INOX: A Managed Service Platform for Inter-Connected Smart Objects

Stuart Clayman
Dept. of Electronic Engineering
University College London, London, UK
sclayman@ee.ucl.ac.uk

Alex Galis
Dept. of Electronic Engineering
University College London, London, UK
a.galis@ee.ucl.ac.uk

ABSTRACT

In this paper we present a service management platform and an architecture which integrates the features of IoT with the management features of modern autonomic network management, and many service features from the world of Services. We present an architecture for INOX, a robust and adaptable Service Platform for the Internet of Things and Inter-Connected Smart Objects. The platform integrates many of the ideas from Autonomic Network Management and Services and provides the functionality which allows for better use of the sensors, things and the smart objects through enhanced application development, more flexible service deployment, virtualized elements, and better service management. This paper presents the current status of our work on a reference framework for the management and integration of smart objects and virtual networks into such a service platform.

1. INTRODUCTION

The Internet of Things (IoT) is evolving from simple sensors with simple network connectivity into a collection of Inter-Connected Objects and Inter-Connected Smart Objects. For maximum benefit and usefulness of these connected objects we need to enable the integration of the functions of these inter-connected objects in the context of both user services and system management. In this paper we present a platform for IoT, which integrates the features of IoT with the management features of modern autonomic network management, and many service features from the world of Services (IoS). This integration encompasses network and service infrastructures, having resources with enhanced management capabilities, together with uniform interfaces.

A significant number of solutions and research activities fail to address some of the main issues of application and service development required for the full realisation of IoT into a fully connected Internet environment, as they are focussed on power issues, wireless networking, protocols, and end-to-end interaction [3]. Service platforms for IoT are a relatively new area of interest. In particular, the following topics need addressing in IoT service platforms: naming, identity, scalability, visibility for small objects with limited connection and computation capabilities, security, and finally orchestration and management of applications for millions of devices. A significant effort in platform design and architecture is needed to enable Internet-connected objects to become seamlessly integrated into services and important real-life stakeholder applications, thus avoiding the commonly heard complaint in IoT that there are too many *silos*. The main purposes of such a platform are (i) to extend the Internet with a large number of connected things and objects and (ii) to extend services to utilise thing and object-based resources.

We have seen that many of the deployed systems of IoT are unable to inter-operate with other IoT systems, even when deployed in the same physical environment. There is the well know effect of *silos*, *independent pillars* or *vertical systems* of systems [17], [21]. We have seen that many of the applications in the arena of IoT are fixed to one set of sensors. That is one application interacts with one set of sensors, and another application interacts with another set of sensors. For example, two systems in the same building will have 2 separate sets of sensors and actuators, and 2 independent control applications. It can be difficult to integrate them, or share data from both systems, to show in a single application. We need to overcome these limitations in the IoT arena, so that we can have sensors from multiple domains within the same applications. Such sharing and combining is commonly seen in many web-services data aggregation and mashup applications [16]. The limitation of this 2 layer approach has been understood, and efforts to extend the IoT model to a 3 layer approach

ACM IoTSP 2011, December 6, 2011, Tokyo, Japan.

are being taken. By making the control application for the sensors and things use an access end-point, more flexibility and adaptability is available. Although efforts such as ZigBee [10], are making some impact in this area, it is limited compared to a full Service Platform. Such efforts are a good start for inter-operability, but they do exhibit some scalability problems. Numerous projects in the area of Internet of Things (IoT), and Internet Connected Smart Objects have proposed connectivity architectures for the systems they develop [2] [20] [9] [15] [26]. All of these systems are faced with very similar problems in design of their service architecture.

The INOX platform presented here borrows ideas from the world of Internet Services and network management in order to benefit the world of IoT. As such, the sensor network is presented as a service element within a larger environment. The environment is managed by a management framework with advanced facilities, an example of this being UMF [25]. UMF In-network management is an approach where management and control functions are distributed and located in or close to the managed system and its elements. INOX represents a new approach with the following capabilities: (i) a new service and smart object based platform, (ii) the embodiment of enhanced management functionality in the platform, for maximising the efficiency of operation, (iii) the dynamic federation of different sensor networks, within the platform.

In the remainder of this paper we present the design approach used to integrate IoT with Internet services and network management, and then outline some of the main elements of the INOX platform. Then we present a testbed that has been developed to evaluate the architecture and the ideas outlined here.

2. BACKGROUND

IoT comprises a digital overlay of information over a highly heterogeneous physical world of objects. In the near future, such *objects* are expected to outnumber the human population by at least one order of magnitude. The IoT is expected to provide to the Internet, a resource fabric interfacing to the physical world, by means of a ubiquitously deployed substrate of embedded, connected, and networked devices. The resources provided are Near Field Communication (NFC) enabled objects, Radio Frequency Identification (RFID) tags, and smart objects that are small computers with communication capabilities as well as the sensors and/or actuators, etc.

Control systems, business services and end-user applications will use such resources and the interaction capabilities of the objects, with respect to real world entities. They will need to find the relevant data, context information, and then the resources that provide information about these entities and allow interactions with them.

As we have already seen, there are efforts in the IoT arena which are trying to address the usefulness of things and sensors *in the large*. These efforts are concentrating in 3 main areas: (i) the lower level communication frameworks; (ii) the mechanisms for integrating the things; and (iii) the frameworks for combining things for higher-level processing. Our work encompasses areas (ii) and (iii), but goes much further.

There is a fundamental need to allow service platforms to take advantage of the things and smart objects. There are on-going discussions regarding these efforts under the auspices of the IAB at the IETF conferences [24]. Moreover, the Internet is seen as a common infrastructure for inter-connecting networks, for interworking services, for inter-operating computing machines, and for the flow of information. Smart objects will require improvements in its general capabilities and its core system components. For services, it should inherently support a framework of general connectivity, mobility, security, Quality of Service with Service Level Agreements. As the network evolves to support a multitude of new devices, services, and applications, the general capabilities need to provide robustness and resilience, but also to provide inherent management in order to simplify the handling of networks for users [12]. Service models for ad-hoc networks have been addressed in previous work [18].

In the following sections we outline the context within which we propose the INOX platform as a solution for enhanced IoT applications.

2.1 IoT - Context

A key research theme is developing globally, which considers the concept of an Inter-Connected Object or an Internet of Smart Objects from the Internet of Things (IoT), as part of a federated Internet. The paradigms of Internet of Systems will enable the IoT arena to achieve the desired levels of dynamicity, efficiency, scalability, and economic incentives, in order to manage both the current and future services, together with networks of object-based resources that are becoming more and more pervasive and sophisticated. This is why there is a need to gather all of the relevant competencies to progress this field into maturity by generating high levels of industrial impact, maintaining a business-driven approach, and utilising the high-value of previous work.

Smart Objects are emerging in very large numbers as new Internet connectivity points, as well as being new resources for use by networks, services, and applications. As such, these Internet-connected objects are an integral part of the Internet and can be defined and viewed as a dynamic and global resource. To participate effectively in this view, they require a network and service infrastructure with self-management capabilities, based on inter-operable communications protocols, and enablers for rapid, cost-effective service and application deployment.

One important and essential integration element is the introduction of a virtualization mechanism that offers a virtualized view of the inter-connected object to the applications as well as views for supporting the virtualization of smart objects and their combination in virtual aggregations [13].

Virtualization has been employed effectively in networking environments [14], and in computing environments [4] as a mechanism to share out the same physical resource to more than one *user* in a concurrent way. To facilitate this, an information system and data model, plus the related services are required. Such an information system and data model would aid the interaction between applications and objects. These virtualized smart objects need to have identities, physical and virtual attributes, and use service interfaces to enable seamless integration into the information, context, and knowledge planes of the Internet.

The service interfaces for such objects would facilitate: (i) object interactions and interoperation over the Internet; (ii) the querying and changing of an objects state, information, and behaviour associated with it; (iii) event-based notification mechanisms and processing; (iv) applications and service deployment with seamless roaming across of the Internet, for mixed object and non-object resources.

2.2 Internet - Context

The current Internet has been founded and developed in the last 40 years on a basic architectural premise, that is: a simple network service can be used as a universal means to interconnect both dumb and intelligent end systems. The simplicity of the current Internet has pushed complexity into the end-points, and has allowed impressive scale in terms of inter-connected devices. However, while the scale has not yet reached its limits, the growth of functionality and the growth in size have both slowed down and may soon reach both its architectural capability and capacity limits. Internet applications increasingly require a combination of capabilities from traditionally separate technology domains to deliver the flexibility and dependability demanded by users. Internet use is expected to grow massively over the next few years with an order of magnitude more Internet services, the inter-connection of smart objects from the Internet of Things (IoT), and the integration of increasingly demanding enterprise and social applications.

Although the current Internet has been extraordinarily successful, as a ubiquitous and universal means for communication and computation, there are still many unsolved problems and challenges some of which have basic aspects. Many aspects leading to these problems could not have been foreseen when the first parts of the Internet were built, but these do need to be addressed now. The very success of the Internet is now creating obstacles to the future innovation of both the networking technology that lies at the Internets core and the services that use it.

We are faced with an Internet that is good at delivering packets, but shows a level of inflexibility at the network and service layers with a lack of built-in facilities to support any non-basic functionality.

There are some missing capabilities and solutions in the current Internet infrastructures which limit the integration of Inter-Connected Smart Objects[12]. The missing capabilities are as follows:

- *System aspects*: Service platforms and facilities, which take advantage of the sharing of resources via virtualization (including connectivity, computation, storage and object-based resources); On-demand provisioning of new functionality; Infrastructures for interconnection of smart objects; Inherent system management; Easy and efficienct deployment of both services and management.
- *Service aspects*: New applications and usage; Guaranteeing and facilities to support QoS and SLA; Knowledge-based society.
- *Enhanced aspects*: Appropriate addressing and naming for smart objects; Security, Trust and Privacy; enablers for Context-Awareness.
- *Economic aspects*: Cost considerations; economic viability of service offering including the need for appropriate incentives, diverse business models, legal, regulative and governance issues.

The list of features here is by no means exhaustive, but gives a flavour of what the issues are, if such large scale deployment of smart objects are to be harnessed into large scale applications.

2.3 Primary Requirements

From the areas of IoT, smart object, services and the Internet, we can derive some main requirements for the INOX platform. Similar requirements also apply to other service based environments, such as those of service cloud computing [23]. Service clouds have been successfully developed and deployed for large scale services, and have even been deployed in federated environments[22], in a way that is required for a successful IoT service platform.

The main requirements of the INOX platform are as follows:

- *Automated deployment*: The platform should support automated provisioning of service applications, possibly complex ones, based on a manifest which specifies the service elements and the run-time behaviour for QoS purposes.

- *Virtualisation technology*: The platform should support different virtualization technologies and in particular the virtualisation IoT resources.
- *Scalability*: ensure that the platform can cope with a large numbers of services and IoT resources.
- *Adaptability*: so that the platform can adapt to altered circumstances, including varying computational and network loads, to keep services running.
- *Federation*: so that any resource, real or virtual, which resides on another domain, can be used and managed correctly from a service.
- *Autonomic*: so that the platform can keep running without manual intervention, management, and reconfiguration.
- *Continuous optimization*: The platform should continuously optimize the alignment of infrastructure resources with management goals.

These main requirements help define the functionality and the features of the the platform.

3. THE INOX PLATFORM

This paper presents the INOX platform, which integrates elements from the world of Internet of Things, the world of Internet Services and network management, and unites them into a single combined world.

The goal of this platform design will be the development of a new architecture and infrastructure that enables the IoT services to be fully deployed as has previously been discussed. We expect it to have the following functionality and characteristics:

- Uniform service interfaces and business driven interfaces to enable generic integration of the interconnected smart object functions in the context of user services, taking special care of preserving their autonomous operation capability.
- Efficient linking of real-world entities with relevant resources of the IoT.
- Information and Context model and services to objects and applications.
- Service and self-management enablers for applications and service provisioning on the platform.
- Virtualisation of all resources: Object resources, Service Computation resources, Networking resources.
- Federation capabilities enabling service access across multiple domains.
- A security and privacy framework tailored to the needs of the IoT world.

This is a wide-ranging set of features, which is commonly seen in the management and services world [14] [12] [5], but is not common yet in the IoT world. We believe that as IoT service platforms become more pervasive more of these functions and characteristics will be seen.

First we consider the transition from both IoT world and the services world to the new INOX platform. This transition is depicted in figure 1.

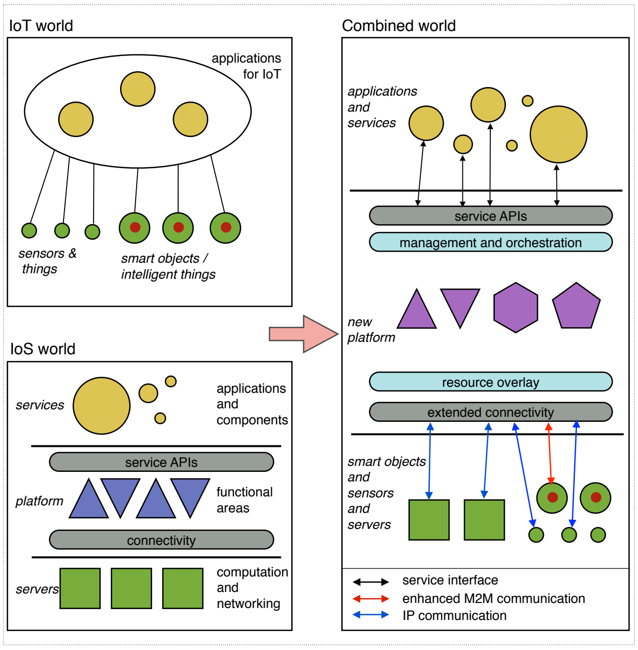

Figure 1: Combining IoT and Services Architectures

The top left part of figure 1 shows the common 2 layer IoT application environment where there are applications talking directly to the sensors and things, as well as to smart objects. The bottom left part shows the common services and management architectural model. We see 5 main aspects in this model: (i) the services themselves, (ii) service APIs, which services use to interact with the service platform, (iii) the service platform middleware with the main functional blocks, (iv) network connectivity, and (v) servers and routers that are used for computation and networking. The right hand side of figure 1 shows the new enhanced, combined, and integrated model. We see a primarily services model with the sensors and things from the IoT model pushed into the layer with the servers, and the applications from the IoT model pushed into the later with the services themselves.

Now we consider the platform in more detail, highlighting the layers and the main building blocks. The

INOX platform is split into 3 main layers. These include:

- *the service layer* - which supports and contains the services themselves. The services use the various service APIs in order to access the elements in the platform layer.
- *the platform layer* - which contains all of the necessary management and orchestration functions needed to build and deploy services and the virtualization technologies that will virtualize the elements in hardware layer, such as the smart objects.
- *the hardware layer* - which contains all of the sensors, things and smart objects which are part of an IoT environment, together with the servers that will provide the functionality for hosting services and virtual machines.

Figure 2, shows these presented main layers of the INOX platform. It also highlights the other main facilities.

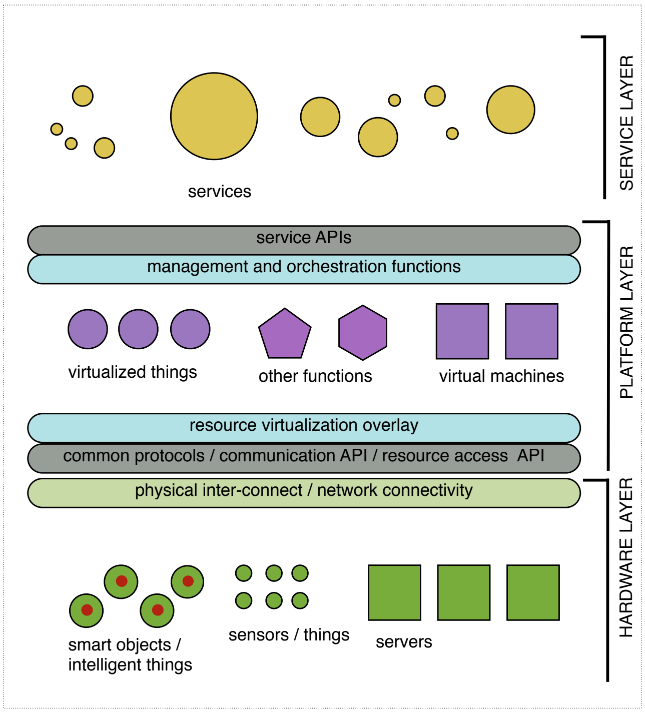
services
SERVICE LAYER
service APIs
management and orchestration functions
virtualized things
other functions
virtual machines
PLATFORM LAYER
resource virtualization overlay
common protocols / communication API / resource access API
physical inter-connect / network connectivity
HARDWARE LAYER
smart objects / intelligent things
sensors / things
servers

Figure 2: INOX: Multiple Platform Layers

Again we see the *service APIs* that are used by services. These service APIs are combined with the *management and orchestration functions*, and the *resource virtualization overlay*, shown in blue in Figure 2, plus main building blocks, shown in purple. The provided platform functionality consists of the following main functions and building blocks:

- a Registry and Discovery of the relevant things and smart objects. These processes utilize the following attributes: identifiers, location, type, provider, topic, or a combination of these.
- Lookup of resources that can provide information about the objects or allow interactions with them.
- Monitoring of objects and keeping the dynamic links between them up-to-date.
- Virtualisation of objects, networking and computational resources and linking virtual resources with real resources.
- Service enablers and self-* functionality (self-management, self-monitoring, self-configuration, self-optimisation, self-healing, self-protection and self-adaptation) which are part of the autonomic functions of the platform.
- Orchestration capabilities for controlling and managing the services within different domains.

The communication API and communication protocols for interacting with the real things and sensors are likely to be the specially devised protocols for the IoT area, including 6LowPAN [19], ZigBee [10], ROLL [1], or CoAP [11].

To aid the development of the INOX platform and to ensure inter-operability between the main components we define some special interfaces. The definition of such interfaces are:

service interfaces: to provide a rich set of service APIs to enable highly customized applications and software as service entities. These service interfaces will allow: (i) smart object interaction and interoperation over the Internet; (ii) the querying and updating of an objects state, information, and behaviour; (iii) event-based notification mechanisms and processing; (iv) application and service deployment for mixed object and non-object resources.

orchestration interfaces: to provide APIs to orchestrate and govern systems and virtual resources that meet stated business goals having specific service requirements. The purpose of the orchestration is to govern and integrate the behaviours of the systems and resources in response to changing context and in accordance with applicable high-level goals and policies. The orchestration supervises and integrates all other behaviour by ensuring the integrity of the management operations. It is responsible for organizing groups of resources in response to changing user needs, business requirements, and environmental conditions.

virtualisation interfaces: these mainly provide APIs that deal with virtual system setup and management issues. The APIs consist of methods for manipulating local network/service/storage/object resources abstracted as objects (i.e. as virtualized resources) or directly into the real resources (i.e. with no virtualisation). The abstraction isolates upper layers from hardware dependencies or other proprietary interfaces. The virtualisation interfaces isolate the diversity of setup and management requests from the actual control loop that executes them. They are responsible for determining what portion of a component (i.e. a set of virtual resources) is allocated to a given task. This means that all or part of a virtual resource can be used for each task, providing an optimised partitioning of physical resources according to business needs, priority, and other requirements.

comms interfaces: these APIs provide access to lower level resources. It is a collection of protocols that enable the exchange of state and control information at a very low level between different types of resources and the external agents of the resources. The resource types considered are: transport resources, forwarding resources, computation resources, storage resources, and content resources.

By having these interfaces defined, it becomes possible to build an INOX platform in a modular and flexible way, based on these interfaces.

3.1 Benefits

The main benefits that INOX brings to the IoT and smart object arena are:

- to evolve from centralized and fixed computing and networking architectures, towards dynamic architectures, based on shared resources using virtualisation and service management in the new platform. This is a shift in control, enabling significant sharing of resources for multiple application design and deployment.

- a reduction in the complexity of inter-networking M2M protocols, networks, devices and data w.r.t enterprise applications, networks, devices and data. This is because the platform deals with many of the complex issues.

- a migration from simple object lookup mechanisms towards standards and approaches to search and discovery for shared resources (e.g. objects, computation resources, networking resources)

- the migration from low-level M2M communication architectures to a service architecture, having a service-aware infrastructure with new communication interfaces towards all resources in the Internet.

- the introduction of a new virtualisation layer, which will facilitate and simplify the interaction between both services and end-user applications with the shared smart object resources. The sharing of smart object resources will be enabled by a virtualisation layer, and includes sensors, actuators, RFID devices, Near Field Communication devices.

As the INOX platform is designed in more detail, the interfaces fully specified, and all of the architectural elements built and tested, more of the benefits of the platform will become apparent.

4. TESTBED

To help verify the design and architecture of the INOX platform that has been outlined in this paper, various evaluations need to be carried out. It is difficult to access multiple live and real sensor networks and then add experimental new features and facilities as outlined in this paper. In order to undertake such evaluations and experiments, UCL is developing an early implementation of the INOX platform. This development is a small software-based service testbed - a Lightweight Service Platform. This testbed allows us to *emulate* all of the elements of a IoT deployment, combined with all of the relevant management functionality outlined for INOX. For example, each thing or sensor can be emulated using a software element. The benefits of our testbed include validating all of the management functions outlined in section 3, as well as testing for scalability, manageability, connectivity, and deployment of services.

To be as versatile as possible, the components in the testbed are very general. The testbed comprises of a very lightweight router combined with virtual network connectivity. These elements can be combined in order to build any network topology required. The created virtual network is designed with the goal of transmitting and routing datagrams from any source to any destination. It behaves like a lightweight datagram network, but it has management facilities to start and stop virtual routers on-the-fly, together with the ability to create and destroy network connections between virtual routers dynamically. Furthermore, these lightweight routers have an application layer interface that provide the capability to start and stop small Java software applications. These applications use a datagram API, which can send and receive datagrams, and thus act as the service elements within the platform. Such flexibility means that a whole virtual sensor/thing environment can be created to fit any need.

By using the Lightweight Service Platform testbed, it is possible to implement each of the emulated sensors as tiny applications. Each of the virtual routers in testbed is implemented within a Java Virtual Machine, and represents a sensor end-point, and each virtual link represents an emulated network link in the Internet. To deploy these virtual machines and their associated applications requires a set of physical machines. However, a physical machine, such as a server, can accommodate up to 70 virtual routers, which in our emulation environment would be equivalent to 70 sensor end-points.

In the Figure 3 we see how the elements of the INOX platform map onto the testbed. Each of the 3 main layers of the platform, namely: the service layer, the platform layer, and the hardware layer are present in the testbed.

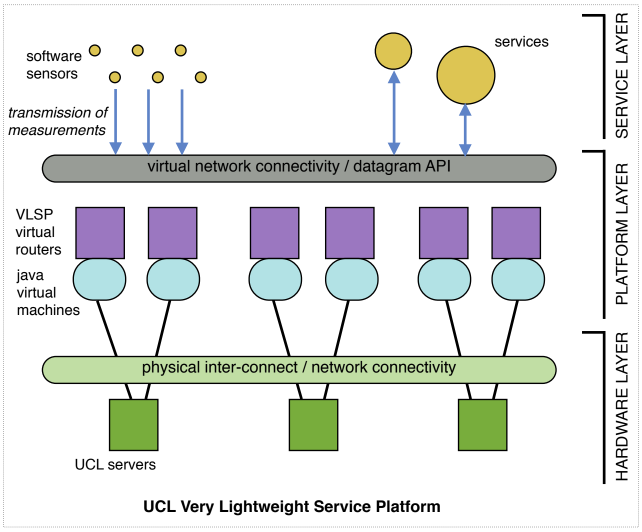
software sensors
services
transmission of measurements
virtual network connectivity / datagram API
VLSP virtual routers
java virtual machines
physical inter-connect / network connectivity
UCL servers
UCL Very Lightweight Service Platform
SERVICE LAYER
PLATFORM LAYER
HARDWARE LAYER

Figure 3: The VLSP Testbed

We have created a virtual sensor layer on the testbed by using the Lattice monitoring system [7]. This monitoring system behaves very similarly to a IoT environment and matches the behaviour of sensors very well, since both a resource constrained environment and mobility are emulated. Lattice is designed to be flexibile and adaptable, and has been also used in virtual networking environments [8].

To test the first parts of the functionality of the INOX system, we have built a Registry component and a Monitoring component. These has been coupled with a dynamic management component, in the style of UMF. The benefits of UMF are support for self-management features, higher automation and autonomicity capabilities, easier use of management tools and empowering the system with in-built cognition and intelligence. Additional benefits include reduction and optimisation in the amount of external management interactions, which is key to the minimization of manual interaction and the sustaining of manageability of large systems and moving from a managed object paradigm to one of management by objective. The current management system that allocates aggregation points around the network, based on dynamic traffic profiles from the virtual sensors. This work can be read in more detail in [6].

In order to create large test systems using the testbed, we utilise a large number of physical servers. In our experiments so far, at UCL we have executed over 700 virtual routers over 12 physical servers. For a very large scale test we will need a considerable number of physical servers to be available.

We will continue the work of developing the main building blocks of INOX within the testbed, and continue experimentation and validation of the INOX design.

5. CONCLUSIONS

Things and Smart Objects are emerging in very large numbers as new Internet connectivity points, as well as new resources for use by networks, services, and applications. As such, these Internet-connected objects are an integral part of Internet and would be defined as a dynamic and global resource. To participate effectively, they require a network and service infrastructure, based on interoperable communications protocols, and enablers for rapid, cost-effective service and application deployment.

How to build large scale, flexible and adaptable applications for these things and connected smart objects is an on-going discussion. In this paper we have presented a model and an architecture for things and smart objects that deals with the service and application deployment and autonomic management in an integrated virtualised network, commutation, storage and IoT resources. We believe that following the *service model* is an effective way to bring about the desired gains.

By having a platform that has the functionality of a service cloud, with virtualisation facilities and the ability to run shared applications, yet accommodates all of the things and smart objects from IoT, we can say that we are working towards an IoT cloud environment.

6. ACKNOWLEDGEMENT

This work is partially supported by the European Union UniverSELF project [25] of the 7th Framework Program.

7. REFERENCES

[1] Routing Over Low power and Lossy networks. `https://datatracker.ietf.org/wg/roll/charter/`.

[2] Advanced Sensors and lightweight Programmable middleware for Innovative RFID Enterprise applications, FP7. Aspire. `http://www.fp7-aspire.eu/`.

[3] Luigi Atzori, Antonio Iera, and Giacomo Morabito. The internet of things: A survey. *Computer Networks*, 54(15):2787 – 2805, 2010.

[4] P Barham, B Dragovic, K Fraser, S Hand, et al. Xen and the art of virtualization. *SIGOPS Oper. Syst. Rev.*, 37(5):164–177, 2003.

[5] L Cheng, A Galis, B Mathieu, K Jean, R Ocampo, L Mamatas, J Rubio-Loyola, J Serrat, A Berl, H de Meer, S Davy, Z Movahedi, and L Lefevre. Self-organising management overlays for future internet services. In *MACE 2008: Proceedings of the 3rd IEEE international workshop on Modelling Autonomic Communications Environments*, pages 74–89, Berlin, Heidelberg, 2008. Springer-Verlag.

[6] S Clayman, R Clegg, L Mamatas, G Pavlou, and A Galis. Monitoring, Aggregation and Filtering for Efficient Management of Virtual Networks. In *Conference on Network and Service Management - CNSM 2011*, 2011.

[7] S. Clayman, A. Galis, C. Chapman, G. Toffetti, L. Rodero-Merino, L.M. Vaquero, K. Nagin, and B. Rochwerger. Monitoring service clouds in the future internet. In *Towards the Future Internet - Emerging Trends from European Res earch*. IOS Press, April 2010.

[8] S. Clayman, A. Galis, and L. Mamatas. Monitoring virtual networks with lattice. In *Management of Future Internet - ManFI 2010*, 2010.

[9] EPCGlobal. EPCGlobal: The EPCGlobal Architecture Framework 1.3. `http://www.epcglobalinc.org/`, March 2009.

[10] S Farahani. *ZigBee Wireless Networks and Transceivers*. Newnes, Newton, MA, USA, 2008.

[11] B Frank, Z Shelby, K Hartke, and C Bormann. Constrained Application Protocol (CoAP). Technical Report draft-ietf-core-coap-07.txt, IETF Secretariat, Fremont, CA, USA, July 2011.

[12] A. Galis, H. Abramowicz, M. Brunner, D. Raz, P.R. Chemouil, J. Butler, C. Polychronopoulos, S. Clayman, H. de Meer, T. Coupaye, A. Pras, K. Sabnani, P. Massonet, and S. Naqvi. Management and service-aware networking architectures for future internet: System functions, capabilities and requirements. In *IEEE 2009 Fourth International Conference on Communications and Networking in China (ChinaCom09)*, August 2009. Invited paper.

[13] A. Galis, S. Clayman, L. Lefevre, A. Fischer, H. de Meer, J. Rubio-Loyola, J. Serrat, and S. Davy. Towards In-Network Clouds in Future Internet. In *The Future Internet - Future Internet Assembly 2011: Achievements and Technological Promises*, volume 6656, page 465 pp. Lecture Notes in Computer Science, May 2011.

[14] A. Galis, S. Denazis, A. Bassi, P. Giacomin, et al. *Management Architecture and Systems for Future Internet Networks*. IOS Press, http://www.iospress.nl, ISBN 978-1-60750-007-0, April 2009.

[15] IOTA. EU FP7 Internet of Things Architecture project. `http://www.iot-a.eu/public`.

[16] M Jarrar and M. D. Dikaiakos. A Data Mashup Language for the Data Web. LDOW2009, April 2009.

[17] 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*, WSNA '02, pages 88–97, New York, NY, USA, 2002. ACM.

[18] A. Malatras, G. Pavlou, and S. Sivavakeesar. A programmable framework for the deployment of services and protocols in mobile ad hoc networks. *IEEE Transactions on Network and Service Management*, 4(3):12–24, Dec. 2007.

[19] G Mulligan. The 6lowpan architecture. In *Proceedings of the 4th workshop on Embedded networked sensors*, EmNets '07, pages 78–82, New York, NY, USA, 2007. ACM.

[20] Cooperating Objects NoE. Cooperating Objects NoE, FP7. `http://www.cooperating-objects.eu/`.

[21] J Paek, K. Chintalapudi, R. Govindan, J. Caffrey, and S. Masri. A wireless sensor network for structural health monitoring: performance and experience. In *Proceedings of the 2nd IEEE workshop on Embedded Networked Sensors*, pages 1–9, Washington, DC, USA, 2005. IEEE Computer Society.

[22] B. Rochwerger, D. Breitgand, D. Hadas, I. Llorente, R. Montero, P. Massonet, E. Levy, A. Galis, M. Villari, Y. Wolfsthal, E. Elmroth, J. Caceres, C. Vazquez, and J. Tordsson. An architecture for federated cloud computing. *Cloud Computing*, April 2010.

[23] B Rochwerger, D Breitgand, E Levy, A Galis, et al. The RESERVOIR model and architecture for open federated cloud computing. *IBM Journal of Research and Development*, 53(4), 2009.

[24] H. Tschofenig and J. Arkko. Interconnecting Smart Objects with the Internet. Internet Draft 01, July 2011.

[25] Univerself consortium. Univerself project. `http://www.univerself-project.eu/`.

[26] O. Vermesan and P. Friess. *Internet of Things - Global Technological and Societal Trends*. River Publisher Series in Communications, 2011.