Cloud of Things

Marcus Vinícius Paulino Gomes

Instituto Superior Técnico
marcus.paulino.gomes@tecnico.ulisboa.pt

Abstract. Abstract should have 200 words at maximum.

1 Introduction

This section will describe the context of this theme, namely:

– What is the scope of the theme?
– What is the actual state of the theme?
– What are the main difficults confronted?

1.1 Internet of Things

The Internet of Things (IoT) is a concept in which the virtual world of information technology integrates seamlessly with the real world of things [1]. Through the amount of computer and network devices available nowadays, the real world becomes accessible to business as well everyday scenarios. A more precise definition of what is Internet of Things was formulated in the Strategic Research Agenda of the Cluster of European Research Projects on the Internet of Things (CERP-IoT 2009):

"*Internet of Things (IoT) is (...) a dynamic global network infrastructure with self configuring capabilities based on standard and interoperable communication protocols where physical and virtual 'things' have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network. In the IoT, 'things' are expected to become active participants in business, information and social processes where they are enabled (...) by exchanging data and information sensed about the environment, while reacting autonomously to the 'real/physical world' events and influencing it by running processes that trigger actions and create services with or without direct human intervention. Interfaces in the form of services facilitate interactions with these 'smart things' over the Internet (...) taking into account security and privacy issues.*"

IoT provides access to a more detailed information, which results that the level of analysis can be performed in a large-scale perspective, as well in a small-scale perspective. But Internet of Things is more than a tool for managing business processes more efficiently and more effectively, IoT will also enable a more convenient life for all peoples. Recently, IoT became relevant to industry and end-users[1], mainly because the reduction of cost and miniaturization of technologies used in IoT applications, such as RFID, sensor networks, NFC and wireless communication. Technologies like these allows the detection of status of things, which toghether with collection and processing of detailed data, allows to create an interactive and responsive network with huge potential for citizens, consumers, business and where is possible to gather immediate responses to events that occurs in the real world[1]. One of the expectations regarding of IoT concerns with real-world awareness provided to information systems [2]. An example is in logistics applications, by the use of RFID, companies can react promptly to relevant physical events, and then manage their processes in a better way, typically increasing efficiency and reducing costs. Another expectation of IoT concerns with providing services to the end-users through common objects, in that way products will be able to provide recommendations for use and maintenance instructions, supply warranty information or highlight complementary products.

1.2 Cloud Computing

This section will describe what is the concept of Cloud Computing? What is the most benefits of Cloud Computing? How Internet of Things could take advantage of Cloud Computing?

1.3 Smart Space

In the context of this work, a smart space can be defined as an ecosystem composed by smart objects such as RFID tags and sensors, that are able to acquire knowledge about this environment and also to adapt this inhabitants in order to improve their experience in that environment [3]. In particular, the deployment of IoT applications in a smart space is a challenge today due the heterogenity of the smart objects that are present in a smart space and also because the required infrastructure. In order to decrease the complexity of the deployment of an IoT application in a smart space, the Cloud computing paradigm allows to virtualize the required physical infrastructure. Nowadays the infrastructure needed by IoT applications can be virtualized by Cloud providers such as Amazon Web Services and Google Compute Engine, which helps to gain more flexibility and reduce the costs in the deployment of an IoT application. Therefore, another challenge concerns with the heterogenity of a smart space, more precisely the variety of smart objects that can be inside of a given space. Many of these objects uses different communication protocols and drivers to communicate with devices, the configuration of this objects must be manually handled, which makes the integration of these objects in the deployment of IoT applications in a smart space a inneficient process.

1.4 TOSCA

This section will introduce the TOSCA standard and its main components.

1.5 OpenTOSCA

OpenTOSCA is an open-source framework developed by the University of Stuttgart that provides an ecosystem for the OASIS Topology and Orchestration Specification for Cloud Applications (TOSCA) standard. OpenTOSCA allows to describe the components and relations of the applications, perform the orchestration and explicitly modeling management aspects of the applications. The components of OpenTOSCA will be described with further detail at the following sections.

OpenTOSCA Container OpenTOSCA Container is a runtime supporting imperative processing of TOSCA applications [4]. Imperative means that deployment and management services is realized through plans. OpenTOSCA main tasks consists in management operations, run plans and manage state. At Fig1 the architecture of OpenTOSCA Container is illustrated, as well a processing sequence regarding the instantiation of a Cloud application.

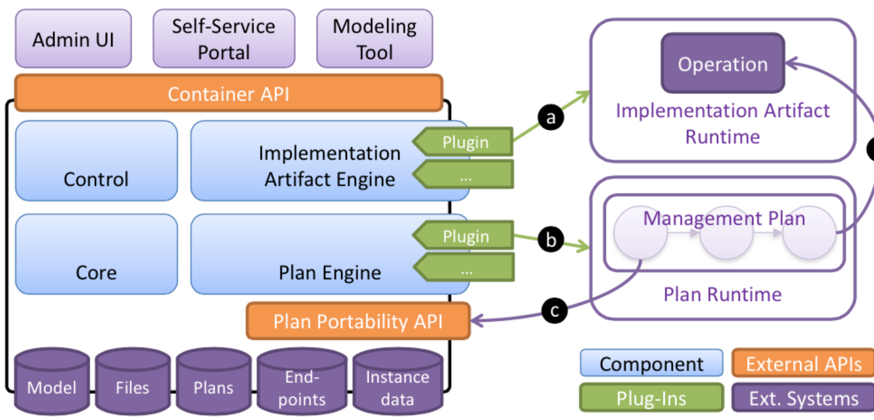

Fig. 1. OpenTOSCA Architecture Overview and Processing Sequence.

Requests to the Container API are redirected to the Control module, that is responsible to orchestrate the different components, track their progress and interprets the TOSCA application. The Core module provides common services to other components, e. g., managing data or validating XML. The Implementation Artifact Engine is responsible to run the Implementation Artifact contained in the CSAR in order to make them available for plans, usually these Implementation Artifacts provides management operations of nodes and relationships. The Plan Engine is responsible to process the management plans contained in CSARs, they also employs plugins to support different workflow languages, usually BPMN or BPEL, and their runtime environments. The portability of management plans between different environments and runtimes is ensured by the Plan Portability API. Plans only provides an abstract description of the required service, it is responsibility of the plan plugin to bind the services invoked by the plans to the endpoint of the management operation before it deploys the plan to the respective workflow runtime.

Winery Winery is web-based tool that enable modeling of TOSCA-based applications and creation CSARs in a tailored environment [5]. Its main features are type management and graphical topology modeling where the defined types are instantiated and interlinked. Winery architecture is composed essentially of three parts, the Topology Modeller, the Element Manager, and the Repository, where all data is stored, as illustrated in Figure 2.

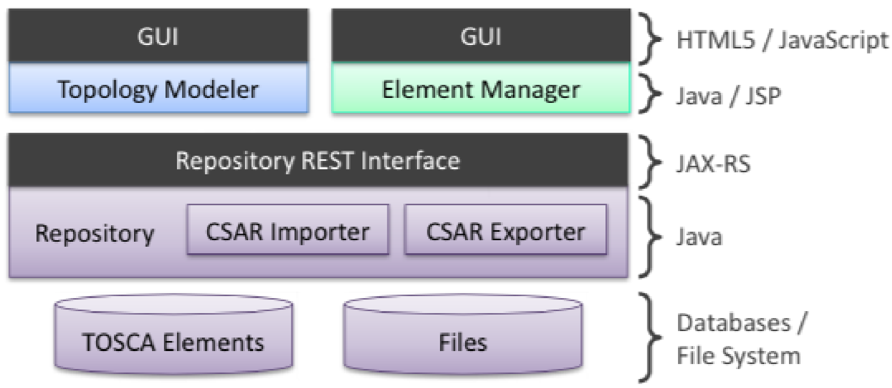

Fig. 2. Winery Architecture.

The Topology Modeller makes modeling of application easier by depicting element and combinations thereof visually, that allows to architects, developers and operators to understand the and model applications without the need for technical knowledge of type implementations and configurations, that is possible because the Topology Modeller handle only with the TOSCA meta models that are directly related to visual topology modeling, such as the node templates, relationship templates, deployment artifacts, requirements and policy. In the other hand the meta models that are used to define semantics and configurations such as types, implementations and policy templates are only created, modified and deleted exclusively by the Element Manager. By accessing the Element Manager, technical experts to provide and configure node types and relationships.

Vinothek Vinothek [6] is a web-based Self-Service portal that allows end-users to provision new TOSCA-based application instances through a simple graphical interface, as illustrated in Figure 3.
Vinothek abstracts the technical details and differences of different TOSCA Runtimes and provides a single interface to provision the applications. To perform the provision of the applications, Vinothek communicates with the server that delegates calls to the *TOSCA Application Lifecycle Manager* via a *RESTFUL API*, as illustrated on Figure 4.
Currently the *Application Lifecycle Manager* only supports dealing with the provisioning of applications, in the future it will support dealing with management and termination of applications. Another component of Vinothek is the *TOSCA Runtime Integration Layer* that is responsible to provide the mechanisms to

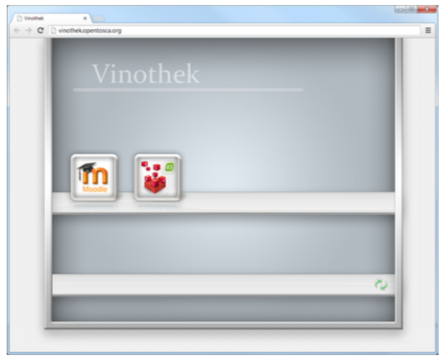

Fig. 3. Vinothek Self-Service Portal

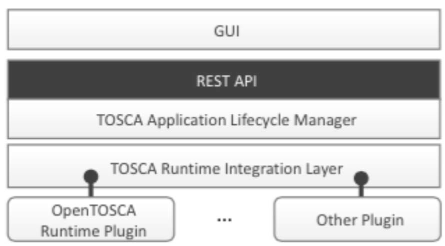

Fig. 4. Vinothek Self-Service Portal Architecture

plug-in TOSCA Runtimes. Although Vinothek facilitates the provisioning of applications, there are key mechanisms identified by the authors that currently Vinothek does not provide and are important in the application lifecycle such as performing management functionalities and processing policies to define security and other non-functional requirements.

1.6 Fosstrak EPCIS

This sections will describe what is the Fosstrak EPCIS framework. How this framework is relationed with IoT applications?

2 Objectives

As pointed out by Waizennegger et. al [7], TOSCA allows to specify policies that express non-functional requirements, like cost, security, and environmental issues. But TOSCA lacks a detailed description of how to apply, design and implement these policies. Thus the objective of this work is extend the OpenTOSCA ecosystem to make possible express business policies in order to define the business rules that are required by the IoT applications in a given context. More specifically, through the Vinotek Self-Service Portal the functional and non-functional requirements can be defined by tranlating a business policie from a more abstract level to a more specific level, for example, if a store has a flow of 500 clients per day, that can be translated that the IoT application requires a certain minimum value of network-bandwith and memory storage.

3 Related Work

Cloud and Internet of Things are emerging computing paradigms that features distinctively different computing resources and system architecture. As its popularity has been growing across the academics and the industry, researchers and developers are spending a lot of effort to investigate how to integrate these technologies in order to take advantage of the benefits provided by both of them.
IoT applications often encapsulate several relatively complex protocols and involves different software components. Moreover, they require a significant investment in infrastructure, besides that the system administrators and users spend time with large client and server installations, setups, or software updates. As most of the computing resources are allocated on the Internet on servers in Cloud computing, integrating these paradigms in a Cloud-based model results in a solution with more flexibility of implementation, high scalability and high availability, and with a reduced upfront investment.

In RFID-based IoT applications, Guinard at el. [8] point out that the deployment of RFID applications are cost-intensive mostly because they involve the deployment of often rather large and heterogeneous distributed systems. As a consequence, these systems are often only suitable for big corporations and large implementations and do not fit the limited resources of small to mid-size businesses and small scale applications both in terms of required skill-set and costs. To address this problem, Guinard at el. proposes a Cloud-based solution that integrates virtualization technologies and the architecture of the Web and its services. The case of study consists in a IoT application that uses RFID technology to substitute existing Electronic Article Surveillance (EAS) technology, such as those used in clothing stores to track the products. In this scenario they applied the Utility Computing blueprint to the software stack required by the application using the AWS platform and the EC2 service. The Elastic Cloud Computing (EC2) service allows the creation and management of virtual machines (Amazon Machine Images, or AMIs) that can then be deployed on demand onto a pool of machines hosted, managed and configured by Amazon. A benefit of this approach is that the server-side hardware maintenance is delegated to the cloud provider which is often more cost-efficient for smaller businesses. Furthermore, it also offers better scaling capabilities as the company using the EPC Cloud AMI, can deploy additional and more powerful instances regarding to the amount of requests.

Distefano [9] at el. proposed a high-level modular architecture to implement the Cloud of Things. According to Distefano at el. things not only can be discovered and aggregated, but also provided as a service, dynamically, applying the Cloud provisioning model to satisfy the agreed user requirements and therefore establishing Things as a Service providers. The *Things as a Service* (TaaS) paradigm envisages new scenarios and innovative, pervasive, value-added applications, disclosing the Cloud of Things world to customers and providers as well, thus enabling an open marketplace of "things". To address this issues, an ad-hoc infrastructure is required to deal with the management of sensing an actuation, mashed up resources provided by heterogeneous Clouds, and things, by exploiting well know ontologies and semantic approaches shared by and adopted by users, customers and providers to detect, identify, map and transform mashed up resources. The proposed architecture provides blocks to deal with all the related issues, while aiming to provide things according to a service oriented paradigm.

CloudThings [10] is an architecture that uses a common approach to integrate Internet of Things and Cloud Computing. The proposed architecture is an online platform which accommodates IaaS, Paas, and SaaS and allows system integrators and solution providers to leverage a complete IoT application infrastructure for developing, operating and composing IoT applications and services. The applications consists of three major modules, the CloudThings service platform, that is a set of Cloud services (IaaS), allowing users to run any applications on Cloud hardware. This service platform dramatically simplifies the application development, eliminates need for infrastructure development, shortens time to market, and reduces management and maintenance costs. The CloudThings Developer Suite is a set of Cloud service tools (PaaS) for application development, such as Web service API's, which provide complete development and deployment capabilities to developers. The CloudThings Operating Portal is a set of Cloud services (SaaS) that support deployment and handle or support specialized processing services. To evaluate CloudThings, a smart home application based on a Cloud infrastructure was implemented. In the application, the sensors read the home temperature and luminosity and the Cloud application stores and visualized them, so that the user can view the smart home temperature and luminosity anywhere. In particular, in these implementation the Cloud architecture was extended by inserting a special layer for dynamic service composition. This middleware encapsulates sets of fundamental services for executing the users service requests and performing service composition, such as process planning, service discovery, process generation, process execution, and monitoring. This first implementation also demonstrates that this middleware as a service releases the burden of costs and risks for users and providers in using and managing those components.

The effort put in the research to integrate the paradigms of Cloud Computing and Internet of Things resulted in a essential contribution, but there are several issues regarding to the integration between Cloud Computing and Internet of Things that must be addressed. In particular, due of the heterogeneity of the IoT applications environments, its hard for solution providers to efficiently deploy and configure applications for a large number of users. Thus, automation for the management tasks required by IoT applications is a key issue to be explored.

TOSCA (Topology and Orchestration Specification for Cloud Applications) [11] is proposed in order to improve the reusability of service management processes and automate IoT application deployment in heterogeneous environments. TOSCA is a new cloud standard to formally describe the internal topology of application components and the deployment process of IoT applications. The structure and management of IT services is specified by a meta-model, which consists of a *Topology Template*, that is responsible to describe the structure of a service, then there are the *Artifacts*, that describes the files, scripts and software components necessary to be deployed in order to run the application, and finally the *Plans*, that defined the management process of creating, deploying and terminating a service. The correct topology and management procedure can be inferred by a TOSCA environment just by interpreting te topology template, this is known as "declarative" approach. Plans realize an "imperative" approach that explicitly specifies how each management process should be done. The topology templates, plans and artifacts of an application are packaged in a Cloud Service Archive (.csar file) and deployed in a TOSCA environment, which is able to interpret the models and perform specified management operation. These .csar files are portable across different cloud providers, which is a great benefit in terms of deployment flexibility. To demonstrate its feasability TOSCA was used to specify a typical IoT application in building automation, an Air Handling Unit (AHU). The commom IoT components, such as gateways and drivers will be modelled, and the gateway-specific artifacts that are necessary for application deployment will also be specified. By archiving the previous specifications and corresponding artifacts into a csar file, and deploying it in a TOSCA environment, the deployment of AHU application onto various gateways can be automated. As a newly established standard to counter growing complexity and isolation in cloud applications environments, TOSCA is gaining momentum in industrial adoption as well academic interests.

Breitenbücher at el. [12] proposed to combine the two flavors of management supported by TOSCA, *declarative processing* and *imperative processing*, in order to create a standards-based approach to generate provisioning plans based on TOSCA topology models. The combination of both flavors would enable applications developers to benefit from automatically provisioning logic based on declarative processing and individual customization opportunities provided by adapting imperative plans. These provisioning plans are workflows that can be executed fully automatically and may be customized by application developers after generation. The approach enables to benefit from strengths of both flavors that leads to economical advantages when developing applications with TOSCA. The motivating scenario that is used to evaluate this approach consists in a LAMP-based TOSCA application to be provisiones. The application implements a Web-shop in PHP that uses a MySQL database to store product and customer data. The application consists of two application stacks, one provides the infrastructure for the application logic and the other hosts the database, were both will be run on Amazon's public IaaS oferring Amazon EC2. To measure the performance of the deployment using the two-flavor approach, the strategy addopted was measure the time spent to generate provisioning plans regarding the number of templates required by the application. The results indicates that the required time increases linealy to the number of templates.

Waizenegger at el. [7] point out that the policies ...

Recently a growing number of organizations are developing Orchestrators, Design Tools and Cloud Managers based on TOSCA. Juju is an Open Source TOSCA Orchestrator that can deploy workloads across public, private clouds, and directly onto bare metal. HP Cloud Service Automation is cloud management solution that supports declaratives services design that are aligned with TOSCA modeling principles. GigaSpaces Cloudify orchestrates TOSCA Service Templates using workflows to automate deployments and other DevOps automation processes. IBM Cloud Orchestrator provides integrated tooling to create TOSCA applications, deploy them with custom polices, monitoring and scale them in cloud deployments.

4 Solution Architecture

This section will describe the solution architecture developed to achieve the proposed objectives.

5 Evaluation Methodology

This section will describe the methodology to evaluate the proposed work.

6 Conclusion

This section will present the general conclusion of the work realized.

Appendix

References

1. D. Uckelmann, M. Harrison, and F. Michahelles, *Architecting the Internet of Things*. Springer Publishing Company, Incorporated, 1st ed., 2011.
2. F. Mattern and C. Floerkemeier, "From the internet of computers to the internet of things," in *From active data management to event-based systems and more*, pp. 242–259, Springer, 2010.
3. D. Cook and S. Das, *Smart environments: technology, protocols and applications*, vol. 43. John Wiley & Sons, 2004.
4. T. Binz, U. Breitenbücher, F. Haupt, O. Kopp, F. Leymann, A. Nowak, and S. Wagner, "OpenTOSCA – a runtime for TOSCA-based cloud applications," in *11th International Conference on Service-Oriented Computing*, LNCS, Springer, 2013.
5. O. Kopp, T. Binz, U. Breitenbücher, and F. Leymann, "Winery – modeling tool for TOSCA-based cloud applications," in *11th International Conference on Service-Oriented Computing*, LNCS, Springer, 2013.
6. U. Breitenbücher, T. Binz, O. Kopp, and F. Leymann, "Vinothek–a self-service portal for tosca," in *Workshop Proceedings*, p. 72, 2014.
7. T. Waizenegger, M. Wieland, T. Binz, U. Breitenbücher, F. Haupt, O. Kopp, F. Leymann, B. Mitschang, A. Nowak, and S. Wagner, "Policy4tosca: A policy-aware cloud service provisioning approach to enable secure cloud computing," in *On the Move to Meaningful Internet Systems: OTM 2013 Conferences*, pp. 360–376, Springer, 2013.
8. D. Guinard, C. Floerkemeier, and S. Sarma, "Cloud computing, rest and mashups to simplify rfid application development and deployment," in *Proceedings of the Second International Workshop on Web of Things*, p. 9, ACM, 2011.
9. S. Distefano, G. Merlino, and A. Puliafito, "Enabling the cloud of things," in *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2012 Sixth International Conference on*, pp. 858–863, IEEE, 2012.
10. J. Zhou, T. Leppanen, E. Harjula, M. Ylianttila, T. Ojala, C. Yu, and H. Jin, "Cloudthings: A common architecture for integrating the internet of things with cloud computing," in *Computer Supported Cooperative Work in Design (CSCWD), 2013 IEEE 17th International Conference on*, pp. 651–657, IEEE, 2013.
11. F. Li, M. Vogler, M. Claeßens, and S. Dustdar, "Towards automated iot application deployment by a cloud-based approach," in *Service-Oriented Computing and Applications (SOCA), 2013 IEEE 6th International Conference on*, pp. 61–68, IEEE, 2013.
12. U. Breitenbücher, T. Binz, K. Képes, O. Kopp, F. Leymann, and J. Wettinger, "Combining declarative and imperative cloud application provisioning based on tosca," *IC2E. IEEE*, 2014.