Service Provisioning For The WSN Cloud

Muhammad Sohaib Aslam, Susan Rea and Dirk Pesch
Nimbus Centre For Embedded System Research
Cork Institute of Technology, Ireland
Email: {muhammad.aslam, susan.rea, dirk.pesch}@cit.ie

***Abstract*—The current growth in embedded ICT infrastructure, driven by visions such as the "Smart Cities" concept, is leading to the deployment of a wide range of embedded systems in our environment, which motivates the need for a reusable, flexible and manageable *Wireless Sensor Network or WSN* infrastructure. However, to simplify the system operation and maintenance as well as to reduce costs, WSNs must become an infrastructure that is capable of providing services to multiple end users concurrently, rather than having to roll out individual infrastructures for specific purposes. Here, we present the concept of a WSN infrastructure as a *WSN Cloud*, which provides services to multiple application and data collection systems which adheres to the cloud computing paradigm. Each instance of the WSN Cloud (i.e. a specific set of services configured by a particular end user/system) utilises the WSN infrastructure as if it was a unique network provisioned for their specific requirements. This realisation of the WSN Cloud as *Network as a Service* or *NaaS* requires the WSN to support a service orientated software architecture allowing other systems to provision the WSN infrastructure for their needs with NaaS allowing multiple systems to use the WSN uniquely and concurrently. The WSN-Service Orchestration Architecture "*WSN-SOrA*" presented here is a novel approach to orchestrate service provisioning for embedded networked systems, and enables WSNs to act as cloud ready infrastructures that facilitate on-demand provisioning for potentially multiple individual backend systems.**

***Keywords*-WSN Cloud; Service Orchestration; IaaS;**

I. INTRODUCTION

Smart Cities [1], [2] is a vision for cities that can offer novel services based on the digital integration of city infrastructures through computing systems that enable on-demand service delivery. One of the key enabling technologies which future smart cites will rely on are WSNs [3]. WSNs are a rapidly evolving technology, but in their current form it will not be able to fully support the *Smart Cities* vision due to the costs associated with the equipment, deployment, operation and maintenance of such expansive embedded system infrastructures. One way of reducing cost is for a WSN to be able to make its infrastructure available on-demand simultaneously to multiple users. In the context of this paper, we view these users as enterprise tier systems that aim at virtualising the underlying hardware infrastructure down to the WSN tier.

Cloud computing [4], [5] is based around the concept of delivering services to users. Cloud computing is particularly important for WSNs in terms of broadening their scope. Where Smart Cities are concerned, cloud computing enables a WSN infrastructure to be delivered as a service and in this paper we refer to the virtualisation of WSN infrastructure as a WSN Cloud. Cloud computing consists of the following hierarchical service delivery models.

- Infrastructure as a Service (*IaaS*)
- Platform as a Service (*PaaS*)
- Software as a Service (*SaaS*)

In the context of this paper, we focus on the IaaS concept for delivering the WSN Infrastructure as a service to the end user and providing a mechanism to provision the infrastructure. IaaS encompasses three domains i.e."Compute, Storage and Network", where NaaS corresponds to network domain in IaaS model. In order to evolve a WSN into a cloud infrastructure and to deliver it as a NaaS, all tiers of a WSN must support the functionality associated with the *Service Oriented Architecture (SOA)* [6] principle. We divide the WSN cloud infrastructure into three tiers

- **(tier 1)** Node Network: This tier consists of a network of largely wireless embedded devices which are capable of sensing and actuation. These devices are envisioned to be based on IPv6/6LowPAN technology with wireless network interfaces.
- **(tier 2)** Gateway: This is the middle tier (e.g. contains devices such as embedded PCs) connecting the node network to the backend system. This tier has enhanced computation capabilities and software services in line with backend systems.
- **(tier 3)** Back-End/Enterprise Core: This is the main and computationally most powerful tier, usually running on a server suite. The core provides a platform for implementing components such as management frameworks and end-points for other users/systems that require data or interfacing with the WSN domain.

The SOA principle is well established at the Enterprise and Gateway tier, however it is a relatively new concept for devices in the embedded Node Network tier. Traditionally, embedded wireless sensor/actuator devices have had low computational power not capable of supporting SOA. However, recently these devices have become powerful enough to support SOA [7] [8] [9] along with the required underlying operating systems such as SOS [10] , Lorien [11] and Squawk [12] to support SOA principles at the embedded

978-0-7695-4755-8/12 $26.00 © 2012 IEEE
DOI 10.1109/CLOUD.2012.132

Node Network tier.

In this paper we introduce the concept of the WSN Cloud. This cloud is an organisational domain to which other organisation/enterpise systems connect and provision the infrastructure to deliver services using the NaaS paradigm. In order for the WSN infrastructure to support delivery of NaaS and act as a cloud infrastructure, the ability to support SOA is required at all tiers of the infrastructure. Provisioning a SOA infrastructure where there are a large number of devices at the Node Network tier is a complex problem being faced by high-end networks as well. As manual provisioning greatly increases the likelihood of errors, automated processes are required. Provisioning becomes even more complex in cloud infrastructures where a single physical infrastructure is expected to be provisioned a number of times for concurrent usage. In this paper, we present an orchestration architecture for automated provisioning of the proposed WSN Cloud.

II. Network Virtualisation

The topic of delivery of service by virtualising infrastructure in high-end networks is an active subject of research by industrial stakeholders such as IBM, Cisco, EMC^2 and BMC together with academia. Most of this research is targeted towards high-end networks such as presented in [13] which is part of the EU FP7 4WARD project [14], this project concerns itself with the virtualisation of a network capable of supporting future internet challenges, where [15] describes the scope of virtualising the wireless medium. The focus of these studies and experimentation revolves around the notion of delivering physical infrastructure as an on-demand platform envisioned by projects such as [16]. For a comprehensive survey of network virtualisation see [17]. A WSN is in many ways analogous to a high-end network in terms of being expected to communicate contextual data to other organisations and enterprise systems. In order to serve multiple organisations concurrently, we employ network virtualisation to support NaaS. In this paper, the application of cloud computing principles such as network virtualisation to support true infrastructure reusability for WSNs is novel as far as the authors are aware, with other approaches such as [18] having undertaken data virtualisation or synergical relationship between WSN and cloud [19] [20], rather than infrastructure virtualisation.

III. Network as a Service

NaaS as show in figure 1, provides reusability of the WSN in order to maximise resource utilisation where resources here are the WSN hardware specifically the network devices at the Node Network tier. NaaS is the end result of provisioning the WSN Cloud, as provisioning of the WSN Cloud enables the infrastructure to be delivered as NaaS. In order to facilitate user requirements, the WSN Cloud needs to provision mutliple instances of services, in particular on

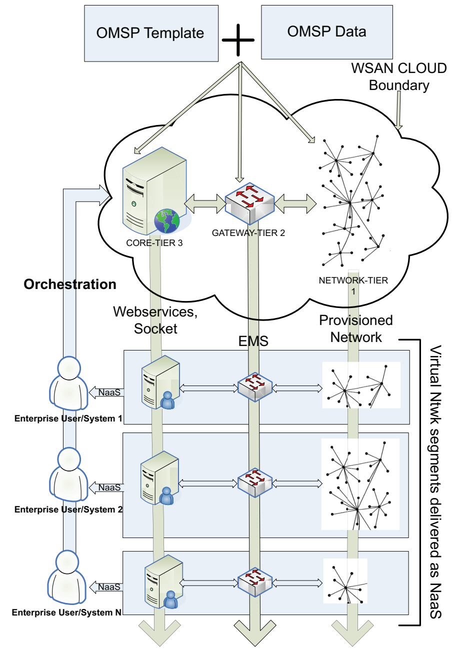
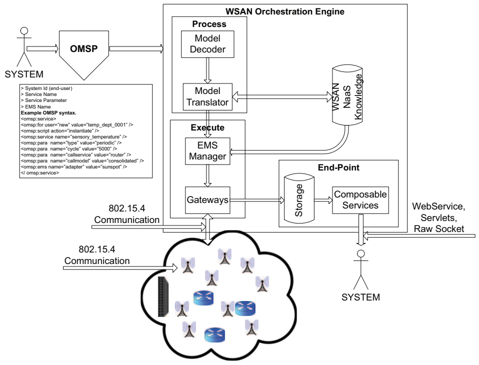


Figure 1. WSN Cloud as NaaS

the embedded network devices such as a sensory service or a routing services in a similar fashion as would happen in a local area network where VLAN, OSPF or trunk-port services are provisioned. In a network with hundreds of devices, however, a requirement for the following is needed:

1) Expedite the process of provisioning services on devices.
2) Reduce the need for the human intervention in the loop to mitigate erroneous output.
3) Reusability of provisioning information for future use.
4) A rapid provisioning engine.

Using an automated approach the provisioning process can be expedited and errors can be reduced. Quantifying the benefits of automated service provisioning can be viewed in terms of time saved, no requirement for expert personnel and reduced labour costs. However, in order to understand how automation expedites the process of provisioning services and reduces the errors associated with the human in the loop, let us consider an example of provisioning a WSN. In order to provision a service in WSN, we need people with expert knowledge of this type of network. A console interface is used to access the network and the devices so that they can be configured based on the end-user requirements. Furthermore in addition to this being a tedious and time consuming task, human intervention may also cause error as the experts have to go through complex configuration parameters manually. Such a manual process demands manpower and time, and this can prove costly for large deployments. Rapid provisioning allows an automated provisioning process to remedy all the forementioned disadvantages. The rapid pro-



(a) Orchestration Engine

OMSP template for : (On-demand): Light Sensors for 4 hours
Pseudo description:
Tier 3: web services (2) for communication on Core J2EE.
Tier 2: Open (1) port 6202 for Raw Socket Com on Gateway SPOT.
Tier 1: Provision Sector 1 Sensors with Service Chain L (Light).
Service Chain L: Interval 2 minutes+ Read from Light Sensor + pass reading to threshold service (if light read is >30) + pass to radio service (use LQRP routing).

OMSP template

OMSP data:
Pseudo description:
Sector 1: 04F:0000.4101.1101 – 1201.
Sector 2: 04F:0000.4101.1201 – 1301.
Sector 3: 04F:0000.4101.1301 – 1401.
Sector All: 04F:0000.4101.1101 – 1401.
Core J2EE: J2EE@192.168.0.2
Gateway SPOT: EMS-SunSPOT@192.168.0.1

OMSP data

(b) Pseudo Description: OMSP Model

Figure 2. Delivery of WSN Cloud as NaaS

visioning concept is extensively applied in the Compute and Storage Cloud domain, i.e. VMware is an example of a tool for rapid provisioning in large scale data centres . However, the network cloud domain is still problematic where custom provisioning is an issue and this is true for both wired and wireless networks. The custom provisioning process is a long and error prone process mainly due to manual service provisioning and the requirement for network experts. WSNs have been traditionally seen as an isolated, specialised domain, where the application of cloud computing and SOA are relatively novel concepts. In order for a WSN to be cloud ready the following properties are necessary:

- Need to expose the WSN as a NaaS, to deliver cloud infrastructure.
- WSN can be shared system used by multiple entities with different requirements.
- Automated and rapid provisioning of WSNs for large scale networks.

IV. Service Orchestration Architecture

The WSN Service Orchestration Architecture (WSN-SOrA) as shown in Figure 2a is the architecture we propose and have developed to make WSN cloud ready, to deliver NaaS and to automate service provisioning and de-provisioning in the WSN Cloud. The two key elements of this architecture described in this paper are the orchestration engine which is used for provisioning services at all tiers of the WSN Cloud and orchestration model that acts as a driver for the orchestration engine.

WSN-SOrA provides a comprehensive automation system for provisioning and de-provisioning, by introducing the concept of model based orchestration. Orchestration is a process which enables the WSN Cloud to be provisioned or de-provisioned rapidly, driven by our *Orchestration Model for Service Provisioning (OMSP)*. Orchestration allows the rapid provisioning of services in network devices, the gateway and core that fulfill cloud user requirements. In a traditional single tier WSN network, devices are configured either using a terminal to the device i.e. ”command line” or using some sort of web interface and in most cases are hard-coded. The reason for such practice lies in the fact that WSNs were never expected to offer cloud services, that is providing services for multiple end users concurrently. A static interface has been sufficient so far. Similarly, in high-end networks there is a similar situation as most of the network devices are provisioned using terminals and static interfaces.

A. Model Driven Orchestration

Models have been used for a number of purposes from simulating a system to using models as a means of understanding how a system functions. The OMSPs are simple system descriptive models that are used as drivers for orchestrating the rapid provisioning of a WSN Cloud.

OMSP is a generic model for service provisioning, which can be written using XML. OMSP is not a template, as templates are static entities provided by the system with fixed properties , where OMSP is a dynamic model which is built by users bearing custom properties and logic. The OMSP can be divided into two segments - the OMSP process logic and the OMSP data. An OMSP process logic contains a pointer to the network devices and a service description of the services required to be provisioned. The OMSP data contains the device data such as the device address corresponding to the pointer in the OMSP process logic. An OMSP process logic can be saved as templates to provide a reusability mechanism for provisioning a number of different networks. The OMSP data is related to the network itself and contains network dependent data. For example figure 2b shows a pseudo description of an OMSP for reading light from arbitrary ”sector one” devices for a limited time. The OMSP process logic contains logic that can be applied to a number of different WSN cloud infrastructures, where the OMSP data contains the data pertaining to the devices in all tiers of a specific WSN cloud.

B. Orchestration Engine

The Orchestration Engine is a model (OMSP) driven system which uses the OMSP for provisioning the required services on all three tiers. The Orchestration engine consists of following components:

- The Core
- Element Management System (EMS)
- NaaS Endpoint

1) Core (Back-End): The core of the orchestration engine is the CPU of the engine. It contains the translating components to read the OMSP. The core also configures the service end points which act as a delivery point of NaaS to the user.

2) Element Management System (Gateway): Once the OMSP is translated, the consolidated data is sent to the EMS. Consolidated data refers to the combination of the data from the OMSP and the knowledge base (this is a collection of meaningful data about the network and is described further in section IV-D) in the orchestration engine to enable the EMS to execute the operations necessary to complete the service provisioning tasks. Each instance of EMS manages a specific set of devices i.e. an EMS for SunSPOTs, an EMS for TelosB, etc. The EMS is an overlaying system that manages the gateways where gateways are systems running on physical computing devices, e.g. embedded wireless devices, specialised embedded PC boards, etc., that are capable of communicating with Node Network tier devices. The EMS calls the gateway device to disseminate the service provisioning data to the network devices in the WSN Cloud supported by communication protocols such as IEEE 802.15.4. The data for service provisioning is simply a few bytes as it does not contain system script or code, rather just the service id and the parameters of the services to be instantiated. While a full service upgrade such as updating functionality is possible, the *WSN-SOrA* does not recommend it due to the fact that devices in the WSN are assumed to run on limited energy resources. A Service upgrade can be compared to a full or part firmware upgrade in a high-end network that modifies the implementation of the service. Similarly, a WSN service upgrade can be used to modify the implementation of the service, however such an operation is risk prone as this means the devices are unavailable while the update is on-going and in battery powered wireless networks this consumes a considerable amount of battery power. Furthermore, even in high-level wired systems it is not common practice as it causes disruption to the network service while a service upgrade is in progress.

3) NaaS Endpoint: This component is an output product of the orchestration engine. Once the orchestration engine executes the OMSP for a user, a service endpoint is provided usually in the form of a web-service or raw socket communication. The data from the network provisioned for a specific user is provided to the through this service point. Figure 2a shows flow of Orchestration Engine.

The orchestration engine is a comprehensive automated mechanism for the rapid provisioning of services to provide a cloud user with the requested service by using WSN as IaaS. This engine is driven by the OMSP which is a generic model that allows the user to translate its service request. These models can be reused for similar WSNs with similar service requirements.

C. Serviceware

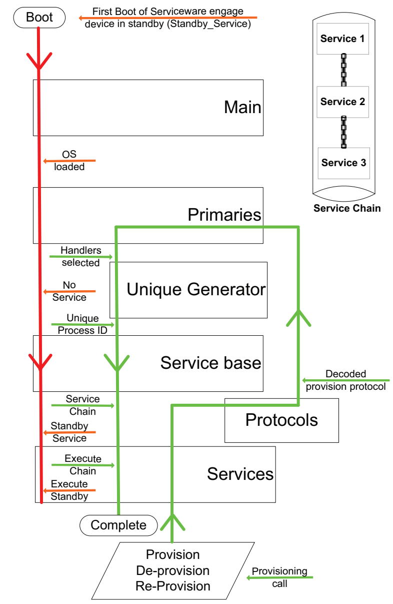


Figure 3. Serviceware for SunSPOTs

The Serviceware (as shown in Figure 3), is what we refer to as our implementation of SOA principles in order to support the WSN-SOrA on Node Network tier devices i.e. embedded network devices. The concept of serviceware is somewhat inspired by the virtualisation of infrastructure in high-end wired (VRouter, VLAN) and wireless networks (Virtual Radio-VR) [21]. The serviceware is an experimental piece of software implemented on SunSPOTs and TelosB running Squawk and Lorien, which virtualises the underlying functionality of Node Network tier device. Provisioning in embedded devices is supported with the concept of a service chain. A service chain is a collection of services which are in a sequence and the output of each service is passed to the next service in the chain, the last service in the chain is usually a radio communication service to send data to the required destination. In the serviceware each service chain execution is controlled by a separate thread and the frequency of execution or duty cycle of the chain depends upon the type of data collection requirement i.e "periodic or event based". In a simple provisioning call after the initial device boot the call is processed in the following manner as shown in figure 3.

- Firstly the device receives the provisioning data from a standby service initiated at the boot time from the "Services package".
- The provisioning call packet is decoded by the classes in the package "Protocols". After parsing, the call is

passed to the classes in the ”Primaries” package where a thread is created and a unique number is associated with that thread in the ”Unique Generator” package. This unique number is the unique identifier of the provisioned service on the device.
- Afterwards a service chain is created by the classes in the package ”Service base”, based on required services and finally the service chain is started.
- The service chain ends when de-provisioning is called or if the time or execution limit has elapsed in the case of an on-demand execution where a time limit is given. The serviceware has been created as a means of supporting SOA on operating system such as Squawk and Lorien in an effort to provide support for virtualistation and prototyping of the WSN Cloud concept presented in this paper.

D. Orchestration Flow

The design of an OMSP in the process of provisioning or de-provisioning is the first stage. As the OMSP is passed to the orchestration engine by interface such as webservice, the rest of the process becomes automated. The OMSP is first decoded and translated. The process of decoding and translation uses references from knowledge base which is a collection of databases. OMSP is typically an XML document and the translation is interpretation of tags and values. If a new interpretation is required this knowledge base acts as a repository for developers to include new references. The knowledge base also stores the network data coming from devices for upgrading information about state of the WSN cloud i.e. Node Network Tier device health data such as energy levels, signal strengths etc. The knowledge base also contains data pertaining to the available services for each device in the network and the current instance of those services. In summary, the knowledge base is a collection of meaningful data about the network and is used to support orchestration and other operations. Once each service is identified and devices are selected, a secondary document is created by the orchestration engine, which contains instructions for the execution part of the orchestration engine. The EMS manager reads this document and instantiate an instance of an appropriate EMS gateway depending on the hardware platform. The number of instances provides scalability and depends upon the hardware capability on which the system runs. These gateways contain the base station device capable of communicating with network devices of the same type in the WSN Cloud. The instruction is then disseminated in the WSN Cloud and the services in the network are provisioned. Dissemination in the WSN Cloud is based on the configured wireless communication interfaces and protocols such as IEEE 802.15.4/6LowPAN. The orchestration engine needs to create an end-point for the newly provisioned network so that the requesting system may extract its data. This can be done in the form of setting webservices (passive polling), raw sockets (stream), or servlets (push). De-provisioning is similar to provisioning only services are de-instantiated instead of being created.

V. Enabling a WSN Cloud

In order to implement a WSN Cloud, based on the architecture and processes outlined above, an operating system which enables a SOA implementation is a basic requirement for Node Network tier devices. Operating systems such as Lorien, SOS, Squawk, .net Micro Framework provide this ability unlike TinyOS. Table I shows the statistical comparison of implementation on different platforms. For our experimentation we have worked with the TinyOs, Lorien and Squawk operating systems and have used the TelosB and SunSPOTs WSN node platforms. We developed the serviceware for this experimentation to enable virtualisation and provide the device infrastructure as a service. The size of service on each platform varies in the number of services it can support, which is directly proportional to the memory available in these devices. The following describes the stages involved in enabling a device for the WSN Cloud:

- Commissioning: this is the factory level stage where a device is loaded with the operating system and the serviceware being programmed to run on these devices. At the end of this stage the device is ready to be inducted into the network.
- Provisioning: Once the network is set (”The process of designing the physical and logical network is outside of the scope of this work and the WSN SOrA considers the physical infrastructure to be in place when instigating provisioning), a system/end-user may request to provision the WSN Cloud infrastructure. The device listens to the command from the gateway and instantiates (or de-instantiates) services i.e (Starting, Stoping or modifying service chain execution). This information comes from OMSP directly.
- Orchestration Engine and End-point: The Orchestration engine and end-point is implemented as a back-end service. The orchestration engine not only performs the provisioning and de-provisioning but also enables the end-point middleware creation which allow the system/end-user to interact with the WSN Cloud. We have implemented this using the Java enterprise Server GlassFish [22]. The orchestration engine is implemented using enterprise beans, webservices and java scripts. Architecturally the orchestration engine can be implemented easily with any server end technology.

VI. Qualitative analysis

In order to qualitatively evaluate the validity of the proposed architecture (the WSN Cloud and WSN-SOrA) concepts, we have taken a similar approach as that taken by Roy Fielding in his doctoral dissertation [23] where the core contribution of his work was the RESTful software

Table I
PROTOTYPE COMPARISON

Platform	TelosB	TelosB	SunSPOT
Flash Memory	48K	48K	4MB
OS	TinyOS	Lorien	Squawk
OS Type	Monolithic	Component, runtime loading, supports SOA	Component, runtime loading, supports SOA, VM based
Serviceware Code Size	10KB (whole image)	38KB	1.6MB (VM based)
Initial Service(s)	Temp	Temp., humidity	Temp., light, humidity, routing (AODV, LQRP), Accelerometer
Commissioning Time (serial)	<1s	<1s	<2s
Provisioning Code Size	N/A Monolithic image, redeploy whole image	127 Bytes* *depends on radio protocol, IEEE 802.15.4 max frame size is 127bytes	127 Bytes
Provisioning Time (OTA)	N/A	<4ms	<4ms

architecture. Table II shows a comparison of the functional qualities of existing categories of solutions available for WSNs against the proposed WSN-SOrA. In addition it is our premise that all of the functional qualities identified in this table are necessary to support the WSN Cloud concept. Table III refers to the scope or tier targeted (i.e. the node, gateway, or backend tier) by existing categories of tools, applications and middleware available for WSNs against the proposed WSN-SOrA. The categories (Traditional, DB, VM, SOA and Management) used in both tables are defined based on the survey and studies of WSNs in [24] and [25]. Traditional refers to hard-coded, single-tier solutions, DB is in reference to database type solutions and relies on a querying approach, VM is virtual machine based and typically platform specific, SOA points to solutions conforming to a service oriented architecture approach and finally, Management is a reference to solutions that focus on network management and that include adaptivity in terms of device reconfiguration. To compare the functional capability in table II, Application is the ability of a solution to provide or act as an application, which is supported by all solutions including the WSN-SOrA. Provisioning refers to the ease and ability to configure/reconfigure different tiers of the WSN, where existing solutions either provide none as in the DB case or are severely limited as in the other cases when compared to the WSN-SOrA approach. Reusability defines the ability to support infrastructure reuse as well as the possibility to reuse the service provisioning design for other networks. SOA and WSN-SOrA are the only solutions able to do this. Automation in our case is the ability to automatically provision all the tiers by using a simple driver such as a model with WSN-SOrA being the only approach capable of doing this. Simplicity encapsulate the idea of models, where complex coding and configuration is simplified by removing the need for expert knowledge every time a network is to be setup or modified. Scalability encompasses the ability to rapidly provision large networks with relative ease where WSN-SOrA being the only solution to support the concept of large scale rapid provisioning in WSN Cloud. From table II it is evident that the proposed WSN-SOrA is the only solution to address all the required functionalities listed making it the only solution capable of acting as an enabler for the WSN-Cloud.

As existing approaches are limited in scope (i.e. most solutions typically target the node network tier only), one of the key features of the WSN-SOrA is its applicability across all tiers of the WSN and that it readily conforms to cloud based architectures as shown in table III. The available tools and technologies in the WSN domain are mostly concentrated at device level (Node Network tier) and as such the divide between this technology and high end networks is reinforced. Relatively few solution in the WSN domain moved beyond Node Network tier and those that do are limited. For example GSN [26], in the category DB only runs at the gateway level and acts as database where all sensory data is logged, likewise in the SOA category TinySOA [8] runs at Node Network tier only and sensorweb2.0 [27] runs at Enterprise Core tier . The proposed WSN-SOrA is an attempt to broaden the scope of WSN technology and to garner the functionalities of the WSN and promote it as an organisation domain (a self contained infrastructure) which forms part of an enterprise network that can easily facilitate other organisations through NaaS. Most of the available solutions in WSNs are outdated (in terms of their ability to support SOA and cloud computing principles) and provide a single tier scope i.e. they are typically limited to the embedded network devices and as such are unable to support the concepts of orchestration and service provisioning in WSN Cloud which WSN-SOrA readily provides.

A. Use Case

Further aspects to be considered in the qualitative analysis are:

1) Virtual Segmentation & Infrastructure Reusability.
2) Expedited Reusable Provisioning

and we describe these in the context of a use case, where we developed a prototype implementation of the WSN-

Table II
FUNCTIONAL COMPARISON

	Application	Provisioning	Reusability	Automation	Simplicity	Scalability
Traditional	X					
DB	X					
VM	X	X				
SOA	X	X	X			
Management	X	X			X	
WSN-SOrA	X	X	X	X	X	X

Table III
TIER SCOPE COMPARISON

	Tier 1	Tier 2	Tier 3	Cloud
Traditional	X			
DB	X	X		
VM	X			
Adaptive	X			
SOA	X	X	X	
Management	X	X	X	
WSN-SOrA	X	X	X	X

SOrA using the TelosB and SunSPOT platforms as the node network tier.

1) Virtual Segmentation & Infrastructure Reusability: By virtualising the WSN infrastructure the provisioning process configures the network into virtual segments for cloud user as isolated pipelines. This allows different users to make use of the WSN simultaneously for their needs without affecting others users. Devices in the WSN Cloud are mostly multipurpose and can have several components such as light sensors, humidity sensors, accelerometers etc. The presented approach allows a single infrastructure to be reused simultaneously. This approach reduces the infrastructure cost significantly, in our case by 50 percent, where three end users required two different services from the same infrastructure of 30 sensors (in this case light and temperature). If the users have different requirements the typical approach is to configure and schedule individual use of the network for each user independently. As technology advances, WSN network devices will have increasing functionality and in this scenario the WSN Cloud approach will provide an efficient mechanism for infrastructure reusability in the form of NaaS.

2) Expedited Reusable Provisioning: We have experimented with 30 sensor devices for three different cloud user service configurations. For each device there are specific service parameters which are required. The OMSP represents those required parameters in the form of a model. The configuration includes:

- Specifying the service interval or conditional constraints for event based reporting.
- Selection of the appropriate parameters for the requested service such as a light sensory service.
- Linking of services when services are dependent on each other.

In our case, the user created three OMSP to represent their provisioning requirements. The models were passed to the system and the provisioning dispatch for each device was measured as being between 4ms-10ms (data transfer, plus retransmissions with variable packet sizes), where provisioning in the device is instant as the devices are not required to do a restart in order to instantiate a new service. The need for a restart is heavily dependent on underlying OS. In this experiment we analysed the manual configuration for 30 devices for three different cloud users. Configuration is typically done using Web interfaces by filling out parameter fields, using hyper terminals or serial command line interfaces. These are one of configurations, as there is no element of reusability in that if a similar network is to be provisioned then the same sequence of steps must be repeated, there is no saving on time or man power, whereas with the proposed system the model process logic can be reused to provision systems with similar requirements. Moreover, for devices which run systems conforming to a monolithic architecture eg TinyOS, it may take extra time to provision a network as the whole operating system has to be redeployed. The factors that can influence the time and cost involved in provisioning an infrastructure are programming skills, manual provisioning and reusability.

VII. CONCLUSION

This paper proposes a new concept for WSNs i.e. the WSN Cloud which extends beyond the norm of considering a WSN as a monolithic infrastructure designed for a specific application. Furthermore, we have applied the NaaS concept to the WSN cloud and provided a solution for orchestrating service provisioning in large scale WSNs derived from concepts used in high-end enterprise networks. The WSN-SOrA has been presented as an enabler of the WSN Cloud concept and virtualises the WSN infrastructure in the form of NaaS.

ACKNOWLEDGMENT

The authors wish to acknowledge the support of Science Foundation Ireland under the 06-SRC-I1091 ITOBO project in part funding the work reported in this paper.

REFERENCES

[1] H. Waer and M. Deakin, *From Intelligent to Smart Cities*. Taylor & Francis, 2012.

[2] M. Kehoe, M. Cosgrove, S. D. Gennaro, C. Harrison, W. Harthoorn, J. Hogan, J. Meegan, P. Nesbitt, and C. Peters, *Smarter Cities Series: A Foundation for Understanding IBM Smarter Cities*. An IBM Redguide publication, 2011. [Online]. Available: http://www.redbooks.ibm.com/abstracts/redp4733.html?Open

[3] T. Watteyne and K. S. J. Pister, "Smarter cities through standards-based wireless sensor networks," *IBM Journal of Research and Development*, vol. 55, no. 1.2, pp. 7:1 –7:10, jan.-march 2011.

[4] G. Reese, *Cloud Application Architectures: Building Applications and Infrastructure in the Cloud*. OReilly, 2009.

[5] D. S. Linthicum, *Cloud Computing and SOA Convergence in Your Enterprise: A Step-by-Step Guide*. Addison-Wesley Professional, 2009.

[6] M. Bell, *Service-Oriented Modeling (SOA): Service Analysis, Design, and Architecture*. Wiley, 2008.

[7] A. Rezgui and M. Eltoweissy, "Service-oriented sensor-actuator networks [ad hoc and sensor networks]," *Communications Magazine, IEEE*, vol. 45, no. 12, pp. 92 –100, december 2007.

[8] E. Avils-Lpez and J. A. Garca-Macas, "Tinysoa: a service-oriented architecture for wireless sensor networks," *Service Oriented Computing and Applications*, vol. 3, no. 2, pp. 99–108, 2009. [Online]. Available: http://www.springerlink.com/index/10.1007/s11761-009-0043-x

[9] E. Meshkova, J. Riihijarvi, F. Oldewurtel, C. Jardak, and P. Mahonen, "Service-oriented design methodology for wireless sensor networks: A view through case studies," in *Sensor Networks, Ubiquitous and Trustworthy Computing, 2008. SUTC '08. IEEE International Conference on*, june 2008, pp. 146 –153.

[10] C. chieh Han, R. Kumar, R. Shea, E. Kohler, and M. Srivastava, "Sos: A dynamic operating system for sensor networks," in *Proceedings of the Third International Conference on Mobile Systems, Applications, And Services (Mobisys*. ACM Press, 2005.

[11] B. Porter and G. Coulson, "Lorien: a pure dynamic component-based operating system for wireless sensor networks," in *Proceedings of the 4th International Workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks*, ser. MidSens '09. New York, NY, USA: ACM, 2009, pp. 7–12. [Online]. Available: http://doi.acm.org/10.1145/1658192.1658194

[12] "Squawk for sunspot," Online, Jan. 2012. [Online]. Available: http://labs.oracle.com/projects/squawk/squawk-sunspot.html

[13] L. Volker, D. Martin, I. El Khayaut, C. Werle, and M. Zitterbart, "A node architecture for 1000 future networks," in *Communications Workshops, 2009. ICC Workshops 2009. IEEE International Conference on*, june 2009, pp. 1 –5.

[14] "The FP7 4WARD Project." [Online]. Available: http://www.4ward-project.eu/

[15] S. Perez, J. Cabero, and E. Miguel, "Virtualization of the wireless medium: A simulation-based study," in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, april 2009, pp. 1 –5.

[16] N. Feamster, L. Gao, and J. Rexford, "How to lease the internet in your spare time," *SIGCOMM Comput. Commun. Rev.*, vol. 37, pp. 61–64, January 2007. [Online]. Available: http://doi.acm.org/10.1145/1198255.1198265

[17] N. M. K. Chowdhury and R. Boutaba, "A survey of network virtualization," *Comput. Netw.*, vol. 54, pp. 862–876, April 2010. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2009.10.017

[18] 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*, sept. 2010, pp. 1 –8.

[19] 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 & Services*, ser. iiWAS '09. New York, NY, USA: ACM, 2009, pp. 512–518. [Online]. Available: http://doi.acm.org/10.1145/1806338.1806435

[20] K. Lee and D. Hughes, "System architecture directions for tangible cloud computing," in *Cryptography and Network Security, Data Mining and Knowledge Discovery, E-Commerce Its Applications and Embedded Systems (CDEE), 2010 First ACIS International Symposium on*, oct. 2010, pp. 258 –262.

[21] J. Sachs and S. Baucke, "Virtual radio: a framework for configurable radio networks," in *Proceedings of the 4th Annual International Conference on Wireless Internet*, ser. WICON '08. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008, pp. 61:1–61:7. [Online]. Available: http://dl.acm.org/citation.cfm?id=1554126.1554202

[22] "Glassfish server," Online, Jan. 2012. [Online]. Available: http://glassfish.java.net/

[23] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[24] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," *Comput. Netw.*, vol. 52, pp. 2292–2330, August 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1389582.1389832

[25] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, pp. 393–422, 2002.

[26] K. Aberer, M. Hauswirth, and A. Salehi, "Invited talk: Zero-programming sensor network deployment," in *Applications and the Internet Workshops, 2007. SAINT Workshops 2007. International Symposium on*, jan. 2007, p. 1.

[27] X. Chu, Kobialka, B. Durnota, and R. Buyya, "Open sensor web architecture: Core services," in *Intelligent Sensing and Information Processing, 2006. ICISIP 2006. Fourth International Conference on*, 15 2006-dec. 18 2006, pp. 98 –103.