Challenges and Assessment in Migrating IT Legacy Applications to the Cloud

Patrizia Scandurra, Giuseppe Psaila
University of Bergamo,
Bergamo, Italy
{patrizia.scandurra, giuseppe.psaila}@unibg.it

Rafael Capilla*, Raffaela Mirandola**
*Rey Juan Carlos University, Madrid, Spain
**Politecnico di Milano, Milano, Italy
rafael.capilla@urjc.es, raffaela.mirandola@polimi.it

***Abstract*—The incessant trend where software engineers need to redesign legacy systems adopting a service-centric engineering approach brings new challenges for software architects and developers. Today, engineering and deploying software as a service requires specific Internet protocols, middleware and languages that often complicate the interoperability of software at all levels. Moreover, cloud computing demands stringent quality requirements, such as security, scalability, and interoperability among others, to provide services and data across networks more efficiently. As software engineers must face the problem to redesign and redeploy systems as services, we explore in this paper the challenges found during the migration of an existing system to a cloud solution and based on a set of quality requirements that includes the vendor Lock-in factor. We also present a set of assessment activities and guidelines to support migration to the Cloud by adopting SOA and Cloud modeling standards and tools.**

***Index Terms*— Cloud computing, service-oriented computing, cloud migration, service-oriented architecture, service-centric engineering, cloud modeling, portability, interoperability, vendor Lock-in.**

I. Introduction

In a globalized world, the delocalization of hardware resources, third-party software, and economies of scale are main reasons that drive the shift for migrating existing core business applications into a service-based or cloud system [6]. A significant number of studies for cloud migration and modernization reported in [8] demonstrate the benefits for cloud adoption.

Nowadays, two major trends steer the migration and development of Internet-based applications: Service-oriented Architectures (SOAs) and cloud computing, both perceived as complementary but overlapping technologies for building modern decentralized systems. The today's challenges to build service-oriented software applications over the Cloud must address a number of quality concerns and end-to-end secure solutions [21] able to manage and deploy IT business services[1]. SOA offers good choice for developing reusable and interoperable services when legacy systems are reengineered using a Web front-end, while cloud technology offers elasticity and scalability of services at lower costs [17].

However, the variety of requirements that drives the selection of a particular SOA and Cloud solution and the interoperability factors between legacy software and services may complicate the adoption of a particular service-centric approach in an industrial setting. Based on our consulting experience, this paper presents through a running case study the challenges a company has to face the migration of an existing software application to a cloud environment. The migration case study should take into account specific quality requirements and the possibility to add to the legacy application new functionalities using third-party Cloud-native services. As part of our lesson learned, we also present a set of general assessment activities and guidelines to support the migration to the Cloud by adopting SOA and Cloud modeling standards and tools.

This paper is organized as follows. Section II presents the case study used throughout the paper and the specific requirements that motivated the adoption of a cloud solution. Section III describes the design challenges we faced during the migration case study. Section IV discusses the solutions adopted during the migration to the Cloud, and Section V presents the set of activities and guidelines for migration assessment. Finally, we discuss in Section VI the related work and Section VII draws our conclusions.

II. Motivating Scenario

In this section, we describe the case of an Italian food trading company (IFT) that adopted a migration strategy towards a service-centric approach using Cloud technology. IFT was formerly equipped with a centralized non-web solution hosted on an IBM AS400, providing functionalities for orders, stock management and fiscal duties; employees working in two warehouses and several stores operate on the system through remote terminals. Due to the international expansion of its core business, IFT made a deal with a third party company (named CC) to provide a call-center service using a Customer Relationship Management (CRM) platform, an e-commerce web site, and a Business Intelligence (BI) platform to support decision-making based on historical data

[1] The use of "service" for SOA and cloud computing is similar but exhibit some differences. Services using a SOA approach can be seen as a self-contained unit of functionality which is exposed and invoked using standard Internet protocols. However, when we adopt a cloud model, the meaning of service may vary according to the SaaS-PaaS-IaaS layer. Services at the IaaS layer offer computing resources to the PaaS layer while at the SaaS level these services can be comparable to SOA services. At the PaaS level an entire computing platform (e.g., Google App Engine) is offered as a service. Scalability and elasticity on demand of cloud services is a typical characteristic that differ from IT SOA services.

analysis. The evolution of IFT systems into a cloud solution triggered the following non-functional requirements [5] [19]:

(1) *Scalability*: The internationalization of the business demanded a rapid growth of transactions yielding to scalability problems that had to be addressed with new computational resources.

(2) *Composability*: The requirement was motivated by the need to extend the original application with (possibly) external heterogeneous systems and services, like the CRM and BI platforms.

(3) *Portability*: Portable software and systems are easy to migrate, but often complex and sophisticated systems based on proprietary technology might not be portable. The company evaluated the drawbacks for portability and adoption of proprietary and ready-to-use services for developing the new system's functions.

(4) *Interoperability*: The composability requirement implies interoperability, because IFT will provision computational resources and software services from different cloud providers and these heterogeneous services must be wired together.

III. Cloud Design Challenges

The requirements stated previously by IFT software engineers affect the migration strategy and led to a number of design and modeling challenges. However, the migration towards a cloud strategy adopted by IFT should not affect the entire core functionalities of the company's systems. The depth impact of the design changes might be different depending on the degree of migration: total rewriting of the application or partial migration by components replacement with SaaS components equivalent and components relocation to the cloud environment. Moreover, the cloud layers affected (e.g., data and business layers or cross-cutting concerns affecting to various layers) can be many [1] if cloud agnostic applications are distributed across multiple cloud platforms [4]. IFT engineers reported the following challenges as important modeling concerns needed to achieve the necessary interoperability as part of the migration strategy.

Challenge c1. *Separation of concerns at different abstraction levels*: The Cloud stack defines a clear separation between high-level business services, middleware and infrastructure. Therefore, SOA and Cloud architecture principles guide the migration strategy when legacy software is ported to the cloud by replacing the functionality easier to migrate (e.g., use Heroku relational database service in place of a local SQL DBMS). Multi-tenancy becomes relevant to achieve the necessary scalability of cloud solutions [12] while interoperability of services seems an important modeling concern for migrating legacy systems.

Challenge c2. *Service design and composability*: Many important design principles and quality concerns such as reusability and composability guide the design principles of web services [12] and the migration strategy to a service-centric approach. The design and deployment of existing applications in the Cloud demand appropriate languages and notations to model and deploy services. Prior to the application design, a key sub-challenge is about "candidate service identification" and "granularity" of the services [10]. Once the services are identified and the granularity is determined, the notion of service modelling becomes important. Moreover, composability of services is a requirement that drives wiring and rewiring services. Composition paves the way to build SOA and Cloud applications and face the challenge of interdisciplinary development methods and distributed platforms [5]. Therefore, designing composite and loosely coupled services facilitates the encapsulation and reuse of services.

Challenge c3. *Vendor Lock-in barrier for Portability and Interoperability:* Porting IT systems to the Cloud or using third-party services require interoperable services and platforms, but in some cases the "vendor Lock-in" [14] barrier may affect the solution adopted for compatibility or business reasons. In such cases using proprietary software may hamper the adoption of certain solutions when third-party services are used. The vendor Lock-in factor affects also the cost and the selection of the middleware and PaaS that must be described in the architecture.

Challenge c4. *Cost factors:* Migration to a service-centric model has a noteworthy impact on the cost of software systems (e.g., cost of software licenses, cost renting, cloud pricing models). Porting applications to the Cloud is many times a balance between the desired qualities and cost. Cost has a visible impact on the selection of a particular cloud platform or SOA middleware, the cost for building own services or renting third-party ones, and for selecting a commercial or open source modeling language that in some case has to be aligned with the business strategy of the company.

IV. Assessing IFT on the Migration Strategy

This section discusses the modeling solutions adopted during the migration to a portable Cloud solution for IFT Company using third-party Cloud-native services. First, we describe which requirements address partially each challenge for each particular solution. Second, we outline the modeling alternatives explored by the company.

A. Cloud requirements and solutions adopted

Hereafter we discuss the solutions adopted to address the requirements of the case study for each challenge of Section III.

(1) ***Scalability***: Scalability can be addressed by cloud services, because the provisioned computational resources can change dynamically based on the current workload of the existing application. In this respect, IFT evaluated the following two alternatives: (i) rewrite the running system over AS400 for a cloud specific PaaS middleware, or (ii) migrate the system to a Cloud Infrastructure. In order to keep low the migration costs and to achieve a quick portability while preserving the user experience for IFT users, the company decided to migrate the legacy AS400-based application and database to IBM Cloud services which offer cloud hosting

(IaaS) for AS400 applications. The scalability requirement addresses challenge *c1* and partially *c4*.

(2) ***Composability***: IFT evaluated two alternatives to extend the original AS400 application with CRM and BI systems. (i) The first one is to rent two Cloud-based hosts (IaaS), purchase CRM and BI software, install it on the hosts, customize it, maintain the software, and periodically install updates; (ii) the second alternative is to use CRM and BI cloud services (SaaS) and compose them with the original AS400 application (yet migrated to the cloud IaaS). Typically, alternative (ii) significantly reduces the total ownership cost as the company pays for the service while it reduces the maintenance costs. Moreover, solution (ii) takes advantage from elasticity offered by IaaS slightly re-architecting the legacy application at the infrastructure level. Well conscious of the vendor Lock-in problem (challenge *c3*), IFT adopted Salesforce.com because considered as one of the best SaaS CRMs available on the market. Besides, its Web-based user interface facilitates the interaction between internal CRM staff and company CC. IFT chose GoodData (a Cloud-based Data Warehouse and BI service that provides a PaaS solution) for BI (http://www.gooddata.com/) as it enhances the composition of cloud services for reporting and data analysis tasks. Finally, the cloud solution selected for the E-commerce web site was the Heroku PaaS, because it offers a Cloud-native platform for web sites based on standard technologies (e.g., like Java, PHP, Python). These solutions were proven effective in terms of quick availability of the new functionalities and for cost reduction. This requirement addresses challenges *c1* and *c2,* and has influence on challenge *c4*.

(3) ***Portability***: As the Heroku platform uses standard web technologies it suitably addresses portability needs. By contrast, Salesforce.com is a pre-built easily customizable CRM SaaS-PaaS solution based on proprietary and non-portable technology (e.g., the Apex programming language), and its adoption will impact negatively on the overall portability of the IT system, even if it provides other benefits such as the quick set-up of services that can be easily configured and deployed using the CRM without writing code and saving development costs. Similar considerations stand for the selection of GoodData as preferred BI solution. Portability satisfies partially challenges *c3* and *c4*.

(4) ***Interoperability***: The alternatives and choices discussed before were evaluated also for the interoperability requirement. IFT observed that interoperability was not difficult to address since SaaS and PaaS services can share data using a REST communication API to avoid typical isolation problems of proprietary solutions like legacy AS400 systems. This requirement satisfies challenge *c3*.

Figure 1 shows the topology of the solution adopted as a result of the decisions made and the alternatives considered by IFT engineers.

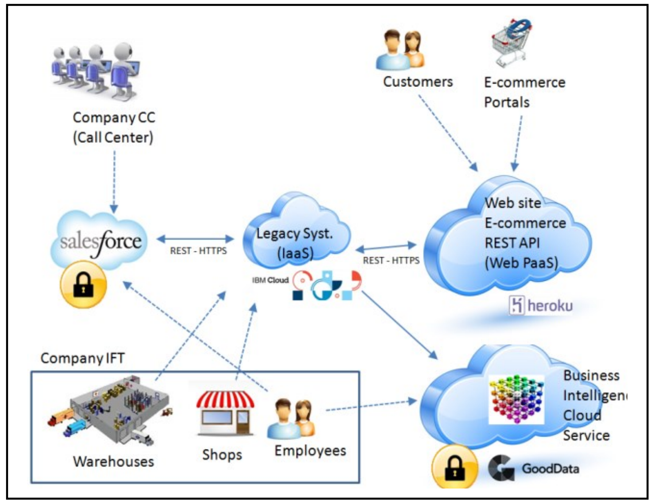

Fig 1. IFT Information System topology

B. Analysis of cloud service modeling alternatives

The migration of IFT's legacy system to the Cloud involved a number of evaluation issues from the designer's perspective and was based on the set of requirements stated by the IFT software engineers. The aforementioned design challenges brought the need to count with suitable modeling notations, languages and tools able to facilitate the development and migration to a service-centric approach.

IFT analyzed existing modeling notations for SOA and Cloud and evaluated the pros and cons of each one at the SaaS and PaaS layers. Table 1 summarizes the results of the evaluation carried out at IFT which served for the selection of a specific Cloud modeling notation to design the services and resources used for the adoption of the cloud model. IFT software engineers conveyed that all SaaS modeling solutions are almost complementary to support orchestrated services. In particular, IFT experimented in practice the OASIS standards SCA and TOSCA because they look promising modeling alternatives that also address interoperability and portability issues [13] [2]. SCA focus on SaaS architectural aspects, and addresses interoperability and portability by allowing a wide range of component implementation types and communication protocols for wiring services.

TOSCA describes a cloud service's topology and its management aspects (such as how to deploy, terminate, and manage a service) at SaaS-PaaS cloud layers. Through "plans" implemented using existing workflow technologies such as Business Process Model and Notation (BPMN) or the Business Process Execution Language (BPEL), TOSCA service topologies can be interpreted by any Cloud platform supporting a TOSCA-compliant management environment that operates the cloud services and manages their instances by executing plans.

TABLE I. EVALUATION OF CLOUD SAAS-PAAS MODELING ALTERNATIVES

Modeling approaches	Quality requirements for Cloud	Notation & tool support	Pros	Cons
SaaS level solutions SoaML[2] SOMA-SOMF[3] CCMN for SOMF SCA[4]	Service composability QoS policy specification and contracts Portability Interoperability	Graphical UML-based and XML textual notations **Tools available:** SOMA-ME for IBM RSA Sparx Systems' Enterprise Architect Papyrus IBM RSA Visual Paradigm OFTEAM Modelio MagicDraw IntelliJ SCA Eclipse SCA plug-in Apache Tuscany FraSCAti Oracle WebLogic IBM WebSphere	Graphical notation for modeling services with extensions for modeling cloud systems Large number of commercial tools Suitable for software designers and connection to other business modeling notations like BPMN Dynamic component binding	Lack of standardized notations Some OSS tools or plug-in do not support well the composition of services Cost of commercial solutions
SaaS-PaaS level solutions TOSCA[5] CloudML[6]	Portability Interoperability Adaptability Scalability QoS policy specification and contracts	Graphical and XML and JSON textual notations **Tools available:** OpenTOSCA runtime ecosystem and Winery modeling tool IBM SmartCloud and SINTEF's CloudML engine	Graphical notation for modeling services and supported by research and commercial tools Wiring service facilities and interoperability with communication protocols Integration with several languages and platforms Describe the overall topology of cloud applications Dynamic configuration and portable workflows	Releases of open source solutions not well supported CloudML lacks for instance of mature MDE techniques for migration to cloud services

[2] http://www.omg.org/spec/SoaML/1.0.1/
[3] http://www.sparxsystems.com/somf
[4] OASIS's standard Service Component Architecture (SCA) http://oasis-opencsa.org/sca
[5] OASIS's Topology and Orchestration Specification for Cloud Applications (TOSCA) http://docs.oasis-open.org/tosca/TOSCA/v1.0/cs02/TOSCA-v1.0-cs02.pdf
[6] http://cloudml.org/

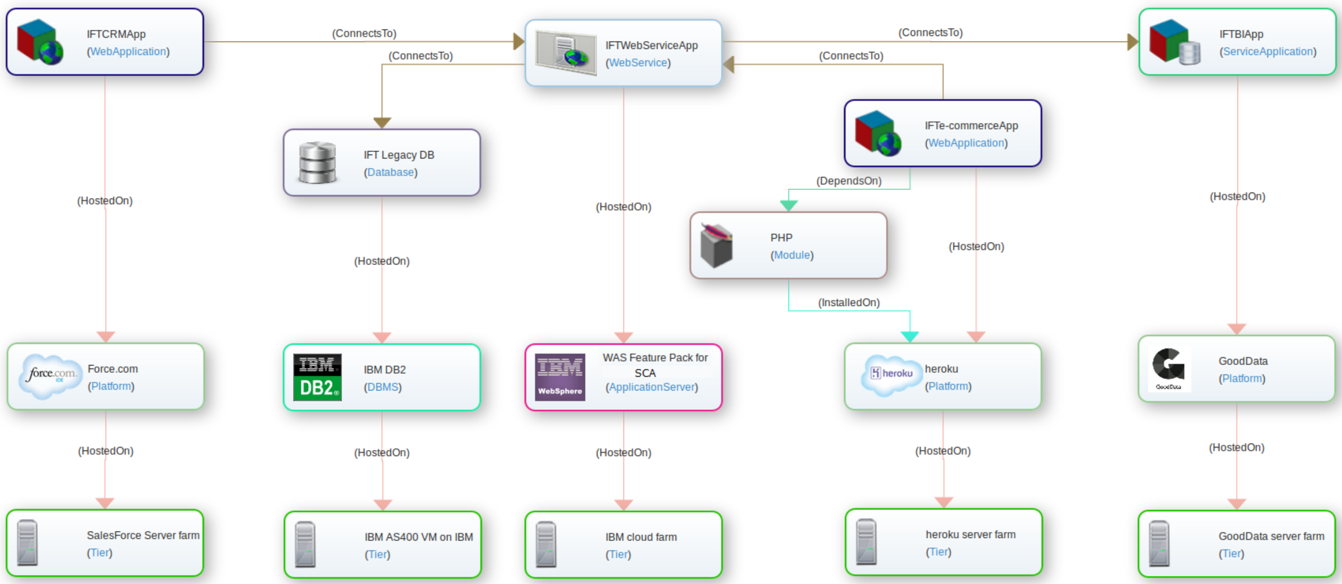

Fig 2. TOSCA IFT service topology diagram

Moreover, service composability is one of the major quality concerns addressed by these languages, along with QoS policies and enforcing (via policy-specific plug-ins) these policies under a runtime environment.

After this preliminary comparison, IFT conducted an experiment using TOSCA and SCA. Initially, IFT adopted TOSCA as modeling notation to describe and deploy the new application topology and understand the interoperability and heterogeneity of services running on hybrid clouds, alongside with their implementation and deployment artifacts. Figure 2 sketches an excerpt of the TOSCA topology of the Cloud solution analyzed by IFT using the visual notation Vino4TOSCA provided by the modeling tool Winery. IFT carried out some experiments using the TOSCA runtime environment, such as the one implemented in OpenTOSCA (http://www.iaas.uni-stuttgart.de/OpenTOSCA/) and in IBM Cloud. These TOSCA-compliant environments are still prototypes and the results did not satisfied IFT engineers due to the low tool maturity level and the low adoption by the other cloud partners involved by IFT. Besides the third-party Cloud-native services described in Figure 2, IFT adopted also the standard SCA to design and implement a web service application as wrapper on the top of the legacy IFT's information system.

SCA provides facilities for different component implementation types like Java and EJB and REST communication bindings. The services wrapper, called IFT Web Service App (see Fig. 2), exposes internal data to the external Cloud-native services using a RESTful API over the HTTPS protocol. The decisions made in favor of SCA were the following:

(i) Flexible and lightweight programming model to implement Service-oriented applications using SCA and non-SCA components.

(ii) Separation between development and assembly enabling technology-agnostic composition and reuse of heterogeneous services at the SaaS level

(iii) Dynamic service binding suitable for linking third-party services on-demand.

(iv) Support for portability and interoperability since SCA provides different component implementation types (Java, C++, Ruby, Spring, etc.) and binding for standard communication mechanisms (e.g., JMS, JCA).

(v) Ability to speed up development and dynamically generate and test code using the commercial tool IBM Rational Application Developer that offers support for SCA runtime.

IFT also learned that some tools supporting SCA (like the Eclipse plugins for SCA) suffer of some flaws to run SCA compositions, so in that case IFT decided to use the IntelliJ IDEA Plugin that provides support for working with SCA compositions.

V. Migration Assessment and Guidelines

From the decisions and results of the IFT migration case study we distilled the following activities that can be performed during the *assessment phase* in similar migration cases. We do not advocate any order and iterations in the execution of these activities as proposing a new migration methodology or process is out of the scope of this paper.

Reasons for migration: It is important to understand the drivers behind a migration effort by capturing specific requirements and constraints (e.g., time, cost, and feasibility) to satisfy and the consequent challenges to face. The scenario can be more complicated than simply making more scalable, consistent, and elastic the existing application in a cloud setting. As in the IFT case, there may be also the need to extend the core functionalities of the application with new ones developed by exploiting Cloud-native software services (SaaS).

Analysis of the application's environment: This analysis includes creating a detailed inventory of the current programs, scripts, and external interfaces involved. It also includes hardware and software configuration information, operating system versions, database versions, features/functionalities in use, and similar information.

Analysis of the new application's environment: The company has to assess the IT resources it needs to create the new environment of the application on the Cloud and evaluate how to provision these resources from different Cloud providers. IT resources include hardware, storage, network bandwidth/performance, critical database features and functions, virtualization software, Cloud-native software services, etc. Hardware and software resources required for performing the migration task itself also need to be identified.

Design and analysis of the application's architecture: Determine the core architecture of the application and report the dependencies with the current context environment (existing libraries, application containers and platforms) is necessary to achieve the desired interoperability and estimate the cost of the solutions adopted. The same architecture design and analysis should be repeated for the target Cloud-based environment. Cloud modeling approaches and supporting tools like TOSCA could be used in this activity.

Migration tools: Conduct small-scale proof-of-concept projects to try different home-grown tools that help with migration (such as Microsoft Migration Accelerator, CloudMigrator from Cloud Technology Solutions, etc.). The goal of such activity is to assess their efficiency, accuracy, and migration options, such as emulation or wrapper technologies. Some of these tools are helpful also for estimating the migration effort (the most common information businesses request when considering a migration project) that depends on factors such as database and application size, components, and complexity factors.

In addition to these assessment activities, we suggest a set of guidelines that may serve during migration as they summarize some relevant points we identified in our migration case study.

Guideline 1#. Service interoperability and composition: The use of standard Web-based interaction protocols such as RESTful APIs guarantees the interoperability of services and facilitates the communication among third party and proprietary services. Moreover, the adoption of standard design languages such as ADLs like the OASIS SCA and TOSCA and associated programming techniques and runtime platforms improves the interoperability and portability of the application across cloud platforms.

Guideline 2#. Avoid the "vendor Lock-in" barrier: An extensive adoption of standards like TOSCA and SCA enable higher levels of service portability avoiding the vendor Lock-in barrier and ease the migration of reusable services.

Guideline 3#. Service modeling and cloud deployment artifacts: It is important to remark how useful cloud modeling notations are to depict the topology of the cloud solution and manage the deployment of the cloud services throughout the life cycle.

Guideline 4#. Invoke legacy software: Direct access to legacy applications is not always practical, secure, or flexible. A basic VM-based migration combined with a Web service wrapper around the legacy application allows hosting existing systems in the Cloud and safely exposing existing functionality to both internal and external customers using a variety of communication methods via configurable protocols.

Guideline 5#. Balance cost and maturity of standards and technology options: Cost is an important factor when adopting commercial and open source solutions. Although SCA and TOSCA seem suitable modeling alternatives at low cost, the promising model-driven features for multi-cloud deployment are still in the infancy, and this may introduce a high risk for the migration strategy.

Many more assessment guidelines can be added, but the main limitation is the lack of maturity of the existing modeling standards and the supporting SOA/Cloud platforms that are still in the infancy and may introduce a high risk for the migration strategy.

VI. Related Work

There has been a number of migration case studies reported in the literature [7]. Here we focus on those works that present a more practical approach to real migration case studies or methods that might be applied in an industry setting.

Migration white papers: As there are several cloud vendors and providers, users can choose among different cloud solutions to port their existing IT infrastructure. For instance, Amazon provides a migration guide for their AWS (Amazon Web Services) [15]. This white paper reports a step-by-step migration strategy for the adoption and use of AWS, scenarios discussing the motivation of migration (e.g., scalability, elasticity, lower TCO), and the benefits for companies. Such assessment involves a staged process ranging from the initial assessment to the optimization of the solutions adopted. Other cloud vendors like Cisco or cloud solutions providers like Cognizant offer similar strategies and specific frameworks from company's white papers for cloud migration. For instance, Cognizant's Cloud360 hyper-platform is an open architecture that manages enterprise cloud services delivered on-demand and acts as an overlay of public/private/hybrid IT components.

Migration experiences: At lower levels in the cloud stack (i.e. the IaaS), some experiences such as the one described in [20] narrates the migration to Amazon EC2 in the Oilfield sector. In this case, the bespoke solution was mainly driven by cost factors of the cloud infrastructure used for the three companies involved that need to run small and large software instances. The authors discuss benefits (e.g., improve satisfaction of work, offer new product/services) and drawbacks (e.g., departmental downsize, deterioration of customer service quality) of the experience from the interviews with the stakeholders and the impact for the company.

However, during any migration strategy it seems sometimes necessary to simulate and evaluate the deployment options migration. The approach described in [11] discusses an evaluation of competing cloud deployment options (CDO) types when existing applications must be migrated to a cloud environment. This simulation is in many cases critical to know the number of cloud resources to be allocated in terms of performance and cost of the solution chosen and hence, measure the intensity of the cloud resources used. In [3] the authors propose Cloudstep is another "step-by-step" decision process to support the migration of legacy systems toward the cloud. They suggest the use of templates adapted to each organization to identify critical points and factors that may influence the selection of a particular cloud solution. The strategy suggested in [4] highlights the need to move to a cloud solution without disrupting the business of existing customers, and the authors propose to face the modernization of existing software to cloud in three phases. A pre-migration phase analyzes the weak points of the existing solution and assesses on the technical and business feasibility before it is decided moving to a cloud approach. Nowadays, any cloud migration strategy is many times constrained by the "vendor Lock-in" factor, and the software tools provided by cloud vendors may tight users to proprietary services. This factor may hamper the interoperability of component-based applications across different clouds and motivates the need for having cloud agnostic applications. Such ideas are deeply discussed in [8] in the form as a framework for building applications from different cloud vendors and enabling developers to choose which components must be deployed in each different cloud. The approach decouples the architecture from services and libraries provided by different cloud vendors. Therefore, each cloud artifact could be migrated between cloud platforms and avoiding the redesign and redevelopment efforts of the application.

Surveys and systematic reviews reporting migration: In addition to these proposals, we can find other studies such as a systematic literature review on cloud migration described in [16] and conducted over 23 case studies that are used to identify research gaps. This study reveals the need to improve the maturity of cloud migration strategies and a lack of tools supporting the automation of migration procedures. The study compares SOA and Cloud migration perspectives and motivates the current work to provide a cloud migration reference model for planning, execution, and evaluation of cloud migration tasks. One of the outcomes of this study is the need to count with additional case studies in real settings.

Standards: Finally, regarding the adoption of cloud standards, many of them changing very fast, the portability and orchestration issues with TOSCA, including the Openstack cloud middleware [9] facilitates the deployment of cloud applications combining different standards and technologies. The Cloud Application Management for Platforms (CAMP) [18] is another standard to support interoperability issues across multiple cloud infrastructures. CAMP is a specification that can ease the management of cloud applications across platforms at the PaaS level. TOSCA and other cloud standards are becoming popular as recently reported in [7], where different working groups are working actively in different cloud areas for portability, interoperability and cloud service-level agreement. All these factors affect to the migration strategies of companies for selecting the right tools and cloud modeling languages.

Nevertheless, all the knowledge gathered by the migration case studies and methods to assist software engineers in the transition from legacy systems to the cloud still need to overcome the important quality concerns to make the existing solutions interoperable with cloud standards.

VII. CONCLUSIONS

Planning the migration of enterprise IT applications to the Cloud is not easy in many cases, as interoperability, use of third-party services, and the vendor Lock-in factor complicate the decisions to make for the selection of a particular technology. In the case of company IFT, it was key important to analyze the current environment and the solutions to adopt to achieve the necessary scalability at reasonable cost and effort. We have faced a number of challenges and requirements we used to evaluate available technologies and standards, but some effort was required to evaluate the appropriateness of some OSS solutions. The role of cloud standards was important to ensure the technology adopted goes in the right direction. However, the plethora of modeling alternatives may complicate the selection of a particular cloud modeling choice, and is cost one of the most influencing factors that can reduce or increase the time for making a right decision when selecting cloud tools. In addition, the tools adopted must be tested (specially in OSS solutions) to know if they suit for orchestration and modeling goals, so services can be wired and deployed quick. Portability across clouds and platforms is another important concern for the adoption of proprietary technologies, but in many cases is more a business or company choice the influencing factor to select a particular technology. In other case how fast services can be deployed drives the decisions for effort and cost savings. Our case study discusses the underpinning reasons for competing solutions and advices software engineers on migration strategies. In our analysis, the availability of several solutions for different cloud layers can complicate the selection of a particular modeling notation accordingly to the services we need to migrate and deploy. TOSCA and SCA are conceived for different purposes but the maturity of some existing modeling notations requires of a careful evaluation by companies transitioning to a cloud model.

Acknowledgment

This work was supported in part by the Italian Ministry of Research within the PRIN project "GenData 2020" and by the EU-FP7-ICT-610531 SeaClouds project.

References

[1] V. Andrikopoulos, T. Binz, F. Leymann, S. Strauch, How to adapt applications for the Cloud environment - Challenges and solutions in migrating applications to the Cloud. Computing vol. 95, no. 6. 6, 493-535, 2013.

[2] T. Binz, Gerd Breiter, F. Leymann, T. Spatzier: Portable Cloud Services Using TOSCA. IEEE Internet Computing 16(3): 80-85, 2012.

[3] F. Fittkau, S. Frey, W. Hasselbring, CDOSim: Simulating cloud deployment options for software migration support. MESOCA IEEE CS 37-46, 2012.

[4] J. Guillén, J. Miranda, J.M. Murillo, C. Canal, A service-oriented framework for developing cross cloud migratable software, Journal of Systems and Software vol. 86, no.9, 2294-2308, 2013.

[5] J. Grundy, G. Kaefer, J. Keung, A. Liu, Software Engineering for the Cloud, IEEE Software vol. 29, no. 2, 26-29, 2012.

[6] S.I. Hashmi, V. Clerc, M. Razavian, C. Manteli, D.A. Tamburri, P. Lago, E. di Nitto, I. Richardson, Using the Cloud to Facilitate Global Software Development Challenges, 6th International Conference on Global Software Engineering Workshops (ICGSE), IEEE, 70-77, 2011.

[7] R. K. A., Idu, J., Hage, J., S., Jansen, S. Legacy to SOA evolution: A systematic literature review. Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments: Challenges in Service Oriented Architecture and Cloud Computing Environments, 40, 2012.

[8] P. Jamsidi, A. Ahmad, C. Pahl, Cloud Migration Research: A Systematic Review, IEEE Transactions on Cloud Computing, vol. 1, no. 2, 142-157, 2013.

[9] G. Katsaros et al., "Cloud Application Portability with TOSCA, Chef, and Openstack," Proc. 2014 IEEE Int'l Conf. Cloud Eng. (IC2E 14), 295–302, 2014.

[10] R., Khadka, A., Saeidi, S., Jansen, S., J., Hage, A structured legacy to SOA migration process and its evaluation in practice. In Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the (pp. 2-11). IEEE CS, 2013.

[11] A. Khajeh-Hosseini, D. Greenwood, I. Sommerville, Cloud Migration: A Case Study of Migrating an Enterprise IT System to IaaS, Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing (Cloud'10), ACM DL 450-457, 2010.

[12] R. Krebs, C. Momm, S. Kounev, Architectural Concerns in Multitenant SaaS Applications, Proceedings of the 2nd International Conference on Cloud Computing and Services Science (CLOSER 2012), 426-431, 2012.

[13] P. Lipton, Escaping Vendor Lock-in with Tosca, an Emerging Cloud Standard for Portability, CA Technologies, 2013.

[14] M.P. Papazoglou, W.-J. van den Heuvel. Service-oriented design and development methodology. Int. J. Web Eng. Technol., 2(4):412–442, 2006. P. Lipton, Escaping Vendor Lock-in with Tosca, an Emerging Cloud Standard for Portability, CA Technologies, 2013.

[15] S. Qanbari, F. Li, S. Dustdar, Toward Portable Cloud Manufacturing Serices. IEEE Internet Computing, 18(6), 77-80, 2014.

[16] M. Razavian, P. Lago, A systematic literature review on SOA migration. Journal of Software: Evolution and Process 27(5): 337-372, 2015.

[17] G. Reines, Cloud Computing and SOA. Service-oriented Architecture Series. The MITRE Corporation. Available at: http://architecture6.com/SERVICE-ORIENTED-ARCHITECTURE-(SOA)-SERIES---Cloud-Computing-and-SOA-download-w369.pdf, 2009.

[18] A. Sill, Cloud Standards News and Updates. IEEE Cloud Computing 2(1): 72-75, 2015.

[19] D.A. Tamburri, P. Lago, Satisfying Cloud Computing Requirements with Agile Service Networks, IEEE World Congress on Services, 501-506, IEEE, 2011.

[20] J. Varia, Migrating your Existing Applications to the AWS Cloud: A Phase-driven Approach to Cloud Migration, Amazon, 2010.

[21] Y. Wei, M.B. Blake, Service-Oriented Computing and Cloud Computing: Challenges and Opportunities, IEEE Internet Computing vol. 14, no. 6, 72-75, 2010.