioning and network management.
- Design and develop a virtualisation ecosystem framework that meets the requirements outlined in section 2.1.
- Evaluate the proposed framework by implementing it within the scope of the SURF project.

2.1 Identified characteristics of the proposed framework

This section identifies and analyses the desirable characteristics of the proposed framework.

High level programming abstractions.

Traditionally, application development for WSNs involves working close to the operating system and solving low level problems. This makes development more difficult and slower. Bridging the gap between the high level application requirements and the low level hardware operations necessitates the availability of sufficient high level programming abstractions.

Ease of (re)configure-ability.

WSN nodes after their deployment tend to be difficult to access therefore it is a non-trivial and in many cases a non-viable task to reconfigure them. Hence an efficient mechanism that advocates over the air (OTA) reprogramming is needed. Furthermore, the energy cost

of radio communication is high and the available bandwidth is low. Therefore a compact data format for optimized packet transmission has to be defined.

(Re)programmability.

The ability to easily implement new user defined functions after deployment -without reloading the entire firmware- is essential in order to respond to changes in the environment or in user requirements. To address this problem, a modular, high level programming scheme is required.

Concurrency.

Support for multi-user and multi-application runtime environment, that accommodates the concurrent execution of multiple application instances is a key requirement to maximize resource utilization.

Abstraction of heterogeneity.

Abstracting the low-level hardware specific details of the heterogeneous WSN nodes (devices with various capabilities potentially from different vendors) and representing them as generic resources is of key importance to assist interoperability and simplifying application development. Additionally, aggregation of resources into a virtual pool offers higher availability and adds extra redundancy to the WSN.

Application/user isolation.

Multiple applications/users concurrently sharing the same resources should not be aware of each other and perceive it as if they were the sole user of the system. Moreover, isolation should provide protection against data corruption by restricting access to other user's/application's resources.

Local dynamic adaptability to environment.

The nodes should not be confined to a set of pre-defined functionality nor should they include application specific functionality. (e.g. support for additional functionality through

extendible modules)

Distributed adaptability.

NEEDS TO BE DEFINED!! The dynamically changing conditions in a WSN necessitate a distributed decision making method which enables applications that are running on individual nodes to adapt to these changes. The ability of the nodes to find optimal solutions collaboratively is an essential ability of future WSNs.

Application resource management (reservation and guarantees).

MAYBE IT COULD BE MERGED WITH THE BELOW POINT.

Constraint management (resource efficiency).

Due to race conditions and risk of over provisioning, proper management and synchronization mechanisms are required for appropriate resource administration. Continuous usage monitoring should be used to allow on-demand up/down scaling of allocated resources.

Mobility and migration (application/VM/service).

Support for transferring an entire application/VM/service is vital to provide QoS (dynamically changing demand) and redundancy (node/link failure). (e.g. support for autonomous and centralized code propagation over the network.

3 Review of the state of the art / current literature / current practice of research

3.1 Introduction

In recent years, the study of WSNs has attracted the interest of many researchers from industry and academia alike. The wide range of practical applications of the technology have been demonstrated in many areas. These include human habitat, [1], health [2] and environmental monitoring [3, 4] as well as industry control [5] and military surveillance [6].

Traditionally, WSNs consist of numerous homogeneous devices, that are programmed for some very specific purpose or application [7, 8], and also owned and controlled by a single organization [9, 10]. WSN nodes can typically be characterized by their physical traits such as power consumption, communication range, memory and storage capacity, computational capability, etc. Additionally, based on their software functions, the nodes perform two different types of operations. One is node-level operations that involve computational tasks and hardware control such as the radio, RAM, CPU and on-board sensors/actuators. The other one is network-level operations which include communication related tasks such as routing. A more detailed and concise classification of wireless sensor nodes is provided in the following publications: [11–13].

Investigators in the field have recognized that the traditional WSN deployment model inherently lacks flexibility and does not facilitate efficiency in resource utilization [14]. In order to address these shortcomings many different research efforts were carried out [15–17]. The common objective of these various approaches was to develop more efficient techniques for creating and managing highly customizable and adaptable WSNs which can accommodate multiple concurrent applications running on heterogeneous nodes.

Virtualisation has been proposed by the WSN research community as a promising solution to address WSN design issues [18] and achieve concurrency, improved efficiency and simplified network and service management. Numerous previous studies have shown how WSNs can benefit from the advantages associated with virtualisation [19] including better resource utilization and scalability, ease of service provisioning and migration as well as im-

proved security by isolation. Moreover, virtualisation offers additional benefits through the abstraction of the underlying hardware and processor architectures. By hiding the heterogeneity of device resource and infrastructure details, and by providing high level programming interfaces to programmers, application development can be simplified and accelerated. Consequently, considering the highly distributed quality of WSNs, a generic virtual system representation may bring further benefits, allowing applications to run transparently over disparate physical processor architectures.

Many viable virtualisation solutions have gained recognition in the general networking [20] and computing [21] domains. More generally the concept of virtualisation is being applied across virtually all sections of a typical IT infrastructure [22]. However, the resource constrained nature of WSNs poses unique challenges that are normally not present in other domains. One of WSNs key characteristic is that the sensor nodes operate from mainly non-renewable battery power. Therefore, extensive research efforts have focused on energy-efficient virtualisation mechanisms to extend battery life, and in turn, prolong operation lifetime [23]. While power consumption is one of the major constraints, there are a number of other limiting factors that are related to WSNs and must be considered when developing a new application. These limiting factors typically include low memory and computational capability as well as the lack of a simple way to maintain and reprogram sensor nodes after deployment.

Over the past years several works have been conducted as an attempt to overcome these difficulties and provide effective virtualisation alternatives to traditional WSN deployments. This section presents a summary of the various approaches to virtualisation in WSNs. The predominant focus of this review is to survey existing research works that have demonstrated how WSNs can leverage from the different virtualisation techniques that have been adopted recently.

3.2 Virtualisation

The notion of virtualisation has been around for decades, since the debut of virtual memory [24] and virtual computer systems [25]. This section provides a brief introduction to virtualisation and discusses the basic characteristics of the technology.

The concept that provides one or more functional environments, which are independent from the underlying resources, is known as virtualisation. It is used to conceal the actual physical traits of computer resources in order to simplify the methods through which those resources can be interacted with. The technology exists in many different and formally uncorrelated shapes, depending on the level of abstraction it provides (e.g. hardware level, instruction set level, system level, process level, library level or application level).

While one form of virtualisation facilitates multiple application instances to concurrently execute in a virtual environment, another form allows the aggregation of hardware resources that are physically distributed across multiple devices into an individual virtual unit. Thus offering load balancing, redundancy and a way to translate heterogeneous hardware resources into a generic virtual resource that in turn can be consumed by various user applications via high-level software interfaces.

At its core, virtualisation simply refers to the decoupling of computation — processing, sensing/actuating, storage and networking — from the resource that performs the action. Virtualisation is typically implemented by applying various techniques such as: hardware/software partitioning, aggregation or composition, time-sharing and indirection or encapsulation [26].

Partitioning: Facilitates the sharing of a single physical resource amongst multiple users or application instances by dividing it into several logical portions (a.k.a hard-partitioning). Through multiplexing, many concurrent applications can consume an individual resource by taking turns to access it(a.k.a. time-sharing).

Aggregation: A technique that makes multiple, typically identical, physical units appear to function as a unified logical unit. Thus, the resulting resource pool can be viewed as a single virtual element.

Composition: A mechanism to combine a number of simpler, typically different, resources into a more complex virtual resource that in turn offers additional functions not provided by any of the original components.

Resource allocation and hardware arbitration are the main enablers of virtualisation, through which the substrate physical computing assets are delineated as software abstractions. Software isolation for fault mitigation and performance management are also fundamental properties of virtualisation.

Virtualisation technology has now matured to the point where it can be used to WSNs. Simplified application development and network management as well as improved resource utilization are the ultimate advantages of employing virtualisation in WSNs. By providing user applications with a logical view of the system, virtualisation offers many additional benefits including increased fault tolerance and improved security through isolation. Greater flexibility, in the shape of application compatibility and portability, are extra benefits that can be attained by abstracting the hardware specific details of WSN nodes with different architectures. By hiding the actual physical environment behind a generic virtual representation, heterogeneous device resources can be automatically mapped to user applications.

3.3 WSN programming

Today's WSN platforms have greatly benefited from the advances in manufacturing technologies. Sensor nodes are becoming smaller, cheaper, more efficient and capable in terms of their processing power. Despite the substantial improvements in hardware there are still significant limitations associated with WSN application development. The literature has identified a significant gap [17, 27–29] which exists between the complexity of the required applications and the lack of sufficient hardware abstractions that would simplify the programming and management of heterogeneous sensor nodes.

Typically, the Operating System (OS) has the central role of providing Application Programming Interfaces (API) for user applications to the underlying hardware resources [30, 31]. Memory and CPU management, low-level resource abstraction and task scheduling for multitasking are some of the main responsibilities of a typical OS. Over the years, important developments have been made to provide light-weight operating systems for WSNs that meet the requirements of diverse and demanding user applications. [32–36]. Due to the unique characteristics of the motes, OS design approaches for WSNs greatly differ from

traditional Operating Systems, hence the realization of the basic OS services is a non-trivial problem. A classification framework, which analyses the different approaches that are used for WSN OS design, is presented in [37]. The survey classifies WSN OSs based on their architecture, execution model, reprogramming ability, scheduling and power management.

In today's WSN deployments application programming involves working close to the OS and solving low-level system problems that requires technical knowledge which is rarely found amongst application programmers. Hence, the need for suitable high-level programming interfaces is apparent and was identified by [12]. The work classifies the various programming approaches that have gained popularity in WSNs, provides a comprehensive comparison amongst the different schemes and highlights the most important design concerns to programming WSN nodes.

Traditionally, WSN application developers create programs that are executed on top of a WSN OS such as Contiki [32] or TinyOS [33], hence exploiting the OS' features (e.g. hardware abstractions, multi-threading, etc). This approach, however, infers that the application has to be bound with the OS into a single firmware image and uploaded to the nodes manually and imposes additional limitations that are related to the particular OS' architecture design (Event driven, Multi-threaded, etc.).

Simplifying application development is arguably the main advantage of accessing hardware resources via high level APIs. In WSNs these are normally provided in a number of different ways. Most commonly an operating system is used to hide operational complexities from user application logic.Apart from OSs, however, other approaches such as virtualisation with the use of Virtual Machines (VM) [27, 38–40] and other middleware [17, 29, 41, 42] have been widely proposed.

3.4 Virtual machines (VMs)

The benefits offered by a VM execution environment include application portability, multi application support and improved hardware utilization. On one hand, applications running in a VM tend to be slower than their native counterparts due to the overhead associated with byte code interpretation. On the other hand, in many cases the overhead is minimal. Nonetheless, benefits of VMs as flexible computational platforms are well known in

mainstream computing[43] and in WSN domains alike [44].

Existing VM approaches for WSNs can greatly vary in how they are implemented. For example Squawk [38], which is a micro edition Java Virtual Machine (JVM) is implemented on the bare metal, while Maté [27], a stack based byte code interpreter is implemented as a single component of TinyOS [33] (an operating system for WSN nodes).

Squawk's core, unlike most VMs, was mainly written in Java that makes it easily portable but currently it is only available for the SunSPOT platform. It implements an application isolation mechanism to allow multiple application instances running concurrently and provides a high level API that enable developers to write wireless applications for WSN in Java.

Maté, on the other hand, runs as a single component of TinyOS. It specifies 24 very compact -one byte long- instructions, 8 of which can be customized for a particular application. In order to reprogram existing WSN deployments, Maté introduces program updates via self-propagating code capsules that are transmitted over the whole network. More complex programs can easily be composed of multiple packets. This research work was one of the first successful attempts to implement a VM on a highly resource constrained WSN node. The developers of Maté identified the need for re-tasking as environmental conditions change and borrowed ideas from Active Networks (AN) architecture in which the execution codes are transmitted via active packets. Concurrency at instruction granularity is achieved through three interleaving execution contexts. Data communication amongst contexts is made possible by a shared variable, a one word heap. Application code running in the VM can be multiplexed with other system tasks, however Maté does not explicitly support multiple concurrent application instances.

The original Maté research was further extended into a framework that enables application developers to generate tailored application specific virtual machines (ASVM) [45] for WSNs. The VMs created by this framework also included an improved code propagation module which was based on the Trickle algorithm [46].

In addition to the ASVM generator other VM generators were developed over the years. One of which was the VMSTAR framework [47] that allows programmers to generate VMs for sensor networks. Apart from the particular application, it also uses hardware specific

details as its foundation to optimize its interpreter to the target platform. Extending application functionality is supported in VMSTAR through incremental linking.

Another VM generator is presented in [48]. The research proposes a framework to create customized stack based VMs based on the hardware capabilities of the devices. For resource constrained sensor nodes only a scaled down version is implemented and the full VM for the more powerful ones where the different versions are fully compatible. This approach aims to assist interoperability amongst heterogeneous devices while exploiting the available performance of the devices with regards to their capabilities.

The research behind Maté was a pioneering work that inspired other researchers and provided a foundation for several VM implementations. The most notable ones include DAViM [49] and Melete [50].

DAViM allows the network to be reprogrammed dynamically by injecting application specific code, and provides isolated execution environments for several running applications.

Melete adopted Maté's TinyScript, a relatively high level programming language, but also introduced a number of modifications to the original Maté VM. These included enhanced code propagation and application deployment mechanisms and most importantly support for concurrent applications.

Another unique VM approach, MagnetOS, is presented in [51]. It is a distributed OS, implemented as a single system image of a unified Java virtual machine that is installed across the sensor network nodes.

A number of other Java based VMs have also been proposed. TinyVM [39] for example -runs atop TinyOS- is a byte code interpreter that implements a subset of the full JVM. Other implementations include Darjeeling [44] and TakaTuka [52].

TakaTuka supports all but two of the Java byte-code instructions, threading, synchronized method calls and is fully compliant with the CLDC library [53], while Darjeeling prioritizes efficiency over functionality and does not provide support for floating point calculations, class reflection or synchronized method calls. Application development using Java is a very attractive option due to its wide portability and simplicity to program in a high level object oriented language.

The authors in [54] describe Scylla, a VM architecture for embedded systems, which

differs in its execution model from all of the previously discussed VM implementations. Scylla VMs use a register based execution model as opposed to the stack based model that is used by most embedded VM implementations. Additionally, unlike most VMs Scylla does not interpret byte code, instead Scylla implements an on-the-fly compiler. Hence compiling byte code to native architectures only requires the re-encoding of the instruction words as they can be directly mapped. Furthermore, the register based execution model does not have the overhead associated with the additional PUSH and POP stack operations and also provide optimization by caching the results of previous calculations allowing faster execution of the instructions. On the other hand, stack based execution models can produce more compact codes because long operand addresses do not need to be explicitly specified instead short stack pointers are used. This property makes stack based VMs more attractive for WSNs due to their lower memory requirement and more efficient dissemination of compact byte code.

3.5 Virtual Sensor (VS)

VSs are software entities that applications can interface with, allowing them to indirectly access the measurements of physical sensors. The concept of virtualising sensors has been studied in a few research works.

Virtual Node Layer (VNL) is a programming abstraction solution that provides a predictable virtual representation of the unpredictable physical resources of sensor nodes. The researchers in [62] propose a generic architecture that simplifies the design and implementation of application development for WSNs. The work utilizes a broad range of virtualisation schemes that help masking the uncertain nature of WSNs. For example, arbitrary regions of the WSN are identified by emulated virtual sensors (VS). This not only makes application programming easier but also increases fault tolerance and provides additional redundancy. On the down side, the clustering approach does not promote concurrency and gives lower accuracy to position sensitive applications (e.g. tracking).

The authors in [63] describe a virtual solution to implement programming interfaces that allow user applications to aggregate sensor data through VSs. This approach allows the heterogeneous physical sensor data to be combined into an abstract measurement. For

example the combination of wind speed, boom angle and load can provide a value to determine whether a construction crane has exceeded its safe working load. This abstraction provides more flexibility and generality to the application developer defining the necessary operations on the various physical data sources via a simple API. The particular focus of this project is on efficient in-network processing as well as data aggregation techniques regarding different data types. The API provided to the developers offer standard data types such as temperature, angle, location, etc. and can be further extended as required. Applications can access sensor data by delegating the corresponding operations to the VS. Using the virtual sensor approach user applications are essentially decouple from the physical data sources so changes in the physical set-up do not effect the applications directly.

Another VS abstraction is presented in [64]. VSs are implemented using the SPINE2 framework [65] that provides programming abstractions based on the task-oriented paradigm. SPINE2 uses graphical constructs to simplify the development of collaborative sensor applications in wireless body sensor networks (WBSN). The project proposes a multi-layer task model that abstracts and combines the components of the WBSN including processing and sampling tasks into a VS.

3.6 Cloud computing aproaches

The idea of utilizing technologies and services that are normally associated with the Cloud have been attracted the attention of investigators in the computing and natural science domains, as it has been indicated by increasing numbers of proposed researches [55–57] . Efforts to overcome the new challenges that implementing Cloud concepts into WSNs imposes have increased over the past years from both academia and industry. The benefits of jointly applying Cloud computing and WSN technologies have been identified in the literature. Existing researches such as [58] and [59] argue that using dynamically scalable Cloud resources is key to provide compelling technical solutions to efficiently store and process -in real time- the exponentially increasing amount of data collected by WSNs. In order to realize the collaboration between the technologies, however, a number of technical challenges needs to be addressed.

Software as a Service (SaaS)

[55] describes a virtualisation approach that combines computing resources in the Cloud with WSN to complement each other. By utilizing virtually unlimited storage and processing power, Cloud computing opens up opportunities for new business models. Following the SaaS paradigm, the work proposes a model that moves sensor data and its processing out of the WSN to into the Cloud hence countervail the limitations of in network computation. The proposed model has a distributed architecture which focuses on offline processing. The parts of the system that are hosted in the Cloud are based on the pipe and filter design patter and offers great flexibility regarding the platforms and programming languages that can be used for the implementation. This provides a number of benefits including better integration of heterogeneous WSN platforms as well as scalability of processing power and data storage. Additionally, data and results can be easily shared amongst multiple users and accessed globally via the Internet.

Network as a Service (NaaS)

Serviceware [41] is a Service Oriented Architecture Based (SOA) middleware that utilizes infrastructure virtualisation techniques to reduce maintenance cost and simplify system operation. Based on the NaaS cloud paradigm, Serviceware exposes the physical WSN substrate to multiple users in the form of virtual WSNs as an on-demand service. To promote infrastructure re-usability, Serviceware specifies an infrastructure slicing approach for cloud based WSNs. Through this approach, Serviceware allows the sharing of WSN resources by implementing a middleware layer -that utilizes multi-threading- on top of a WSN Operating System. The limitation of this approach is that it relies on the underlying OS's multi-threading support.

Sensing and Actuation as a Service (SAaaS)

In [60] the authors introduced a cloud-like paradigm to the Internet of Things (IoT) world. The study presents the SAaaS business model and describes the implementation details of the underlying infrastructure to provide such service. Commoditization of generic sensing

and actuation utilities was enabled by virtualising the participating device resources, offering them as leased services, in accordance with the SLA and QoS requirements. The work also looks beyond WSNs and by including other portable devices such as PDAs and mobile phones equipped with sensors and actuators - offered on a volunteer basis- as part of this infrastructure. The approach taken by the developers was to customize and combine existing solutions wherever available or create new ones where necessary. The proposed software stack implemented the following three components: 1) The hypervisor -operating on the nodes- to virtualise and abstract the sensing and actuating resources; 2) The Volunteer Cloud Manager -operating on the Cloud- that provides the functionalities to manage the offered services; 3) The Autonomic Enforcer -operating on the Cloud- that mediates between the virtual resources and the SAaaS Clouds, and enforces the policies based on SLAs. Ultimately, the work presented an example to successfully applying virtualisation to Cloud inspired paradigms in a WSN environment, focusing on service management and delivery, but did not address the issue of concurrency. Moreover, the services were made accessible via Web interfaces which lacks flexibility in terms of its potential use.

Sensor as a Service (SenaaS)

SenaaS, another cloud inspired model, was introduced in [61]. The research presents a semantic enhanced framework that exposes the capabilities of sensor nodes and other IoT devices as Web services. It is based on event driven, service oriented architecture (SOA) that uses virtualisation as its key technology to provide a service virtualisation layer. Access to the physical layer is provided through interfaces that follow an adapter oriented approach. Moreover, a semantic overlay layer was created to provide a model of the underlying resources using ontology. While the framework successfully demonstrated a simple way to gather telemetry information by exposing sensor resources as virtual services. The work did not target challenges that are associated with in-network node level operations such as programmability, adaptability or constraint/resource management.

3.7 Data Virtualization (DV)

The ROADNet [66] project introduces a Virtual Object Ring Buffer (VORB) framework to virtualise sensor data. The researchers proposed a catalogue that contains meta data about the sensors' system wide properties in a descriptive and structural manner. This catalogue is then used by the users to discover and access the abstract data stream of sensors of interests. At its core, VORB virtualises data streams by using the Storage Resource Broker (SRB) data grids system, which is used to federate storage resources, from multiple ORBs. The framework offers a highly scalable data virtualisation solution due to the fact that the limitations of the underlying physical structures are not limiting factors in deploying VORB.

Making data available as a service, [67] presents the Sensor-Cloud Infrastructure that virtualises multiple physical sensors as a virtual sensor. This allows access to sensor resources by multiple users through automatic provisioning that can scale the allocated resources dynamically. The work identifies two areas that needs to be addressed in order to deliver the proposed service. One is sensor system management and the other is sensor data management. For the management of the system and to represent the relevant characteristics of the physical nodes the Sensor Modelling Language (SensorML) was used. This allowed the mapping of physical resources to virtual ones, that in turn was made accessible via a Web user interface. Virtualisation of sensor data was achieved by implementing publisher/subscriber mechanisms which allowed multiple user applications to subscribe to one or more virtual sensors that published real-time data. On one hand, there are obvious merits to the sensor virtualisation that Sensor-Cloud Infrastructure provides. On the other hand, grouping many physical sensors into a virtual one would result in the loss of position granularity, making it unsuitable for location sensitive applications.

3.8 Mobile agent approach

SensorWare [68] is a framework that was designed to create WSNs that are open to multiple transient users with dynamic needs. By following the active networks paradigm, the framework defines dynamically deployable mobile scripts to re-task WSNs with a particular

focus on programming it as a whole. SensorWare, furthermore, extends the original active networks model by implementing a VM that in turn interprets the high-level scripts that are dispersed through the network as data packets. This allows individual nodes to autonomously re-task other WSN nodes via these data packets. Additionally, code replication and migration is also supported. SensorWare characterises an execution environment that due to the abstraction level it implements allows mobile scripts to be very concise. Moreover, it offers an event-based, high-level programming language model that provides ways to share resources and significantly simplifies implementing distributed algorithms. The disadvantage of defining a model with such high-level abstractions is that, unlike other approaches such as Maté, SensorWare cannot fit into more resource constrained nodes. Maté on the other hand sacrifices ease of programming for an ultra compact instruction set, which ultimately requires more packets (program code) to be transmitted over the network.

Agilla is another mobile agent that was designed to support multiple applications, implemented on top of TinyOs, using tuple space abstraction (a form of distributed shared memory [69]) model. At its core, a VM kernel controls all running agents on the node using a round-robin scheduler for concurrency. The use of the round-robin scheduling algorithm is an inherent limitation of TinyOS's task based execution model. While it ensures starvation free scheduling and it is simple to implement, it does not support prioritization thus it is not suitable to provide dynamic resource management (QoS). Moreover, Agilla uses an assembly like language, that although very compact, does not provide high level abstractions and makes programming more difficult.

3.9 Query Processing Approach

Research efforts have been made to abstract the WSN into a distributed Database (DB) which uses declarative query languages for data acquisition. Examples of this model include COUGAR [70] and TinyDB [71].

COUGAR was one of the first approaches to abstract away much of the complexity of WSNs by viewing it as tables based on the database paradigm . COUGAR threats the entire WSN as a virtual database system and implements management operations using SQL like queries. In-network query processing is optimized in order to reduce resource

usage and extend battery lifetime.

TinyDB was implemented on top of TinyOS, yet it does not require the developer to write code using nesC (the native programming language for TinyOS), instead it provides an interface to extract sensor data using simple DB queries. Furthermore, it supports multi-hop communication and dynamic node reconfiguration at runtime. TinyDB also introduces an advanced energy saving scheme by using semantic routing trees which help reducing the amount of communication involved with the queries.

SwissQM [72] takes data acquisition to another level by implementing a VM that runs optimized byte code instead of queries to process the telemetry data that was extracted from the WSN. Furthermore, SwissQM builds upon the data aggregation model that was first introduced in TinyDB and offers a user extensible instruction set. The work highlights and addresses the following two major limitations of existing systems: *1) lack of data independence, and 2) poor integration with the higher layers of the data processing chain.*

DB approaches can simplify programming and reduce power consumption but in the same time they lack real-time application and multi platform support. Moreover, researchers have identified [73, 74] that in many cases they do not provide sufficient detection accuracy of spatial and temporal relationships between events as required by hard real-time applications.

3.10 Sensor Network Virtualisation (SNV)

SVN has been defined [18, 75] as the separation of the roles of a traditional service provider into two distinct entities: one is sensor infrastructure provider (SInP) that manages the physical substrate; the other one is virtual network service provider (VNSP) who aggregates resources from several SInPs and offers end-to-end services to application level users. Virtual WSNs are composed by a subset of sensor nodes that are typically allocated to particular application or task [76]

The authors in [14] survey the challenges of using virtualisation in federated WSN environment. The challenges primarily derive from the heterogeneity of multi vendor devices as well as from the conflicting economic interests and strict administrative control imposed by the different WSN owners. Moreover, they identify that by utilizing network virtualisation,

heterogeneous WSNs can coexist on a shared physical infrastructures, providing flexibility, promoting diversity, ensuring security and increasing manageability.

One recent work is that attempts to overcome the aforementioned challenges is the VITRO project [19]. VITRO defines a framework of a complete system architecture model for virtual WSNs. The work advocates solutions that can take advantage of virtualisation across the system from node-level to service-level. The report in [77] presents the details of the VITRO framework for efficient service provisioning in which the idea of using traditional virtualisation techniques is applied to WSNs. The framework in the project allows several virtual networks to coexist in a federated fashion, spanning across multiple administrative domains and interconnected via the Internet. The collaboration amongst a subset of sensor nodes supports new dynamically created applications and services that are beyond the scope of the original deployment. The aim of the VITRO architecture was to enable interoperable, adaptive and scalable virtual network platforms.

The SenShare [78] project proposes the idea of open access infrastructures that can support multiple co-running applications. SenShare builds upon the concept of overlay networks and separates the applications from the underlying infrastructure. Furthermore, it provides and isolated execution environment for each application through a node level abstraction layer. By decoupling the ownership and the management of the network substrate, the efficient sharing of a pool of resources is promoted..

The benefits that are provided by SNV are generally identified as improved manageability and security as well as increased flexibility [79].

3.11 Conclusion

This section is still a working progress so please ignore it!!!!!!!!!!!!!!!!!!!!!!!!????????????????????????xxxxxxxxxxxx

This section has reviewed several previous researches that successfully utilized various virtualisation techniques in the context of sensor networks. The review has clearly demonstrated the versatility of virtualisation and has highlighted the potential implications of applying the technology to WSNs regarding both benefits and challenges.

The divergence of potential WSN application domains together with the severe resource

limitation makes system design difficult. In the same time, virtual system run times are required to be adequately efficient and lightweight and support various user applications.

Existing virtualisation approaches often focus on platform [38] or application specific solutions [45]. Over customization, however reduces re-usability and can further aggravate interoperability issues. Future WSNs are expected to be flexible, multi-purpose infrastructures to accommodate numerous concurrent users an support a wide range of versatile applications. Moreover, existing WSN deployments need to be able to dynamically adapt to changes in the environment or in application requirements. Therefore the ability to remotely reconfigure and reprogram the individual sensor nodes or the whole network is imperative as it was demonstrated in [47] and [49] respectively.

4 Expected progress with respect to the state of the art / current literature / current practice of research

5 Research methodology

It is proposed to perform a rigorous study in the context of WSNs in order to meet the objectives outlined in section 2 using the following systematic approach:

- Conduct a comprehensive review of the relevant areas with emphasis on WSN virtualisation techniques (node level, network level) and WSN programming.
- Analyse the review of current literature/practice to gain insight and identify open research challenges.
- Design and develop a new framework for WSN virtualisation by combining existing and novel techniques.
- Design and create appropriate experiments by setting up and configuring a virtualised WSN test bed.
- Evaluate the implemented framework against the desired characteristics specified in section 2.1.
- Validate the scalability and performance of the approach and architecture against traditional approaches and analyse the implications of combining different virtualisation techniques within the various layers of the WSN infrastructure.
- Identify the optimal configuration settings with the aim of achieving a balance between maximum resource utilization/performance and minimum power consumption/management complexity.

6 Work plan

7 Ethical issues

None.

References

[1] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, "Wireless sensor networks for habitat monitoring," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*. ACM, 2002, pp. 88–97.

[2] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, Z. He, and J. Stankovic, "An advanced wireless sensor network for health monitoring," in *Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare (D2H2)*, 2006, pp. 2–4.

[3] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *Information Processing in Sensor Networks, 2007. IPSN 2007. 6th International Symposium on*, April 2007, pp. 254–263.

[4] R. Cardell-Oliver, K. Smettem, M. Kranz, and K. Mayer, "Field testing a wireless sensor network for reactive environmental monitoring [soil moisture measurement]," in *Intelligent Sensors, Sensor Networks and Information Processing Conference, 2004. Proceedings of the 2004*. IEEE, 2004, pp. 7–12.

[5] K.-S. Low, W. Win, and M. J. Er, "Wireless sensor networks for industrial environments," in *Computational Intelligence for Modelling, Control and Automation, 2005 and International Conference on Intelligent Agents, Web Technologies and Internet Commerce, International Conference on*, vol. 2, Nov 2005, pp. 271–276.

[6] M. Durisic, Z. Tafa, G. Dimic, and V. Milutinovic, "A survey of military applications of wireless sensor networks," in *Embedded Computing (MECO), 2012 Mediterranean Conference on*, June 2012, pp. 196–199.

[7] H. M. Ammari, *The Art of Wireless Sensor Networks*. Springer, 2013.

[8] S. Mittal, A. Aggarwal, and S. Maskara, "Contemporary developments in wireless sensor networks," *International Journal of Modern Education and Computer Science (IJMECS)*, vol. 4, no. 3, p. 1, 2012.

[9] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless Sensor Network Survey," *Comput. Netw.*, vol. 52, no. 12, pp. 2292–2330, Aug. 2008. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2008.04.002

[10] M. Obaidat and S. Misra, *Principles of Wireless Sensor Networks*. Cambridge University Press, 2014. [Online]. Available: https://books.google.ie/books?id=V7IkBQAAQBAJ

[11] A. Davis and H. Chang, "A survey of wireless sensor network architectures," *Department of Electrical and Computer Engineering, Tufts University, Medford, USA, International Journal of Computer Science & Engineering Survey (IJCSES) Vol*, vol. 3, 2012.

[12] L. Mottola and G. P. Picco, "Programming wireless sensor networks: Fundamental concepts and state of the art," *ACM Computing Surveys (CSUR)*, vol. 43, no. 3, p. 19, 2011.

[13] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S138918601003024

[14] M. M. Islam and E.-N. Huh, "Virtualization in wireless sensor network: challenges and opportunities," *Journal of Networks*, vol. 7, no. 3, pp. 412–418, 2012.

[15] A. P. Jayasumana, Q. Han, and T. H. Illangasekare, "Virtual Sensor Networks - A Resource Efficient Approach for Concurrent Applications," in *Information Technology, 2007. ITNG '07. Fourth International Conference on*, Apr. 2007, pp. 111–115. [Online]. Available: http://dx.doi.org/10.1109/ITNG.2007.206

[16] I. Leontiadis, C. Efstratiou, C. Mascolo, and J. Crowcroft, "SenShare: Transforming Sensor Networks into Multi-application Sensing Infrastructures," in *Proceedings of the 9th European Conference on Wireless Sensor Networks*, ser. EWSN'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 65–81. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-28169-3_5

[17] C.-L. Fok, G.-C. Roman, and C. Lu, "Agilla: A Mobile Agent Middleware for Self adaptive Wireless Sensor Networks," *ACM Trans. Auton. Adapt. Syst.*, vol. 4, no. 3, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1552297.1552299

[18] M. M. Islam, M. M. Hassan, G.-W. Lee, and E.-N. Huh, "A Survey on Virtualization of Wireless Sensor Networks," *Sensors*, vol. 12, no. 2, pp. 2175–2207, 2012.

[19] M. Navarro, M. Antonucci, L. Sarakis, and T. Zahariadis, "VITRO Architecture: Bringing Virtualization to WSN World," in *Mobile Adhoc and Sensor Systems (MASS), 2011 IEEE 8th International Conference on*, Oct. 2011, pp. 831–836. [Online]. Available: http://dx.doi.org/10.1109/MASS.2011.96

[20] A. Wang, M. Iyer, R. Dutta, G. Rouskas, and I. Baldine, "Network virtualization: Technologies, perspectives, and frontiers," *Lightwave Technology, Journal of*, vol. 31, no. 4, pp. 523–537, Feb 2013.

[21] A. Younge, R. Henschel, J. Brown, G. von Laszewski, J. Qiu, and G. Fox, "Analysis of virtualization technologies for high performance computing environments," in *Cloud Computing (CLOUD), 2011 IEEE International Conference on*, July 2011, pp. 9–16.

[22] A. Murphy, "Virtualization defined–eight different ways," *F5 White Paper*, 2008.

[23] K. Hong, J. Park, S. Kim, T. Kim, H. Kim, B. Burgstaller, and B. Scholz, "TinyVM: An Energy-efficient Execution Infrastructure for Sensor Networks," *Softw. Pract. Exper.*, vol. 42, no. 10, pp. 1193–1209, Oct. 2012. [Online]. Available: http://dx.doi.org/10%2e1002/spe%2e1123

[24] P. J. Denning, "Virtual memory," *ACM Computing Surveys (CSUR)*, vol. 2, no. 3, pp. 153–189, 1970.

[25] R. P. Goldberg, "Architectural principles for virtual computer systems," DTIC Document, Tech. Rep., 1973.

[26] J. Fortes, R. Figueiredo, and P. A. Dinda, "Guest editors' introduction: Resource virtualization renaissance," *Computer*, vol. 38, no. 5, pp. 0028–31, 2005.

[27] P. Levis and D. Culler, "Maté a tiny virtual machine for sensor networks," *SIGARCH Comput. Archit. News*, vol. 30, no. 5, pp. 85–95, Oct. 2002. [Online]. Available: http://doi.acm.org/10.1145/635506.605407

[28] N. Costa, A. Pereira, and C. Serodio, "Virtual machines applied to wsn's: The state-of-the-art and classification," in *Proceedings of the Second International Conference on Systems and Networks Communications*, ser. ICSNC '07. Washington, DC, USA: IEEE Computer Society, 2007, pp. 50–. [Online]. Available: http://dx.doi.org/10.1109/ICSNC.2007.83

[29] S. Hadim and N. Mohamed, "Middleware: middleware challenges and approaches for wireless sensor networks," *Distributed Systems Online, IEEE*, vol. 7, no. 3, pp. 1–1, March 2006.

[30] M. Hailperin, *Operating Systems and Middleware: Supporting Controlled Interaction*. Max Hailperin, 2007.

[31] W. Stallings, *Operating Systems (3rd Ed.): Internals and Design Principles*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1998.

[32] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki - a lightweight and flexible operating system for tiny networked sensors," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, Nov 2004, pp. 455–462.

[33] P. Levis, "Tinyos: An open operating system for wireless sensor networks (invited seminar)," in *Mobile Data Management, 2006. MDM 2006. 7th International Conference on*, May 2006, pp. 63–63.

[34] S. Bhatti, J. Carlson, H. Dai, J. Deng, J. Rose, A. Sheth, B. Shucker, C. Gruenwald, A. Torgerson, and R. Han, "Mantis os: An embedded multithreaded operating system for wireless micro sensor platforms," *Mobile Networks and Applications*, vol. 10, no. 4, pp. 563–579, 2005.

[35] A. Eswaran, A. Rowe, and R. Rajkumar, "Nano-rk: an energy-aware resource-centric rtos for sensor networks," in *Real-Time Systems Symposium, 2005. RTSS 2005. 26th IEEE International*. IEEE, 2005, pp. 10–pp.

[36] M. O. Farooq and T. Kunz, "Operating Systems for Wireless Sensor Networks: A Survey," *Sensors*, vol. 11, no. 6, pp. 5900–5930, 2011. [Online]. Available: http://www.mdpi.com/1424-8220/11/6/5900

[37] A. M. Reddy, A. V. U. P. Kumar, D. Janakiram, and G. A. Kumar, "Wireless Sensor Network OperatingSystems: a Survey," *Int. J. Sen. Netw.*, vol. 5, no. 4, pp. 236–255, Aug. 2009. [Online]. Available: http://dx.doi.org/10.1504/IJSNET.2009.027631

[38] D. Simon and C. Cifuentes, "The squawk virtual machine: Java™ on the bare metal," in *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*. ACM, 2005, pp. 150–151.

[39] K. Hong, J. Park, T. Kim, S. Kim, H. Kim, Y. Ko, J. Park, B. Burgstaller, and B. Scholz, "Tinyvm, an efficient virtual machine infrastructure for sensor networks," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*. ACM, 2009, pp. 399–400.

[40] N. Costa, A. Pereira, and C. Serodio, "Virtual machines applied to wsn's: The state-of-the-art and classification," in *Systems and Networks Communications, 2007. ICSNC 2007. Second International Conference on*, Aug 2007, pp. 50–50.

[41] S. Rea, M. Aslam, and D. Pesch, "Serviceware - a service based management approach for wsn cloud infrastructures," in *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2013 IEEE International Conference on*, March 2013, pp. 133–138.

[42] A. Azzara, S. Bocchino, P. Pagano, G. Pellerano, and M. Petracca, "Middleware solutions in wsn: The iot oriented approach in the icsi project," in *Software, Telecommunications and Computer Networks (SoftCOM), 2013 21st International Conference on*, Sept 2013, pp. 1–6.

[43] R. S. Montero, R. Moreno-Vozmediano, and I. M. Llorente, "An elasticity model for High Throughput Computing clusters," *Journal of Parallel and Distributed Computing*, vol. 71, no. 6, pp. 750–757, 2011, special Issue on Cloud Computing. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0743731510000985

[44] N. Brouwers, K. Langendoen, and P. Corke, "Darjeeling, a Feature-rich VM for the Resource Poor," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 169–182. [Online]. Available: http://doi.acm.org.ezproxy.cit.ie:2048/10.1145/1644038.1644056

[45] P. A. Levis, D. E. Gay, and D. E. Culler, *Bridging the gap: Programming sensor networks with application specific virtual machines*. Computer Science Division, University of California, 2004.

[46] P. Levis, N. Patel, D. Culler, and S. Shenker, "Trickle: A self-regulating algorithm for code propagation and maintenance in wireless sensor networks," in *Proceedings of the 1st Conference on Symposium on Networked Systems Design and Implementation - Volume 1*, ser. NSDI'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 2–2. [Online]. Available: http://dl.acm.org.ezproxy.cit.ie:2048/citation.cfm?id=1251175.1251177

[47] J. Koshy and R. Pandey, "VMSTAR: synthesizing scalable runtime environments for sensor networks," in *Proceedings of the 3rd international conference on Embedded networked sensor systems*. ACM, 2005, pp. 243–254.

[48] D. Palmer, "A virtual machine generator for heterogeneous smart spaces," in *Proceedings of the 3rd Conference on Virtual Machine Research And Technology Symposium - Volume 3*, ser. VM'04. Berkeley, CA, USA: USENIX Association, 2004, pp. 1–1. [Online]. Available: http://dl.acm.org.ezproxy.cit.ie:2048/citation.cfm?id=1267242.1267243

[49] S. Michiels, W. Horré, W. Joosen, and P. Verbaeten, "DAViM: A dynamically adaptable virtual machine for sensor networks," in *Proceedings of the International Workshop on Middleware for Sensor Networks*, ser. MidSens '06. New York, NY, USA: ACM, 2006, pp. 7–12. [Online]. Available: http://doi.acm.org/10.1145/1176866.1176868

[50] Y. Yu, L. J. Rittle, V. Bhandari, and J. B. LeBrun, "Supporting Concurrent Applications in Wireless Sensor Networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, ser. SenSys '06. New York, NY, USA: ACM, 2006, pp. 139–152. [Online]. Available: http://doi.acm.org/10.1145/1182807.1182822

[51] R. Barr, J. C. Bicket, D. S. Dantas, B. Du, T. W. D. Kim, B. Zhou, and E. G. Sirer, "On the need for system-level support for ad hoc and sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. 2, pp. 1–5, Apr. 2002. [Online]. Available: http://doi.acm.org/10.1145/509526.509528

[52] F. Aslam, C. Schindelhauer, G. Ernst, D. Spyra, J. Meyer, and M. Zalloom, "Introducing TakaTuka: A java virtualmachine for motes," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, ser. SenSys '08. New York, NY, USA: ACM, 2008, pp. 399–400. [Online]. Available: http://doi.acm.org.ezproxy.cit.ie:2048/10.1145/1460412.1460472

[53] M. Debbabi, M. Saleh, C. Talhi, and S. Zhioua, "Security evaluation of j2me cldc embedded java platform." *Journal of Object Technology*, vol. 5, no. 2, pp. 125–154, 2006.

[54] P. Stanley-Marbell and L. Iftode, "Scylla: a smart virtual machine for mobile embedded systems," in *Mobile Computing Systems and Applications, 2000 Third IEEE Workshop on.*, 2000, pp. 41–50. [Online]. Available: http://dx.doi.org/10.1109/MCSA.2000.895380

[55] W. Kurschl and W. Beer, "Combining cloud computing and wireless sensor networks," in *Proceedings of the 11th International Conference on Information Integration and Web-based Applications &Amp; Services*, ser. iiWAS '09. New York, NY, USA: ACM, 2009, pp. 512–518. [Online]. Available: http://doi.acm.org.ezproxy.cit.ie:2048/10.1145/1806338.1806435

[56] K. Ahmed and M. Gregory, "Integrating wireless sensor networks with cloud computing," in *Mobile Ad-hoc and Sensor Networks (MSN), 2011 Seventh International Conference on.* IEEE, 2011, pp. 364–366.

[57] C. O. Rolim, F. L. Koch, C. B. Westphall, J. Werner, A. Fracalossi, and G. S. Salvador, "A cloud computing solution for patient's data collection in health care institutions," in *eHealth, Telemedicine, and Social Medicine, 2010. ETELEMED'10. Second International Conference on.* IEEE, 2010, pp. 95–99.

[58] R. Liu and I. J. Wassell, "Opportunities and challenges of wireless sensor networks using cloud services," in *Proceedings of the workshop on Internet of Things and Service Platforms.* ACM, 2011, p. 4.

[59] M. M. Hassan, B. Song, and E.-N. Huh, "A framework of sensor-cloud integration opportunities and challenges," in *Proceedings of the 3rd international conference on Ubiquitous information management and communication.* ACM, 2009, pp. 618–626.

[60] S. Distefano, G. Merlino, and A. Puliafito, "Sensing and Actuation as a Service: A New Development for Clouds," in *Network Computing and Applications (NCA), 2012*

11th IEEE International Symposium on, Aug. 2012, pp. 272–275. [Online]. Available: http://dx.doi.org/10.1109/NCA.2012.38

[61] S. Alam, M. M. R. Chowdhury, and J. Noll, "Virtualizing sensor for the enablement of semantic-aware internet of things ecosystem," *Guest Ed*, vol. 2, pp. 41–51, 2010.

[62] M. Brown, S. Gilbert, N. Lynch, C. Newport, T. Nolte, and M. Spindel, "The virtual node layer: A programming abstraction for wireless sensor networks," *SIGBED Rev.*, vol. 4, no. 3, pp. 7–12, Jul. 2007. [Online]. Available: http://doi.acm.org/10.1145/1317103.1317105

[63] S. Kabadayi, A. Pridgen, and C. Julien, "Virtual sensors: abstracting data from physical sensors," in *World of Wireless, Mobile and Multimedia Networks, 2006. WoWMoM 2006. International Symposium on a*, 2006. [Online]. Available: http://dx.doi.org/10.1109/WOWMOM.2006.115

[64] N. Raveendranathan, S. Galzarano, V. Loseu, R. Gravina, R. Giannantonio, M. Sgroi, R. Jafari, and G. Fortino, "From modeling to implementation of virtual sensors in body sensor networks," *Sensors Journal, IEEE*, vol. 12, no. 3, pp. 583–593, March 2012.

[65] G. Fortino, A. Guerrieri, F. Bellifemine, and R. Giannantonio, "Platform-independent development of collaborative wireless body sensor network applications: SPINE2," in *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, Oct. 2009, pp. 3144–3150. [Online]. Available: http://dx.doi.org/10.1109/ICSMC.2009.5346155

[66] A. Rajasekar, S. Lu, R. Moore, F. Vernon, J. Orcutt, and K. Lindquist, "Accessing sensor data using meta data: A virtual object ring buffer framework," in *Proceedings of the 2Nd International Workshop on Data Management for Sensor Networks*, ser. DMSN '05. New York, NY, USA: ACM, 2005, pp. 35–42. [Online]. Available: http://doi.acm.org.ezproxy.cit.ie:2048/10.1145/1080885.1080892

[67] M. Yuriyama and T. Kushida, "Sensor-cloud infrastructure-physical sensor management with virtualized sensors on cloud computing," in *Network-Based Information Systems (NBiS), 2010 13th International Conference on*. IEEE, 2010, pp. 1–8.

[68] A. Boulis, C. chieh Han, and M. B. Srivastava, "Design and Implementation of a Framework for Efficient and Programmable Sensor Networks." ACM Press, 2003, pp. 187–200.

[69] N. Carriero and D. Gelernter, "Linda in context," *Commun. ACM*, vol. 32, no. 4, pp. 444–458, Apr. 1989. [Online]. Available: http://doi.acm.org.ezproxy.cit.ie:2048/10.1145/63334.63337

[70] Y. Yao and J. Gehrke, "The cougar approach to in-network query processing in sensor networks," *ACM Sigmod Record*, vol. 31, no. 3, pp. 9–18, 2002.

[71] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: an acquisitional query processing system for sensor networks," *ACM Transactions on database systems (TODS)*, vol. 30, no. 1, pp. 122–173, 2005.

[72] R. Mueller, G. Alonso, and D. Kossmann, "Swissqm: Next generation data processing in sensor networks." in *CIDR*, vol. 7, 2007, pp. 1–9.

[73] S. Hadim and N. Mohamed, "Middleware: Middleware challenges and approaches for wireless sensor networks," *IEEE distributed systems online*, no. 3, p. 1, 2006.

[74] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Computer networks*, vol. 52, no. 12, pp. 2292–2330, 2008.

[75] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *Communications Magazine, IEEE*, vol. 47, no. 7, pp. 20–26, Jul. 2009. [Online]. Available: http://dx.doi.org/10.1109/mcom.2009.5183468

[76] H. B. Lim, M. Iqbal, and T. J. Ng, "A virtualization framework for heterogeneous sensor network platforms," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, ser. SenSys '09. New York, NY, USA: ACM, 2009, pp. 319–320. [Online]. Available: http://doi.acm.org/10.1145/1644038.1644080

[77] L. Sarakis, T. Zahariadis, H. C. Leligou, and M. Dohler, "A framework for service provisioning in virtual sensor networks," *EURASIP Journal on Wireless Communications and Networking*, vol. 2012, no. 1, pp. 1–19, 2012. [Online]. Available: http://link.springer.com/article/10.1186/1687-1499-2012-35/fulltext.html

[78] I. Leontiadis, C. Efstratiou, C. Mascolo, and J. Crowcroft, "SenShare: transforming sensor networks into multi-application sensing infrastructures," in *Wireless Sensor Networks*. Springer, 2012, pp. 65–81.

[79] N. M. M. K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *Communications Magazine, IEEE*, vol. 47, no. 7, pp. 20–26, Jul. 2009. [Online]. Available: http://dx.doi.org/10.1109/mcom.2009.5183468