A Systematic Mapping Study on the Multi-tenant Architecture of SaaS Systems

Institute of Mathematical and Computer Sciences, University of São Paulo (ICMC-USP)
São Carlos-SP, Brazil
victor.santiago@usp.br, helderfl, ricardoramos, pssouza, srocio{@icmc.usp.br}

Abstract—Background: SaaS (Software as a Service) is a services delivery model in Cloud Computing whose applications are remotely hosted by the service provider and available to customers on demand over the Internet. Multi-tenant Architecture (MTA) is an organizational pattern for SaaS that enables a single instance of an application to be hosted on the same hardware and accessed by multiple customers, so-called tenants, with the aim of lowering costs. Tenants are able to configure the system according to their particular needs. Objective: This research aims at the obtaining an overview of the challenges and research opportunities in MTA context for SaaS through a Systematic Mapping Study. Results: Eighty nine primary studies were selected for discussions on advances and opportunities for further investigations. The results showed the relevancy of MTA and pointed out the main research trends for next years in this topic.

Cloud Computing; software as a service; multi-tenant architecture; systematic mapping study.

I. INTRODUCTION

Cloud Computing has emerged from the contribution of techniques from parallel computing, distributed computing and platform virtualization technologies [1]. It provides dynamic resource allocation and has become one of the main research fields in Software Engineering. Furthermore, this technology enables cost reduction, optimization and opportunity for the creation of new business models [2]. A set of resources can be efficiently accessed on demand from anywhere and managed with a minimum possible interaction [3]. Cloud can be understood as a repository of virtualized resources (hardware, development platforms/services) easily accessible [4]. These resources can be dynamically reconfigured to be adjusted to diversified loads, which optimizes their usage. This wide range of resources has directly contributed to the emerging of different services delivery models, as SaaS (Software as a Service), which is a software deployment model of applications remotely hosted by a service provider and available to customers on demand. It offers benefits, as improved operational efficiency and reduced costs. As an instance, Salesforce.com\(^1\) provides an SaaS for Customer Relationship Management. Salesforce uses a subscription revenue model and charges clients per user on a monthly basis.

The cloud computing environment is different from a traditional environment in terms of hosted deployment, configuration, execution and management of applications. The main difference is related to type of users, security and sharing of resources such as databases, virtual machines or network connections among customers [5]. The sharing of resources among customers through logical separation is one of the main characteristics of multi-tenant architecture (MTA) for SaaS systems.

Multi-tenancy can be referred to an organizational pattern in which a single instance of an application is hosted on the service provider, and multiple companies, so-called tenants, access the same instance [6]. MTA enables a high degree of customization of software according to the requirements of many tenants and resources required for its execution are shared and provided on demand. For the end users, the application is executed in a dedicated environment, i.e., a fault of software in use by another tenant should not affect them. Furthermore, they are able to exclusively configure the system to their specific needs. MTA provides benefits, such as (i) optimization of the use of hardware resources, (ii) costs reduction by the maintenance of applications and (iii) new opportunities for data aggregation. However, challenges as those related to security, data sharing, database, customization, validation and testing, performance and migration from conventional web applications [7][8] must be overcome.

A Systematic Mapping Study (SMS) is a proper method to map a certain topic when few evidence exists or the research topic is wide or scattered. Therefore, we have carried out an SMS on the multi-tenancy of SaaS systems following the guidelines proposed by Kitchenham [9]. Eighty nine primary studies were selected to answer two research questions from the academic perspective. The analysis of the results focuses on presenting the frequencies of publications for different research categories. As main contribution, we have provided a definition of main challenges to guide future research on the multi-tenant architecture domain.

The paper is organized as follows: Section 2 discusses the phases of the SMS; Section 3 addresses the threats to validity and Section 4 reports the conclusions and future work.

II. THE SYSTEMATIC MAPPING STUDY

The SMS was conducted considering three main phases: (i) planning, (ii) conducting and (iii) reporting. The next sections address these phases and the obtained results.

\(^1\)www.salesforce.com/sales-cloud/overview/

DOI reference number: 10.18293/SEKE2016-068
A. Planning

In this phase, the review protocol containing (i) research questions, (ii) search strategy, (iii) inclusion and exclusion criteria and (iv) data extraction process and methodology for the synthesis of the data was defined.

The main goal was the achievement of a background of difficulties related to MTA, alternatives proposed in the literature and research opportunities. Therefore, two research questions (RQ) were defined:

RQ1: What research topics related to MTA can be found on the current literature?

RQ2: What are the main research challenges and opportunities related to the development, testing and evolving of multi-tenant SaaS applications?

A search string and the electronic databases were also defined. The search string was elaborated and refined according to an initial set of key papers selected and based on citations of these papers. During the string validation these papers must always be retrieved from electronic databases (Table I). Although subjective, this control enabled the string calibration and identification of possibly relevant studies.

TABLE I. List of key papers used to calibrate the search string

<table>
<thead>
<tr>
<th>Authors</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Seungsook et al.</td>
<td>[10]</td>
</tr>
<tr>
<td>Sengupta and Roychoudhury</td>
<td>[7]</td>
</tr>
<tr>
<td>Tsai et al.</td>
<td>[8]</td>
</tr>
<tr>
<td>Ru et al.</td>
<td>[11]</td>
</tr>
</tbody>
</table>

We defined the search string considering the following keywords: cloud, SaaS and multi-tenancy, their frequent variations and boolean operations. Figure 1 shows the search string elaborated. The following databases were considered: ACM, IEEE, Scopus and Wiley Online. Such databases cover the main conferences and journals on cloud computing.

Relevant primary studies were selected based on the following inclusion (IC) and exclusion criteria (EC). Not all inclusion criteria should be satisfied for each primary study; IC\(_a\) is the only mandatory criterion for the inclusion of papers.

IC\(_a\): The primary study presents at least one challenge or research opportunity in the context of MTA;

IC\(_b\): The primary study presents at least one tool, framework, process or APIs for MTA context;

IC\(_c\): The primary study addresses at least one difficulty involving the MTA in usage and migration terms;

IC\(_d\): The primary study presents at least one property, classification or evaluation of a solution considering the MTA;

EC\(_a\): The study presents a challenge or a research opportunity in the MTA context. However, it is a short paper;

EC\(_b\): The study is a Systematic Literature Review;

EC\(_c\): The whole study is unavailable.

We have used a data extraction form to answer the review questions, presented in Table II. We have included some categories (Item 5) in order to classify the main domain of each primary study, for instance, “Customization” and “Database” are defined categories.

TABLE II. Contents of data extraction form

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metadata</td>
<td>ID, reviewer and date.</td>
</tr>
<tr>
<td>Content</td>
<td>Title, year, source (i.e. conference or journal) and search database.</td>
</tr>
<tr>
<td>Data extracted</td>
<td>1) Challenge/opportunities; 2) Tools, frameworks and APIs; 3) Difficulties in the MTA usage and adoption/migration; 4) Specifications, classification and evaluation of solutions for MTA and 5) Category of paper.</td>
</tr>
</tbody>
</table>

During the data extraction process, the data from primary studies were collected by three reviewers, PhD candidates in Computer Science. They were extracted by one researcher and checked by another. This SMS was performed between May and August, 2015 and the data have been documented and are available\(^2\).

B. Conducting

In this phase, the primary studies were identified in the aforementioned search databases. Scopus returned a larger set of studies (638). IEEE, ACM, Wiley returned 168, 594 and 25, respectively. Figure 2 shows the distribution of papers retrieved in each search database and after applying the inclusion and exclusion criteria in the reading process. From this initial set, 135 duplicated studies were identified and removed. In the selection phase, based on the partial reading (titles and abstracts), a set of 149 papers was selected according to the inclusion and exclusion criteria; after the full reading, only 89 papers were selected. We wanted to be conservative as possible, therefore the search string has become generic to retrieve many studies from electronic databases, even if it would give us more effort in the selection process. Many papers were introduced as primary studies, but only few of them had more contributions or larger impacts.

In order to validate the inclusion and exclusion criteria application, each primary study was scored by reviewers in

\(^2\)https://goo.gl/0K68jp
both partial (title and abstract) and full readings. A score “0” means rejected and “1” accepted. In cases of doubt, the reviewer scored the paper with “0.5” and the other reviewers were asked about its relevance, so that a consensus could be reached through the adoption of score “0” or “1”.

The scoring process was conducted in a sequential and independent way, i.e., one reviewer read and scored each paper without interference from others. In partial reading all papers were scored and posteriorly, in full reading they were classified again by reviewers until reaching a set of relevant studies to answer the research questions.

C. Reporting

This section discusses an overview of MTA based on selected primary studies.

1) RQ1: What research topics related to MTA can be found on the current literature?

In order to clarify the focus of the selected studies in quantitative terms, we have defined some categories according to the paper domain, as aforementioned. Figure 3 shows a mapping containing number of primary studies distributed according to publication year and category, which one paper can be classified and more than one category.

Security, testing activity and experiments may be considered important issues to quality assurance and, therefore, these issues are into quality assurance category.

Figure 4 presents the disposal of the papers selected from workshops, journals and conferences. Conferences have a dominant position with 63 papers (70.8%), followed by journals with 18 papers (20.2%) and workshops, 8 papers (9%). It can indicate that workshops still have to be formed and researchers submit their results to conferences and journals with a larger scope.

The next sections discuss the main idea of the selected studies organized by the categories shown in Figure 3.

a) Development: Architectures, frameworks, requirements and variability management, and migration from web conventional applications to multi-tenant SaaS applications are mentioned in the current multi-tenancy literature as important issues to address in future research.

b) Performance: MTA has introduced new challenges related to load balancing and resource allocation, including the requests on the tenant level, service level agreement, performance objectives and quality of services. Sun et al. [18]
TABLE III. Main issues about development of multi-tenant SaaS applications

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Description</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementing</td>
<td>Context-oriented programming</td>
<td>[12]</td>
</tr>
<tr>
<td></td>
<td>Guide to implementing new tenants without impacting those already deployed</td>
<td>[7]</td>
</tr>
<tr>
<td>Frameworks</td>
<td>Process to support the portability, customization, security and scalability</td>
<td>[14]</td>
</tr>
<tr>
<td></td>
<td>Framework to address availability, extensibility and scalability in a multi-tenant application</td>
<td>[17]</td>
</tr>
<tr>
<td>Migrating</td>
<td>Process to support the migrating of conventional web applications towards multi-tenant SaaS application with relatively less effort</td>
<td>[16]</td>
</tr>
</tbody>
</table>

proposed a suitable load balancing policy for a multi-tenant environment to provide satisfactory quality of services. On a database level, Moon et al. [19] presented a load balancer for multi-tenant databases to increase the performance and sharing of resources among tenants. Patikirikorala et al. [20] developed an approach that uses a nonlinear replenished control to keep the performance in distinct usage levels for different tenants, depending on their priorities. It enables the detection of overload, therefore the control of tenants operations can be dynamically changed.

Krebs et al. [21] extended a web benchmark called TPC-W to include multi-tenancy and compared the cloud usage under two perspectives: (i) multi-tenancy and (ii) virtualization. Multi-tenancy shown higher efficiency than virtualization considering the throughput, number of tenants and when memory was a bottleneck.

c) Customization: A considerable number of papers have addressed applications customization. A multi-tenant SaaS application which address a large set of tenants should make possible a large number of customizations [22]. The customization of a complex application is an error-prone task, it requires high manual efforts and the users may not know the best choice in terms of customization. Thus, the authors performed a study of the possibilities available for the customization of an SaaS application, and a semi-automatic customization process was created to reduce efforts.

Ramachandran et al. [23] observe that the customization may result in high cost of readjustments. For multi-tenant systems, it involves the configuration of specific instances and management of allocated resources for the tenants. For Walraven et al. [24] customization involving variations in the core of the application is expensive for SaaS providers, introducing an additional complexity.

d) Quality assurance: Quality assurance is a promising research topic in MTA [25] that includes testing strategies, metrics and quality criteria, and alternatives related to security. For Tsai et al. [8], one of the main challenges in the testing activity of multi-tenant SaaS application is to deal with the large set of composition possibilities and interactions among components. The authors provided a combinatorial testing approach to generate dynamic test sequences and achieve a high structural coverage. The main idea was to identify the compositions likely to result in failures by an algorithm. When a new component is composed in a certain application instance becomes available, the algorithm reveals defects in the interactions among components.

The complexity of the cloud computing model and lack of standardization become the security a critical issue for cloud providers and customers. According to Wood et al. [26], multi-tenancy directly impacts on the applications development and the way they are provided. Almorsy et al. [27] created a framework to improve collaboration between service providers and consumers and manage the security of cloud platform and its hosted services.

Table IV presents the main issues investigated in relation to quality assurance in the MTA context. Due to space limitation we are showing only some papers for each issue.

TABLE IV. Main issues about quality assurance of multi-tenant SaaS applications

<table>
<thead>
<tr>
<th>Description</th>
<th>Ref.</th>
</tr>
</thead>
<tbody>
<tr>
<td>System Testing</td>
<td>Combinatorial testing: dynamic test sequences were used to achieve high architectural coverage</td>
</tr>
<tr>
<td>Regression Testing</td>
<td>Continuous testing with partitioning of data from tenants and generation of test case based on meta-data</td>
</tr>
<tr>
<td>Security</td>
<td>Framework for security management</td>
</tr>
<tr>
<td></td>
<td>SecPlac: resource allocation model to support the security in the sharing of infrastructure among tenants</td>
</tr>
<tr>
<td></td>
<td>TOSSMA (Tenant Oriented SaaS Security Management Architecture): an architecture to isolate resources for tenants through the injection of authorization controls</td>
</tr>
<tr>
<td></td>
<td>Data combination privacy</td>
</tr>
<tr>
<td>QoS</td>
<td>MSSOptimiser (Multi-tenant SaaS Optimizer): an approach to select services addressing quality requirements</td>
</tr>
</tbody>
</table>

e) Database: Nineteen papers in SMS have cited database-related issues as a promising research direction. Saraswathi and Bhuvaneswari [33] presented two alternatives for multi-tenant data architecture: i) one related to authentication and authorization and ii) a non-intrusive approach for large-scale applications. The authors described a process to apply them and guide engineers in the development of databases.

Maenhaut et al. [34] developed an approach for data management in a hierarchical way and taking into account some performance metrics. The main question addressed concerned the distribution of users and data into multiple instances of database. Yaish et al. [35] discussed an access control model based on a database schema. They also proposed an access control algorithm that enables users to access the data granted based on users groups or assigned roles.

2) RQ2: What are the main research challenges and opportunities related to the development, testing and evolving of multi-tenant SaaS applications?
Although most studies have addressed the development of multi-tenant SaaS applications, standards are scattered, and do not often follow a methodical approach. Furthermore, the solutions are proprietary and rarely interoperable [25]. Traditional software testing cannot be applied to test applications in a Cloud environment due to it is designed for on-premise single-tenant applications [36].

Several issues should be considered during the testing of multi-tenant SaaS application: (i) resources are shared among tenants and their end-users, (ii) each variant application addresses a specific requirements set for a tenant, it is executed as if it was in a dedicated environment and can be composed of several components and (iii) a variant application is delivered to the customers through a run-time engine from cloud provider that weaves the tenant customization data and specific metadata to kernel code. Thus, each application provides different screens and logic.

According to Alkhatib et al. [25], the community still does not have effective quality metrics for the SaaS and new testing strategies are required to meet the challenges imposed by the cloud computing model. Software integration testing issues, validation methods and quality assurance standards addressing the interaction interfaces must be established. Since high system availability is essential to SaaS, the re-testing techniques considering the multi-tenancy feature are mandatory whenever software is changed for improvements or bug-fixing.

Recent studies on tests in cloud computing have addressed the verification of nonfunctional requirements as performance and security. Considering the selected studies, we have identified that research fields as (i) adjustments of conventional test criteria, (ii) test strategies for customization components, (iii) alternatives to verify the composition interfaces and impact new components, and (iv) approaches to regression test require more cooperation between industry and academia.

Regarding the evolution of multi-tenant SaaS applications, the community still has not provided well-defined approaches. The evolving activity of distributed systems may indicate guidelines for dealing with the isolated execution of instances of these applications.

3) Research agenda: In order to guide future research in the area of multi-tenancy, this section presents the major trends identified in this study, as follows:

Development process. Most of selected studies about development process proposed a solution or a process without a practical evaluation. Methodical approaches to guide the development of cloud-based applications require more research effort, especially taking into account the multi-tenancy.

Quality metrics. Metrics are used to guide managers during the software quality evaluation. Despite many studies mentioning their importance, there is still a lack of quality metrics for this context.

Testing activity. Few organizations and academy provide security testing, recovery testing, fault-tolerance testing or some alternative to cover the complexity of multi-tenancy for SaaS. In addition, there is a lack of standards to driven the development of interoperable test tools. From our point of view,
In case of disagreement among the reviewers, a specialist was called to guarantee the correct decision.

IV. CONCLUDING REMARKS

Software as a service is a way of delivering applications over the Internet as a service for multiple customers. From the point of view of service providers, the computing resources to be offered must be broadly shared. For the users, it is important to customize the application according to their specific requirements. In this scenario, an architectural pattern called multi-tenancy is gaining more ground in the application space on cloud.

In order to provide a mapping of research topics on the multi-tenancy of SaaS systems and identifying new research opportunities, we have conducted an SMS in which 89 primary studies were selected for discussions. We have defined two research questions that reflect the scope of the study and five categories to map the contributions and challenges.

This mapping study also points out the need for experimental studies evaluating the proposed approaches and a systematic test strategy extending different techniques to increase the quality of these applications. In our future research, we intend to compare the evidence identified in this work with evidence from industrial cloud projects in order to define new hypotheses, which will guide the definition of approaches for testing of multi-tenant SaaS applications.

REFERENCES