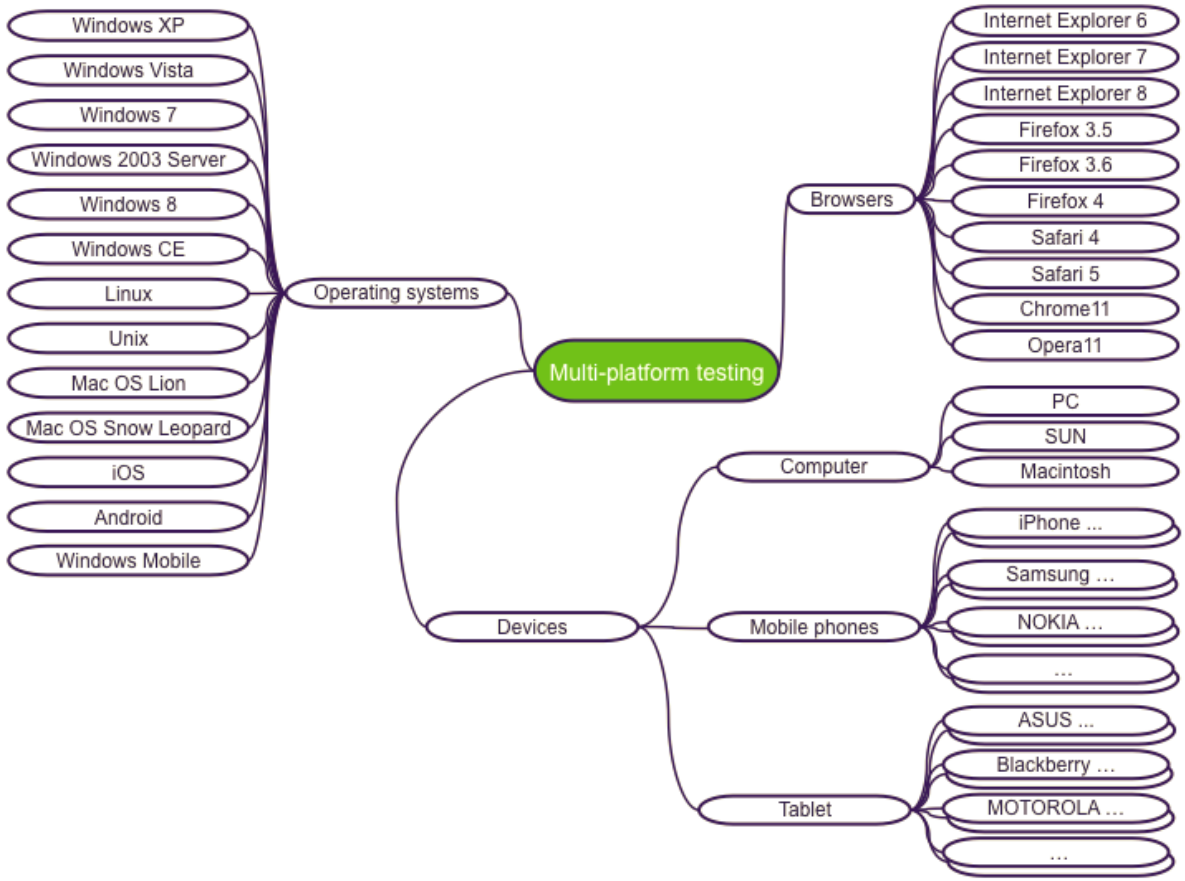
Impact of the Vendor Lock-in Problem on Testing as a Service (TaaS)

Ricardo Ramos de Oliveira
Rafael Messias Martins
Adenilso da Silva Simao

5th IEEE International Conference on Cloud Engineering
IC2E 2017

Summary

1 - Introduction



Test of Multiple Platforms (**Client**)

Context

- In this context, Testing as a Service (TaaS) is defined as a test tool offered as a service in the cloud for the **verification** and **validation** of the functionalities of a web system through the Internet;
- However, the lock-in problem imprisons the user in the platform of a specific vendor or test service due to the difficult migration from one TaaS provider to another;
- The vendor lock-in problem limits the use of those new technologies and prevents a widespread adoption of TaaS.

Objectives:

1. Identification of the impact of the **lock-in** problem on the entire **cloud testing process**;
2. Proposal of a **solution** to the lock-in problem in both **writing** and **execution** of tests through a test architecture that abstracts TaaS providers;

2 - Background and Related Work

- Testing as a Service (TaaS)
- Vendor Lock-in Problem
- Portability and Interoperability

Testing as a Service (TaaS)

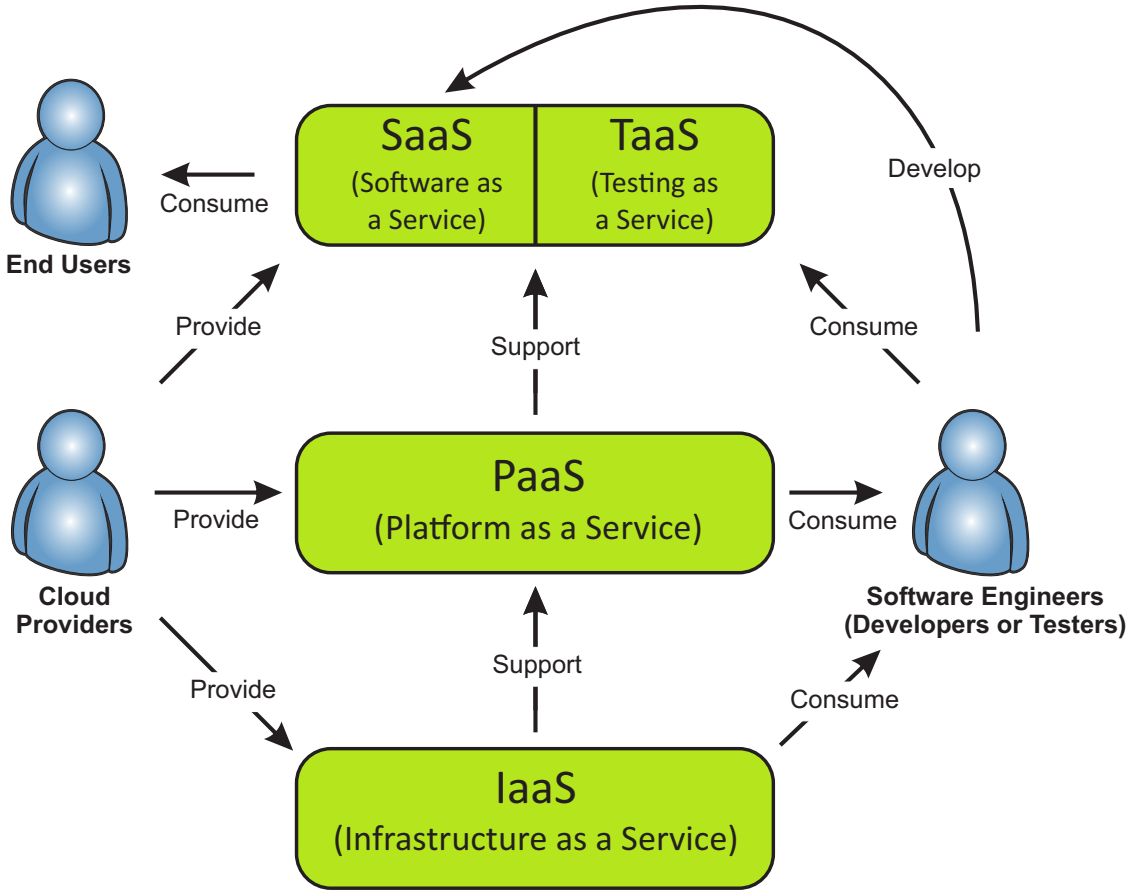
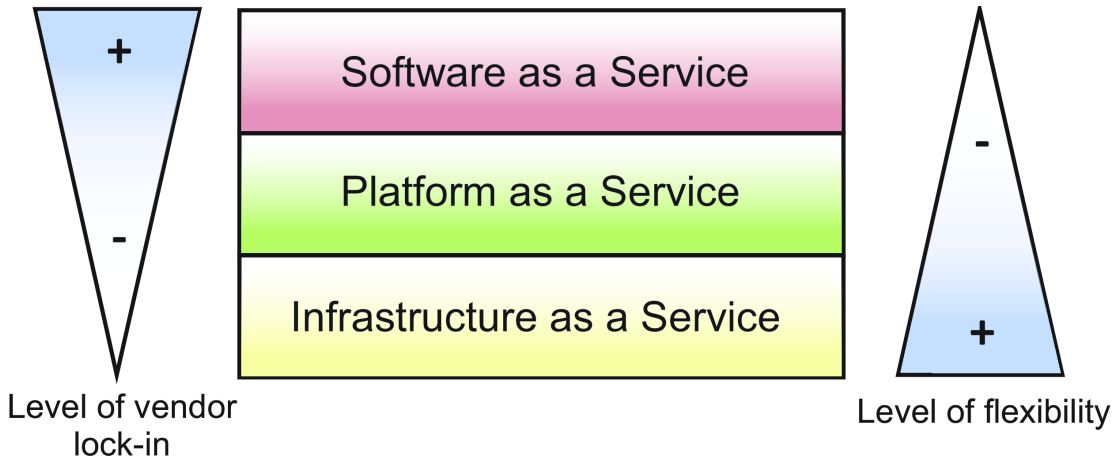


Figure: Example of interactions between actors and service models

Lock-in Problem

- One of the main problems associated with **software development and testing** in a cloud environment is known as **vendor lock-in**;
- It results from the lack of **portability** and **interoperability** among providers and causes the imprisonment of users on a particular cloud vendor due to proprietary technologies.



Portability and Interoperability

- **Portability** in a TaaS context refers to the ability of software engineers to write a test once and run it in multiple TaaS platforms with no changes in it;
- **Interoperability** in a TaaS context refers to a standard protocol (or an abstraction layer) that enables heterogeneous TaaS providers to **collaborate** with each other in a transparent way to their clients. This is possible only if their REST APIs follow a common specification.

Related Work

- Petcu (Petcu, 2011) [1] listed the main approaches, namely **Open APIs**, **Open protocols**, **Norms or standards**, **Abstraction layers**, **Semantic repositories** and **Domain Specific Languages (DSL)** for the solution of the **lock-in problem** in cloud computing;
- In general, all studies on lock-in have focused on the **cloud environment**;
- In contrast, the approach adopted here focuses specifically on the application of design patterns in the **context of the TaaS service model**.

[1] D. Petcu. **Portability and Interoperability between Clouds: Challenges and Case Study**. In W. Abramowicz, I. Llorente, M. Surridge, A. Zisman, and J. Vayssi'ere, editors, Towards a Service-Based Internet, volume 6994 of Lecture Notes in Computer Science, pages 62–74. Springer Berlin Heidelberg, 2011.

3 - The Lock-in Problem in TaaS

- The lock-in problem in the context of TaaS is caused by **specific Selenium capabilities** offered by different TaaS providers;
- The **capabilities** are configuration parameters formed by a **key and value pair** directly related to the execution of tests;
- Our main focus is on the steps of the **writing process of the capabilities** and **execution of automated User Acceptance Testing UAT**, categorized here as a lower-degree problem.

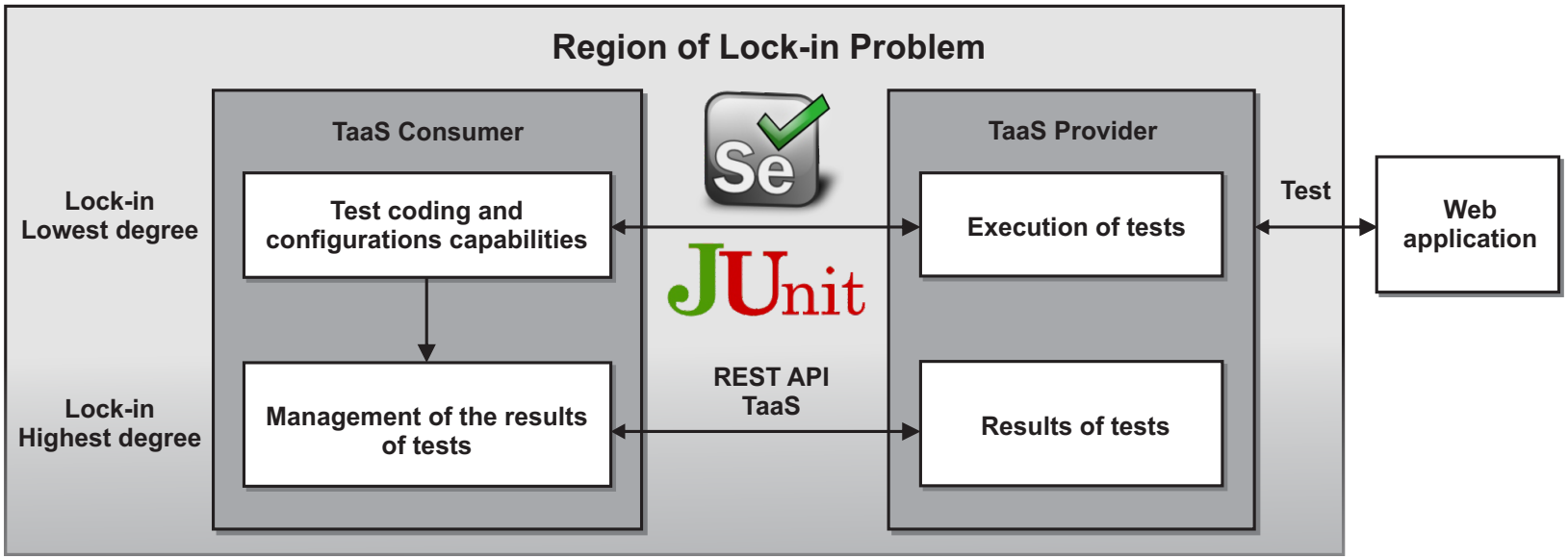
Cloud Testing Process

Steps in the cloud testing process affected by the vendor lock-in problem:

1. **Test Coding**
2. **Test Execution**
3. **Test Management**
4. **Test Configuration**

Cloud Testing Process

This research has identified the main stages of the software testing process in the context of TaaS that are affected by the lock-in problem.



4 - Development Guidelines and Test Architecture

- The central idea of our approach lies in the use of a combination of design patterns guided by a set of environment variables through a test architecture;
- We propose a set of guidelines that are independent of vendor technologies for minimizing the impact of vendor lock-in.

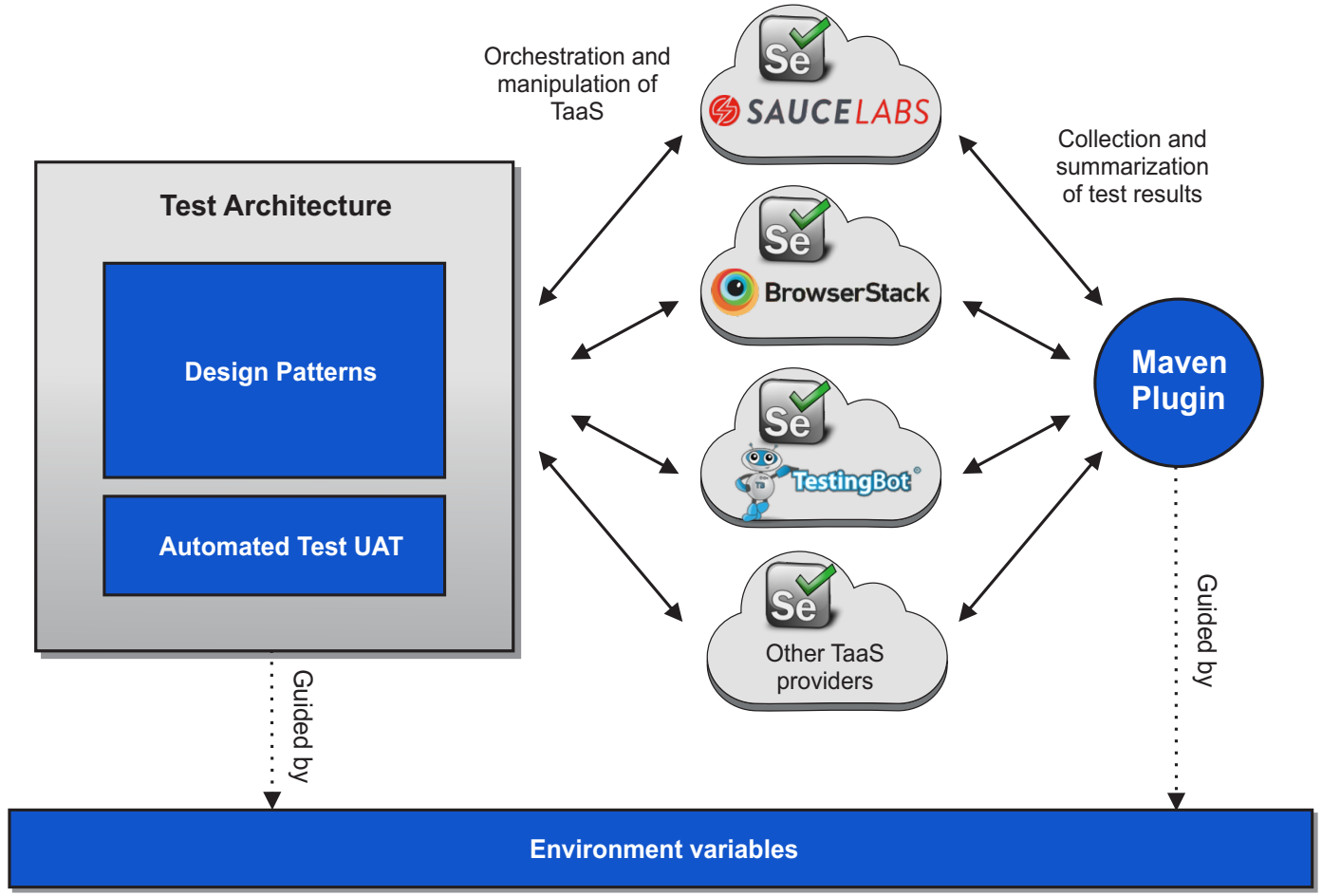
Capabilities of TaaS

- The way capabilities are written is **not standardized**, which hampers the adaptive maintenance of the **test code** for all the different combinations of capabilities;
- As each test set may have different capabilities, a company that uses **hundreds** or **thousands** of **tests** will incur exorbitant costs for test code maintenance if it changes provider.

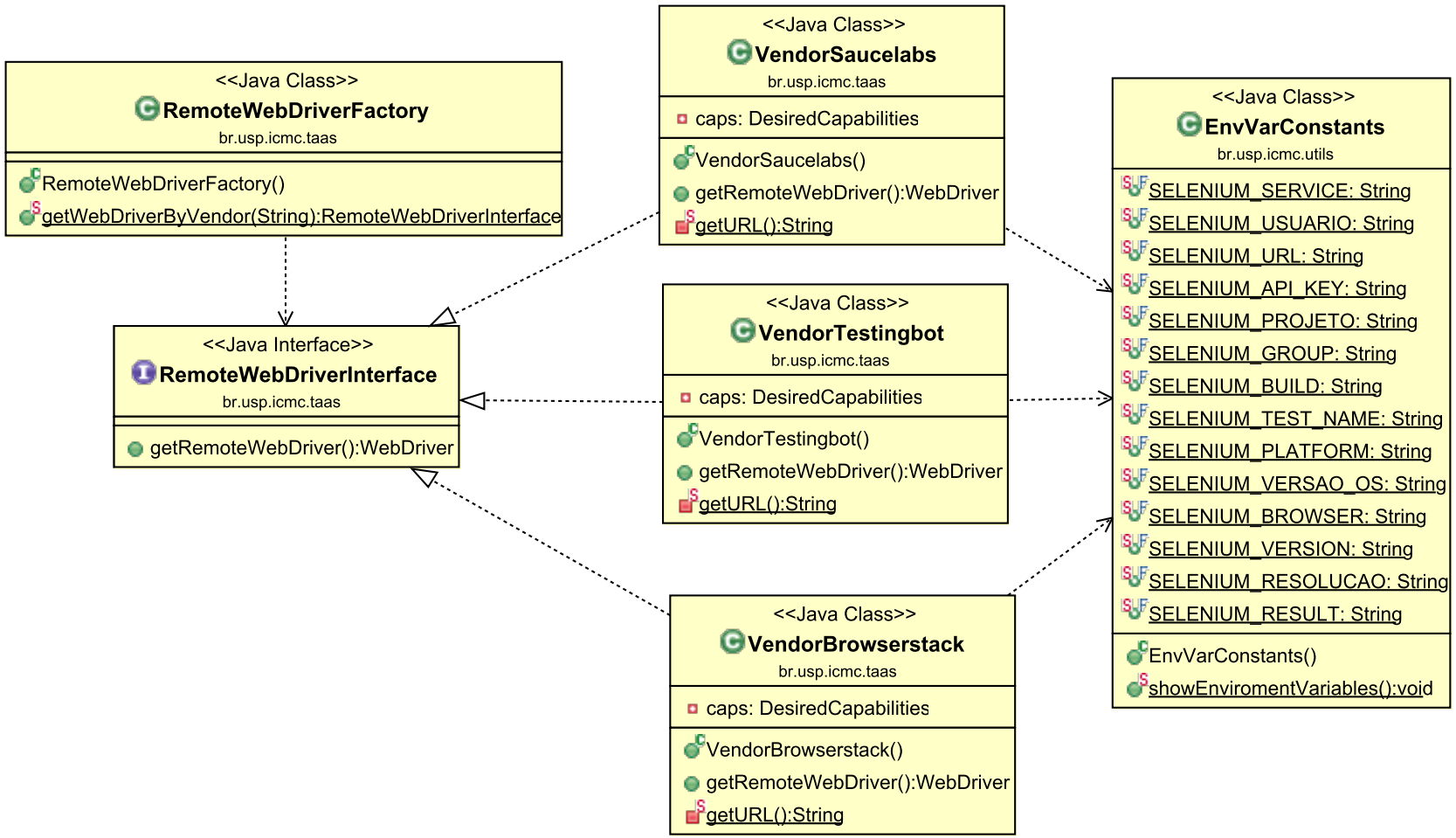
Contributions

- The coding and execution of tests become more **practical**, **efficient** and **secure**, as changes in the environment variables do **not impact** on the **remainder of the code**;
- Therefore, the source code of the application requires no change, which **reduces the possibility of new faults** as a **result** of the **system maintenance**;

Test Architecture



Class diagrams of the Abstract Factory and Strategy design patterns



5 - Conclusions

- This study has proposed a solution to the lock-in problem in the writing and execution of tests for mitigating the impact of vendor lock-in on TaaS;
- It has also identified the four main stages of the software testing process, namely **writing**, **execution**, **configuration** and **management**, which are affected by vendor lock-in in the TaaS context;
- The main contribution of the proposed approach regards the increased **portability** of the **tests** and **interoperability** of **TaaS** through a **common interface**.

Future Work

- In order to validate our approach, we have conducted **controlled experiments** with students to measure the **effort** and **time spent** on the migration among TaaS providers.
- We also aim at demonstrating its **applicability**, **effectiveness** and **relevance** in the real world;
- The solutions to the lock-in problem in the **management** and **configuration** stages of automated tests in a cloud environment have been investigated and will be the subject of future studies.

Thank you.
My contact: ricardoramos@icmc.usp.br