Provisioning within a *WSAN Cloud* Concept

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

ABSTRACT

Traditionally, Wireless Sensor and Actuator Networks or (*WSANs*) have been used as a standalone technology for a specific application purpose such as heating control. The current growth in embedded ICT infrastructure is leading to the deployment of a wide range of embedded systems in our environment, which motivates *System of Systems* [5] architectures and ultimately, with deployment of IP technologies into this space, the *Internet of Things* [10] paradigm. However, to simplify system operation and maintenance as well as to reduce costs, WSANs must become an infrastructure that is capable of providing services to multiple end users concurrently rather than requiring a new infrastructure for a new purpose. Here, we present the concept of a WSAN infrastructure as a *WSAN Cloud*, which provides services to multiple application and data collection systems following the cloud computing paradigm. Each instance of the WSAN cloud (i.e. a specific set of services configured by a particular end user/system) utilises the WSAN infrastructure as if it was a unique network provisioned for specific requirements. A realisation of the WSAN Cloud in the form of *Network as a Service* or *NaaS* requires a WSAN to support a service orientated software architecture allowing other systems to provision the WSAN infrastructure for their specific needs and allowing multiple systems to use the WSAN uniquely and concurrently. The WSAN-Service Orchestration Architecture "*WSAN-SOrA*" presented here, is a novel approach to service provisioning of embedded networked systems and enables WSANs to act as cloud ready infrastructures that facilitate on-demand provisioning for potentially multiple individual backend systems.

1. INTRODUCTION

While research in the areas of routing, channel access, duty cycle adaption, power control and such like are relatively mature in the WSAN space, approaches to virtualisation, shared sensor network infrastructure access and the subsequent operations, administration and maintenance (Operations, administration and management (OAM)) systems necessary to manage such networks are still in their infancy. Traditionally WSANs are designed around specific applications and as such are seen as fit for purpose deployments. This tight coupling between the application and the WSAN is suited to short terms and small scale deployments but beyond that the utility of the WSAN is minimised. For next generation WSANs the likes of which are envisioned by concepts like internet of things which in turn is driving future WSN application spaces such as *Smart Cities* [14], [7] will see large scale and potentially massively distributed sensor networks[9]. The fit for purpose WSAN design and deployment common nowadays will not be sustainable in next generation WSANs networks, where multiple applications and end users are expected to act on a single infrastructure harmoniously. Smart cities is a view to the future for cities where novel services will be offered based on the digital integration of city infrastructures through computing systems that enable on-demand service delivery. Smart cities will revolutionise the way in which organisations will communicate in future urban environments, which will be based on the paradigm of integration and on-demand service delivery. While WSANs are seen as an enabling technologies for next generation networks a shift away from the single application serving a single end user is needed. WSANs are a rapidly evolving technology but in their current form will not be able to fully support the *Smart Cities* vision due to the cost associated with equipment, deployment, and operations and maintenance of such an extended embedded systems infrastructure. One way of reducing cost is for WSANs 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 based on cloud computing concepts that aim at virtualising the underlying hardware infrastructure down to the WSAN tier. Cloud computing [13], [8] is a mechanism that offers end users a platform where dynamically scalable resources are accessible virtually as services over the Internet. With WSANs being seen a promising enabler of smart cities applications and the like, the widely held belief is that the synergies between WSANs and cloud computing will offer a potential solution to managing next generation networks [9] in the form of both data and infrastructure virtualisation. Cloud computing enables WSAN infrastructure and resources to be delivered as a service and in this paper we refer to the virtualization of WSAN infrastructure as a WSAN Cloud. Cloud computing follows SPI (Software, Platform and Infrastructure) model

of service delivery. In the context of this paper, we focus on the IaaS (Infrastructure as a Service) concept for delivering the WSAN 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", and here we focus on the Network as a Service or NaaS aspect. In order to evolve a WSAN into a cloud infrastructure for NaaS, all tiers of a WSAN must support the functions associated with the *Service oriented Architecture (SoA)* [3] principle. We divide the WSAN 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, such as IEEE802.15.4, to communicate with the Gateway tier.
- **(tier 2)** Gateway: This is the middle tier connecting the node network to the backend system. This tier has enhanced computation capabilities and software services in line with backend systems, with interfaces to the Back-end/Enterprise Core tier being RPC, web services or sockets.
- **(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 WSAN 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 concepts. However, recently these devices have become powerful enough to support SoA [2] [11] along with the required underlying operating systems such as SOS [4] , Lorien [12] and Squawk [1] to provide necessary platform to implement SoA principles at the embedded Node Network tier.

In this paper we introduce the concept of the WSAN 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 WSAN 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 automatic provisioning of the proposed WSAN Cloud.

2. PROVISIONING FOR NAAS

2.1 Provisioning

Provisioning is an ambiguous term which is widely used in networking. For example [6], [15] use the term provisioning to define the configuration of a sensor network application. Provisioning in the context of this paper refers to service provisioning in a network to support the concept of NaaS. Although the focus of our research presented here considers the WSAN Cloud within building management, the software architecture proposed is not restricted to buildings but can be used for a wide range of smart infrastructure. As delivery of NaaS requires provisioning, we must support provisioning at the following tiers.

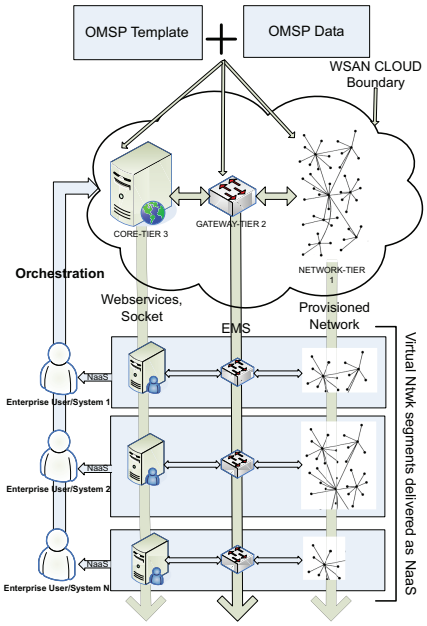
- Node Network Tier: Provisioning device services.
- Gateway Tier: Provisioning service to open end-points and data processing logic.
- Enterprise Core Tier: Provisioning of high-end services to manage segments of the cloud infrastructure for each end user.

2.2 Delivering NaaS

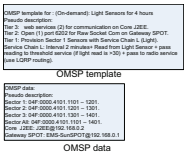
NaaS is required to provide reusability of the WSAN in order to maximise resource utilisation where resources here are the WSAN hardware i.e. (network devices at the Node Network tier). The user of a NaaS can request a service based on certain requirements e.g. a new Subnet or Humidity readings in a specific WSAN zone. In order to facilitate user requirements the WSAN Cloud needs to provision services, in particular on the embedded devices such as a sensing service or a routing services in a similar fashion as 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 in the loop to mitigate erroneous output.
3. Reusability of provisioning information for other networks
4. A rapid provisioning engine.

Using automation 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 WSAN. In order to provision a service in a WSAN we need people with expert knowledge of this type of network. A console 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



(a) Provisioning Overview



(b) OMSP Model

Figure 1: WSAN SOrA

can prove costly for large deployments. Rapid provisioning allows an automated provisioning process to remedy all the forementioned disadvantages. The rapid provisioning concept is extensively applied in the Compute and Storage Cloud domain, i.e. VMware is an example of a tool for rapid provisioning. However, the network cloud component is still problematic where custom provisioning is still an issue and is true for both wired and wireless networks. The custom provisioning process is long and error prone mainly due to manual service provisioning and the requirement for network experts. WSANs 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 WSAN to be cloud ready the following properties are necessary:

- Need to expose the WSAN as a NaaS providing cloud infrastructure.
- The WSAN can be used by multiple systems with different requirements.
- Automated and rapid provisioning.

3. SERVICE ORCHESTRATION ARCHITECTURE

The WSAN Service Orchestration Architecture (WSAN-SOrA) as shown in Figure 1a is the architecture we propose and have developed for cloud ready WSANs to deliver NaaS and to automate service provisioning and de-provisioning in the WSAN Cloud. The key attribute of this architecture is to provide an orchestration engine for provisioning services at all tiers of the WSAN Cloud. The ability to support service provisioning in the WSAN by multiple end user systems supports reusability of the WSAN in form of NaaS.

The *WSAN-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 WSAN Cloud to be provisioned or de-provisioned rapidly, driven by *Orchestration Model for Service Provisioning (OMSP)* shown in Figure 1b. Orchestration allows the rapid provisioning of services in network devices, gateway and core to fulfill end-user requirements. In a traditional single tier WSAN 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 case are hard-coded. The reason for such practice lies in the fact that WSANs were never expected to offer cloud services, that is providing services for multiple end users. 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.

3.1 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. Rather than using mathematical model, an OMSP is instead based on simple system descriptive models that are used as drivers for orchestrating the rapid provisioning of a WSAN cloud. OMSP are a generic model for service provisioning, which can be written using XML. The OMSP can be divided into two segments - the OMSP template and the OMSP data. An OMSP template 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 template. An OMSP template provides a reusability mechanism for provisioning as it can be reused for provisioning on number of different networks. The OMSP data is related to the network itself and contains network dependent data. For example Figure 1b shows a pseudo description of an OMSP for reading light from sector 1 devices for a limited time. The OMSP template contains logic which can be applied to a number of different WSAN cloud infrastructures, where the OMSP data contains the data pertaining to the devices in all tiers of a specific WSAN cloud.

3.2 Orchestration Engine

The Orchestration Engine is a model (OMSP) driven system which uses the OMSP for provisioning the required services on the devices in the network (atomic service provisioning). The Orchestration engine consists of following components:

- Core
- Element Management System (EMS)

- NaaS Endpoint or middleware

3.2.1 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 for integration between NaaS and the end-users.

3.2.2 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 (3.3) in the orchestration engine to enable the EMS to execute the operations necessary to complete the service provisioning tasks. This is the entry point to the WSAN Cloud. Each 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 tier 1 devices. The EMS calls the gateway device to disseminate the service provisioning data to the network devices in the WSAN Cloud. 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 service to be instantiated. While a full service upgrade such as updating functionality is possible, the *WSAN-SOrA* does not recommend it due to the fact that devices in the WSAN run with limited energy resources. A Service upgrade can be compared to a full or part firmware upgrade in a high-end network. Similarly, a WSAN 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 a common practice as it causes disruption to the network service while a service upgrade is in progress.

3.2.3 NaaS Endpoint (middleware)

This component is an output product of the orchestration engine. Once the orchestration engine executes the OMSP for an end-user, a service endpoint is provided usually in the form of a web-service. The data from the network provisioned for a specific end-user is provided to the end-user through this service point.

3.3 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, the rest of the process becomes automated. The OMSP is first decoded and translated using the syntax knowledge base. If new syntax is required this knowledge base acts as a repository for developers to include new syntax. The knowledge base is a comprehensive database for the orchestration engine. It contains information such as data related to the OMSP syntax. With the OMSP being usually written in XML, this format acts as a standardised way of representing a document where the tags and attributes in the documents are meaningless unless there is a reference document. This data relates to the translation that is stored in the knowledge base. The knowledge base also stores the network data coming from devices and contains data pertaining to the available services for each device in the network and the current instance of those services. Service descriptions are written and saved in the knowledge base using XML based SDL (Service Descriptor Language) document. 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 calls the appropriate EMS gateway depending on the hardware platform. These gateways contain the base station device capable of communicating with network devices of the same type in the WSAN Cloud. The instruction is then disseminated in the WSAN Cloud and the services in the network are provisioned. Dissemination in the WSAN 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 (middleware) for the newly provisioned network so that the requesting system may extract its data. This can be done in the form of setting a webservice (passive polling), raw socket (stream), or servlets (push). De-provisioning is similar to provisioning, but services are de-instantiated instead of being created.

4. CURRENT DEVELOPMENT STATUS

A prototype of the WSAN-SOrA implementation was developed as an experimental WSAN Cloud infrastructure and configured using a heterogeneous mix of sensor device platforms consisting of SunSPOTs (4MB Flash) running the Squawk OS and TelosB (48KB Flash) devices running TinyOS or the Lorien OS. This prototype has been used to analyse the manual configuration of 30 devices for three different end-user systems and this is compared against the WSAN-SOrA orchestration approach. Table 1 shows the experimental results.

TinyOS running over the TelosB platform was used a baseline in this experiment to compare against WSN-SOrA. TinyOS is a monolithic OS and as such a complete OS image must be deployed for sensor configuration and re-tasking. Lorien and Squawk on the other hand are component based operating systems that support multi-threading and OOD making them compatible with WSN-SOrA. The size of NaaS endpoint middleware code is directly effected by the program memory available on the embedded network devices, where the NaaS endpoint middleware code size reflects the number of services that an implementation can hold. For example a NaaS endpoint middleware for SunSPOTs can hold services for temperature, light, humidity, multiple routing protocols and other aggregation services with the TelosB device offering temperature, light and humidity. The telosB devices running tinyOS were configured to support a temperature service and the NaaS endpoint middleware code size (in this case the complete OS image) was 8KB. For the telosB devices running Lorien, temperature and humidity services were configured with an initial code size of 38KB and finally for the SunSPOT devices temperature, light, humidity, rout-

Table 1: Quantitative Analysis

Platform	TelosB	TelosB	Squawk
Flash Memory	48KB	48KB	4MB
OS	TinyOS	Lorien	SunSPOT
OS Type	Monolithic, Event based	Component Multi-threading, SOA compatible	Multithreading SOA compatible
NaaS Endpoint Middleware Code-size	8KB	38KB	1.6MB
Initial Service(s)	Temp	Temp, Humidity	Temp,light, humidity, routing (AODV, LQRP), Accelerometer
Commission time(Serial)	<1s	<1s	<2s
Provisioning time	N/A, Monolithic image	127B* *depends on radio protocol IEEE 802.15.4 max frame size is 127bytes	127B
Provisioning Time(OTA)	N/A	<4ms	<4ms

ing and acclerometer services were configured with an initial code size of 1.6MB. The commissioning time depends upon the gateway program and serial connection with the device. In a batch commissioning operation, i.e. the deployment of the NaaS endpoint middleware, commissioning takes in the order of seconds and here the commissioning time was 1-2s and is dependent on the NaaS endpoint middleware code size. During the provisioning phase (i.e. the operational phase where the network can be reconfigured) of a WSAN-SOrA based infrastructure the provisioning packet size depends upon the communication standard in use for example IEEE 802.15.4 support a maximum frame size of 127 bytes and transmission rate of 250kbps and as such the provisioning packet maximum size is 127B. Provisioning is exclusive to systems that can support SOA and in this case this means that the devices running Lorien and squawk are capable of being provisioned using a single packet that is 127B in size. TinyOS follows a monolithic image deployment methodology and so a provisioning operation is not applicable in this case as deployment of the whole OS image is required each time (this is equivalent to redeploying the NaaS endpoint middleware repeatedly), which is analogous to performing a commissioning operation each time an update is required. The provisioning time depends upon the provisioning packet size, bandwidth and any disruption or interference in the radio communication between the wireless devices. Using over the air programming (OTA) the provisioning time was measured as being approximately 4ms. Provisioning in a device is instant as the devices are not required to do a restart in order to activate a new service. The need for a restart is heavily dependent on underlying OS. The orchestration of services in the proposed WSAN-SOrA approach allows for a fully expedited automated process without any human intervention. For WSANs configuration is typically done using web interfaces where parameter fields are set using hyper terminals or serial command line interfaces. These are once off 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 in terms of time or man power, whereas with WSAN-SOrA the OMSP process logic can be reused to provision similar systems. Configuration commands can be aggregated into a script and these scripts can be used to repeatedly configure the same type of device, but if the device type changes then the scripts must be updated using device compatible syntax whereas OMSP are device agnostic with the relevant EMS managing the device specific translation.

5. CONCLUSION

This paper presented the concept of a WSAN infrastructure as a WSAN Cloud that provides services to multiple application and data collection systems with an underlying architecture in the form of WSN-SOrA. WSAN-SOrA enables the rapid orchestration, commissioning and provisioning of services in WSANs, which can be vertically scaled on-demand. WSAN-SOrA is an architectural model that enables on demand access to a shared pool of configurable WSAN resources whose primary function is the management of the allocation and consumption of WSAN resources in order to serve multiple applications and end users alike.

6. REFERENCES

[1] Squawk for sunspot. Online, Jan. 2012.

[2] E. Avilés-López and J. A. García-Macías. Tinysoa: a service-oriented architecture for wireless sensor networks. *Service Oriented Computing and Applications*, 3(2):99–108, 2009.

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

[4] 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.

[5] C. Dagli and N. Kilicay. Understanding behavior of system of systems through computational intelligence techniques. In *Systems Conference, 2007 1st Annual IEEE*, pages 1 –7, april 2007.

[6] S. J. Habib. Analysis of sensors coverage through application-specific wsn provisioning tool. *IJMCMC*, 3(1):51–62, 2011.

[7] M. Kehoe et al. *Smarter Cities Series: A Foundation*

for Understanding IBM Smarter Cities. An IBM Redguide publication, 2011.

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

[9] 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*, IoTSP '11, pages 4:1–4:7, New York, NY, USA, 2011. ACM.

[10] Y. Lu, N. Huansheng, Y. Laurence T., and Y. Zhang. *THE INTERNET OF THINGS: From RFID to the Next-Generation Pervasive Networked Systems*. Auerbach Publications, 2008.

[11] 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*, pages 146 –153, june 2008.

[12] 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*, MidSens '09, pages 7–12, New York, NY, USA, 2009. ACM.

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

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

[15] J. Yick, B. Mukherjee, and D. Ghosal. Wireless sensor network survey. *Comput. Netw.*, 52:2292–2330, August 2008.