The Platform as a Service (PaaS) Cloud Model: Opportunity or Complexity for a Web Developer?

Serena Pastore
INAF – Astronomical
Observatory of Padova
vicolo Osservatorio 5, 35122,
Padova, ITALY

ABSTRACT
Technology revolution brought by cloud computing, is affecting the work of web developers that could benefit of installed frameworks and platforms where to deploy applications without management needs. Cloud computing in its different usage models according the kind of offered utilities (e.g., storage or computing facility as software, web framework as a platform and virtual machine as infrastructure), is becoming a successful distributed computing paradigm thanks to the services made available by several commercial organizations in many cases with a free starting plan. This paper wants to analyze the cloud offerings from the web developer's perspective that often need to configure and manage the environment where to design, develop and deploy their applications. The aim is to understand whether such services are effective for the development and deployment of general-purpose web applications (i.e., from website to mobile web apps) or introduce further complexity requiring additional skills to developers.

General Terms
Social and Professional Topics, Management of computing and information systems. Software and its engineering, distributed systems organizing principles

Keywords
Cloud computing, PaaS, web development, open stack software.

1. INTRODUCTION
Cloud computing [1] seems to promise benefits for web developers offering online frameworks and platforms in which develop, deploy and manage web applications. With the term web applications, we refer to every form of applications (e.g., generic application, portal, social framework, mobile application or apps, widgets, and web services) that is executed through the web usually within the engine of the web browsers. Technologies used for such development include standards relied to web architectures (HyperText Markup Language, HTML, HyperText Transport Protocol, HTTP and Uniform Resource Identificator, URI) [2], and web languages used to structure, rendering, visualizing and interacting with the content (e.g., eXtensible markup language, XML-based languages [3], Cascade Style Sheets, CSS [3], ECMAscript [3] and its dialects). Implementation of these standards and languages are available as frameworks composed by a web server, a programming language environment (e.g., Java, PHP, Ruby, and Python) and a set of related software needed to execute the web applications. The design and delivery of a web application involve the choice of the working environment and therefore its installation and configuration within a reference operating system. This environment includes a lot of software that should be maintained and updated (e.g., operating systems, server software such as a web server, application server and related middleware, libraries, tools and framework). Perform these tasks involve an increase of the web application development time, and then of the general costs. Cloud computing offers software utilities through three main usage models (Software as a Service, SaaS, Platform as Service, PaaS and Infrastructure as a Service, IaaS) that eliminate end-user operations associated with installing hardware and software facilities.

Cloud-based services are resources that could be accessed through Internet usually in a web-like form by means the browser. The usage model choice depends on the needs and the service required, for example, a software, a framework or a virtual machine providing a computer system. The cloud model approach distinguished in private, public or hybrid, is a decision to be made considering five key features: the budget, the speed, demand and users to support, resources, and privacy related to data sensitivity. Fig 1 shows the different types of cloud model approach according the different kind of usage.

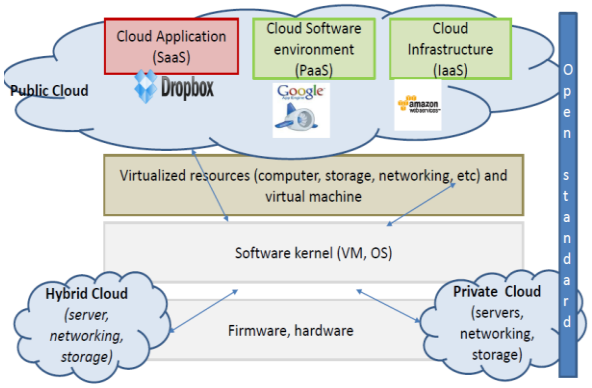

Fig 1 Cloud usage and approach models.

SaaS or IaaS models are the major cloud implementations providing a service available to the user respectively a software (e.g., Dropbox [4] that offer an online file storage) and an infrastructure (e.g., Amazon Web Services, AWS [5] that offers entire virtual machines). Focusing on web developers that would like to minimize the tasks related to setting and configuring the environment, the PaaS usage design could be an opportunity to free developers from system/server administration operations. In this context the

terms "SysOps" or "DevOps" [6] refer to the tasks required respectively by computer system administrators/managers and applications developers in setting and configuring, managing and monitoring the underlying software environment and frameworks needed by the applications.

Cloud moves in some way towards a "NoOps" context meaning that all management tasks related to the environment and the lifecycle management are made by cloud providers giving the possibility to developer to concentrate totally to the application. Developers can code and let the cloud service deploy, manage and scale their code by means of automated systems. A web developer will like to have a full working development platform that do not require configuration duties and thus develop applications that, however, could be portable to other platforms. There is the need of a solution not lock into some platform. The paper is so a study on the available PaaS offerings to assess whether the developer work will improve by using this tools or if such services introduce a further level of complexity requiring specific skills that could increase costs and time required to the development. It wants to give an overview of PaaS offerings focusing on the aspects of requirements. The goal is to verify if such solution are applicable in the context of public research institute to deploy web applications in a general-purpose perspective, ranging from website to mobile app.

2. NEED OF THE PAAS OFFERING OVERVIEW

Web developers make different choices as regards the platform to use in their development. We should consider that the application is a collection of source code written in a programming language, and a dependency description that defines which additional software are needed in order to build and run the application. Even if each developer has a "favorite" programming language or development platform, the choice of the environment depends on the kind of application, its purpose and the business context. Such decision could need a specific runtime environment or a layered framework that manages the application lifetime's phases. The PaaS usage model is the offering of a development platform where to design, implement and deploy web applications. In this context platforms are usually composed by the different layers (Fig. 2) needed for web applications: a backend system (i.e., a database management system – DBMS) [2], a web server and web application server that could be different according the programming language server-side chosen for the development (e.g., PHP, Java, Python, Ruby) [2] and a front-end platform able to deal with web standards.

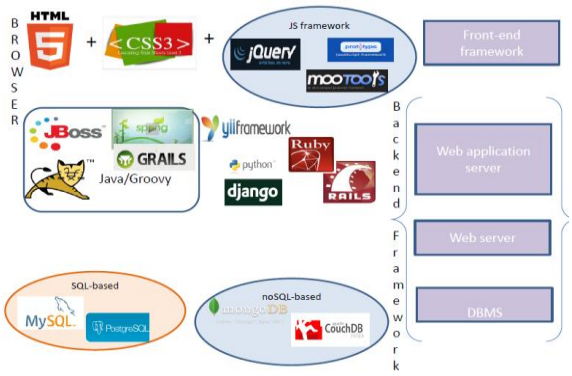

Fig 2 Architecture of a web application hosting environment.

The basis is a web server that handles with HTTP request/response, then, depending on the programming languages used to design server-side applications needed for interaction, there is a plethora of solutions and platforms available to help in the development.

Among different programming languages specifically used in a web context, as the Figure 3 shows, we could cite Java, PHP, Python, Ruby or JavaScript. The same figure shows for each programming language the main features and some of the referring frameworks supporting web development. For example, next to Java programming language, Groovy [7] is an agile and dynamic language for the Java Virtual machine (JVM) build upon the strengths of Java, but with additional features inspired by other languages used in a web environment like Python, Ruby and Smalltalk. Groovy is an object-oriented language used in the Java platform, with native support to markup languages like XML, which it is easier to learn.

Programming language	Features	Web application framework (open source)	Website-based framework
Java (Groovy)	General purpuose object-oriented language (agile dynamic language for the Java Virtual machine (JVM)	Jboss projects (www.jboss.org), Openlaszlo (www.openlaszlo.org), google web toolkit (developers.google.com), apache tapestry/structs (tapestry/structs.apache.org) + Grails (grails.org)	
PHP	General purpuose	Zend framework (framework.zend.com), Yii (www.yiiframework.com)	Website-based (Joomla!Platform)
Python	General-purpuose	Diango (www.djangoproject.com)	Zope www.zope.com
ColdFusion Markup Language (CFML)	Scripting language for web development (runs on the JVM, .NET and Google app engine)	CMFL engines (CooldSpring/Coldbox –platform www.coldbox.org; Fusebox, www.fusebox.org	
Ruby	General-purpuose, object-oriented (agile web development)	Ruby on rails (rubyonrails.org), Sinatra,	
Perl	General purpuose, interpreted	Catalyst (www.catalystframework.org)	
Scala	Multi-paradigm language (Java-like)	Play! (www.playframework.com)	
SmallTalk	Object-oriented reflective	AIDA/web (www.aidaweb.si), seaside (seaside.st)	
ECMAscript	Standard with different dialects implementation (Javascript, JScript)	Flex (Apache/Adobe) software development kit (SDK) for cross-platform rich internet apps (RIA) based on Adobe Flash (flex.apache.org)	
ASP .NET	.net windows development framework built on the Common Language Runtime (CLR)	ASP .NET MVC Framework (www.asp.net/mvc), OpenRasta (openrasta.org)	DotNetNuke www.dotnetnuke.com
C++	General purpuose language (object-oriented)	Tntnet (www.tntnet.org), Wt (www.webtoolkit.eu/wt) widget-oriented	CppCMS (cppcms.com)

Fig 3 A comparison between different environments

Groovy has a reference framework known as Grails [7] that provides a complete ready-to-use development environment usable also by the Java platform. It is related to Java technologies such as the Spring framework [8] that is the most popular application development framework for enterprise Java that helps to create high performing, easily testable, reusable code without any lock-in. The case of Javascript language is peculiar. Born as client-side scripting language, with the development of server-side Javascript framework (e.g., node.js [9] a server-side software built on Chrome's Javascript runtime) and of the web server able to process such code, the gap between client and server programming is eliminated.

Next to the selection of a platforms related to a particular programming language, the selection of a Database Management System (DBMS) [10] currently imposes a decision on the model of data management: not only relational as in the relational database management system (RDBMS), but based, for example, on document or key-value stores such as in the so-called noSQL databases. Most of the currently web platforms are based on Linux, Apache, MyqSQL or Postgres and PHP or Python software (i.e., LAMP/LAPP platforms) and thus are SQL-based, but in recent times, non-relational models are taking over the management of a large amount of data. Therefore also NoSQL database solutions

such as MongoDB, CouchDB, Cassandra [10] should be taken into consideration. In any case, regardless the platform chosen, web developers have a large part of the work dedicated to the installation, configuration and further maintenance of the right environment. The required operations or ops are usually part of the tasks required by system managers or administrators and include the software packages installation, servers' configuration and database interactions. And these are tasks that require time and expertise from a web developer. A side effect of cloud adoption is a reduction of the time occupied in such duties (even if partially and possible only in some contexts), allowing web developers to concentrate on the real application development and all the issues related to the application (i.e., usability, mobility, interoperability, and so on). In this framework PaaS offerings fall in a DevOps or NoOps [6] view. These terms indicate a platform that requires the developer for its use, in the first case (Devops) only specific tasks of web application developers. In the second case (NoOps), it does not need operations related to configuration, installation or maintenance of the frameworks supporting web applications. Famous example of PaaS are Heroku [11], Google App Engine (GAE) [12], Windows Azure [13], and Cloud Foundry [14]. These providers have Application Programming Interfaces (APIs) that are exported to applications allowing to deploy, run and manage applications on the platform. They consist of a vast software layered architecture built atop any infrastructure that abstracts the scaling, management of instances, and other processes that routinely are needed for hosting the application. In this sense, PaaS can dramatically reduce the complexities of deploying a hosted application. The promise of PaaS is to focus the work on the code (business logic) and leave the framework the automation in the application' lifecycle. The goal is a productive approach that could improve the work, since developers deploy application to the cloud without caring about its management.

Despite this, currently the user do not choose PaaS solutions, but prefer IaaS solutions in order to guarantee full flexibility and control of the software. The terms NoOps, popularized by a PaaS society [15], can be then associated to hosted applications that are built entirely on a PaaS framework. Many instead when talking to a PaaS environment, prefer a term like AppOps, which is more descriptive of where the operations actually are. The term means that all developer tasks are associated only to the application context.

In a PaaS deployed solution, a developer should focus on the code, although it must then, however, know the platform, install and configure needed software in order to deploy and run the application on the platform. In any case, the required operation are automated through specific software by the PaaS solution.

3. METHODOLOGY

PaaS seems to be devoted to application programmers. Its primary function is allowed to build and run web-based and custom applications in an on-demand fashion. Furthermore, developers have access to specific tools and libraries, while controlling software deployment and configuration settings. The company offering the Paas service provides the other underlying utilities (e.g., networks, servers, and storage) that are needed for the application execution. The external hosting allows for easy scaling based on paying a fee depending on how you use. Cloud solutions that take accounts of DevOps requirements place themselves between the IaaS and the PaaS layers. We can distinguish some PaaS categories depending on their main purpose:

1) Design and deploy of software provided as standalone application development tools. The benefits of such offering include advanced security, scalability, no additional hardware costs, and no software licensing fees;

2) Social application development allowing an easy integration with social websites' APIs;

3) Improvement in the development of web-based application allowing to modify and add features to existing SaaS.

4) Provision of open-computing platforms for supporting applications written in different programming languages and using any type of database, operating system, and server.

The design, development, deploy and maintenance of PaaS-hosted applications are different from the multiple-steps process of traditional hosted application. The purpose of this study is to understand if the two methodologies are comparable and thus the use of PaaS platforms actually facilitates application development throughout its lifecycle, or if the required skills in the use of the tools introduces an additional level of requirements. Traditional development requires an effort to install, configure and then maintain the hosted environment which must accommodate the application, chosen between the different frameworks according the selected programming language. These tasks are time consuming and painful. The development on a PaaS platform requires steps to build the needed environment on the local machine in order to deploy the application. Usually these are operations performed using simple command line tools.

For example, Heroku PaaS solution requires that you install a software named Heroku Toolbet that includes a tool for creating and managing Heroku applications (Heroku client), an options for running the application locally (Foreman) and the Git software [16] that is the free and open source distributed version control system. Then, for supported programming languages (e.g., Ruby, Java, Python, Node.js, Clojure and Scala), it is needed the installation of additional software needed by the language. For example, if the application is written in Java, will be necessary to install the Java runtime kit (JVM) and the Maven 3 software that is a software project management and comprehension tool. Then the application must run on the OpenJDK version that is an open-source implementation of the Java software developer kit (JSE). Anyhow, PaaS solutions have some limitations, since it is not possible to access to features such as the core instances of the processes, storage, networking and routing that could be required by some types of applications such as the legacy ones.

Cloud offerings can be compared, as Fig. 4 shows, by considering two main driving forces behind the development of the cloud stack, that are productivity and control.

PaaS solutions (e.g., Cloud Foundry and Cloudify [17]) manifest a high productivity rather than IaaS solution (i.e. OpenStack [18]), that allow extensive control. The passage to major control also takes places by means of configuration management tools (i.e., Puppet, Chef) [19] that allow an automation of some phases of the installation process. Most of the PaaS solutions than allow different degrees of control, even if the primary goal remains the productivity.

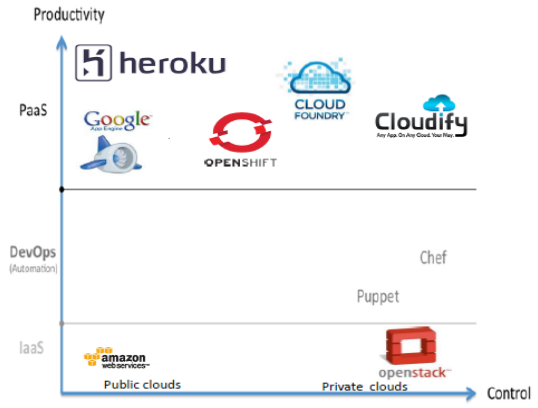

Fig 4 Driving forces behind cloud evolution and platforms examples

Before going into the details of PaaS solutions in order to highlight particular features that allow a cataloging, we remark that there are a large number of platforms. To assess the complexity of their use, we should focus on the differences between a cloud application and a web application. They share similarities, but a web application manifests a different approach going to a cloud environment.

3.1 Web application vs. cloud applications

A cloud application can be seen and an evolution of a web application, since it is accessed and executed on the web platform. But despite the similarities, there are some aspects that differentiate (Fig. 5). Web application are almost designed, as described before, by using different programming languages and web technologies. An application is hosted on a backend layered infrastructure made of server components (e.g., web server, application server, DBMS installed in virtual machines) and is usually executed from a web browser. The benefit of the web computing model is that a web applications are accessible from any type of device connected to the Internet and equipped with a web browser. A cloud application can also be accessed through a web browser, but the web interface is one of the different access methods. Not always, a cloud application is exclusively dependent on a web browser to work. Some cloud applications could be solely available over the web browser from service providers. In other cases, the web interface is used as alternative access method for application use and management, and the application can be accessed through other tools.

Cloud applications, as the Fig. 5 shows, are use other services provided in the cloud in their use (i.e., storage, infrastructure and core).

A web application can use a cloud service (i.e., an infrastructure service as a deployed environment), but generally a cloud application uses many cloud services along with locally installed services. For example in a cloud application, data is stored in a cloud/cloud-like infrastructure, but can be also cached locally for offline mode. A cloud application uses other cloud services offered by a specific platforms for data backup, security, and backup schedule. It can be used from the web browser and/or a custom built application locally installed. For example, cloud applications such Evernote [20] or Dropbox can be accessed either through a web browser that a program installed locally in the different device. They also work in offline mode, providing automatic data synchronization when the device is reconnected to Internet. Furthermore, a cloud application be used to access a wider range of services such as on-demand computing cycle, storage and application development platforms. So not all web applications could be considered also cloud applications, since cloud applications should have enhanced properties and support different requirements and needs.

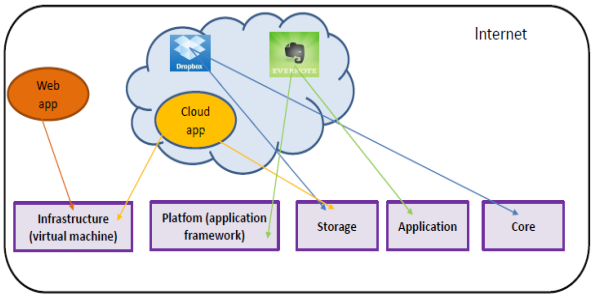

Fig. 5 Examples of cloud applications vs. web applications

Indeed some cloud solutions provide practices and tools to migrate a web application into the cloud. These functionalities allow for reducing time to market and software development complexities. Among the PaaS solutions classifications could be based on their private/public cloud position or language support but this limit the context.

In the next section, some PaaS solutions are analyzed, highlighting their peculiarities and trying to make a catalog that may help in the choice of the platform.

4. RESULT OF THE PAAS ANALYSIS

The plethora of solutions offered by, the market as PaaS framework created a fragmented world. We propose a first subdivision of the PaaS solutions in two lists: those that are related to some famous vendors (e.g., Google, Microsoft and Vmware) and therefore seem to be in some way vendor-lock and those that have been developed as open solutions (e.g., Cloudyfy). Other distinctions may be based on whether to deploy public or private cloud. Furthermore, the provision of the platform as a service is made in different ways: there are providers that focus on delivering PaaS through a self-service portal (a sort of service providers) to have a full control over the underlying infrastructure and the entire operation. Other providers give developers the tools to build an own PaaS on a private or public cloud (known as PaaS stack providers). In any case, these lists are not exhaustive, are continuously updated considered that the cloud market is evolving and just want to be an attempt to help you understand the differences between the many solutions.

Since all PaaS offerings are not equal, it is necessary to analyze the various options, though the difficulty of this analysis depends not only by the large number of solutions available in the market, but the fact that some PaaS functionalities are available only selecting and paying the fee for a plan. Moreover, some solutions have limited support to programming languages and frameworks and do not deliver key applications services needed for cloud application, or restrict deployment to a single cloud.

4.1 PaaS solutions in some way vendor-lock or limited

In the first categorization, we have put solutions listed in Fig. 6, such as Google App Engine (GAE), Windows Azure, Heroku and CloudFoundry. GAE belongs to the Google services ecosystem. App engine provides an application environment to build, maintain and scale primarily Java and Python applications, although lately is including support for Go [20] and PHP programming languages. The other programming languages (e.g., JavaScript and Ruby) are not natively supported, but run only if Java virtual machine is enabled. The engine provides dynamic web serving, persistent storage, automatic scaling and load balancing and a local development environment that allows to simulate the engine locally on a computer for application testing. In this way, the applications can be uploaded on the Google infrastructure that acquires the management. The applications run in a secure environment (the sandbox) in order to distribute web requests to applications across multiple server and guarantee security and reliability. Moreover, the engine provides a range of options for storing data that are a particular datastore with a NoSQL schemaless object datastore, a relational database (Google Cloud SQL) and a storage service for objects and files knows as Google Cloud Storage.

PaaS solutions in some way «vendor-lock» or limited

Name	Features	Link
Oracle	It provides a platform that is both elastic and scalable while allowing users to consolidate new app deployment and development. It includes a database based on Oracle db and oracle extended database machine	www.oracle.com/us/technologies/cloud/oracle-platform-as-a-service-408171.html
Windows Azure	Microsoft's PaaS allows users to build, scale and deploy web app using Microsoft's data centers. It is also an open source solution which allows developers to use any language, tool or framewok	http://www.windowsazure.com/en-us/
Google App Engine (GAE)	Allow users to develop and host web apps in google data centers. It offers automatic scaling of web apps and the platform will automatically increase the aomount of resources allocated as demand rises	https://developers.google.com/appengine/
Cloud Foundry	Is a subsidiary of Vmware. Suppports programming languages java and scale and Vmware offers a hosted service using this software that runs oin Vmare's own infrastructure and uses its vSphere suite packeage underneath	http://www.coloandcloud.com/cloud-foundry/
Heroku	Is a subsidiray of salesforce.com and was one of the first PaaS services offerend on the market. Support multiple languages (including Ruby, Scala, and Python)	http://www.coloandcloud.com/heroku/
Cloud swing	Is a subsidiary of OpenLogic and is portrayed as the first ever PaaS solutioj that offers cost-tracking and complete customization of technology stacks.	http://www.openlogic.com/cloud/
Acquia Cloud	A drupal tuned PaaS that enables devs to customized its capabilities including workflow, site management and provisioning	www.acquia.com/products-services/acquia-managed-cloud
Longjump	PaaS hosted in the Longjump cloud with access to multi-tenant arch, scalable capacity, industry-standard tech stack	www.longjump.com/
CloudBees	Simple way to build and run Java apps.	www.cloudbees.com/

Figure 6 A list of PaaS solutions tied to famous vendors or with limited functionalities.

Depending on the selected language, there are some operations that a developer should do about the configuration of the development working environment and the installation of specific tools like the Google App Engine Service developer kit (SDK) which allows the full use of the framework. A GAE app may use the Google web toolkit (GWT) that is Java/Ajax library helping to write java code that gets compiled into Javascript and thus can be run in a web browser. This tool that in the starting plan is free for developers, allows to build and host web apps for faster development and deployment processes.

Once loaded into the platform, the control is done through a web admin console that give the complete access to the public version of the application allowing performance settings and logging.

Even if such a solution provides guarantees related to Google's infrastructure (the hosting on Google own servers is someway synonyms of availability and scalability), and application of this type is strongly linked to the vendor. The solution seems so useful for the specific type of applications, considering also the payment method based on quotas for the use of resources (i.e., storage, cpu and bandwidth) of which a set amount given free. It is possible the porting of a Google app engine application in another platform, even if it requires some efforts [21]. Microsoft Windows Azure is not only a PaaS solution (labelled as Cloud services) offered by Microsoft, but a set of cloud services offering infrastructures, storage and platforms. Azure distinguishes between Azure Web Sites and Cloud Services. Azure Web site allows to build scalable website by means of Azure Portal or the command line tools to set up a website using .NET, PHP or Python programming languages. It provides third-party applications such as Content Management Systems (CMS) [22] like Drupal of Wordpress and thus it is usable for this type of application. Cloud Services provide a PaaS environment and thus is usable for multi-tier web applications. This last solution needs, as the other PaaS, the setting of local development environment (e.g., Visual Studio for programming with .NET languages) before deploying an app to Windows Azure. Data can be stored along with a relational database SQL or MySQL. Moreover, Microsoft Azure provides other ways to store data that are a NoSQL approach based on key/value store and blobs that are designed to store unstructured binary data.

Next to these two services, there is Virtual Machines that are the Microsoft IaaS offering. The three approaches can be used separately or combined. The virtual machines are based on Windows operating systems environment and thus websites are managed by Microsoft web server that is Internet Information Services (IIS). Furthermore, since the PaaS solution supports other programming languages than the .NET platform, Cloud Services, allows the deployment of other frameworks (e.g., PHP-based) that, however, requires custom deployment tasks to install them. The cloud service of this option consists of a frontend web role and one or more worker roles.

Azure Web Sites and Cloud services have similar use-cases and advantages, especially considering web sites as application to port in a cloud solution. Such solution offers great opportunities even if related to Windows environments. The pricing and plans, given that there is a free trial, is somehow complex, though is typical in the calculation of the costs in the cloud offerings that are based on the resources' use. For example, the plans for the Azure Web Sites are plans to 6 or 12 months with a consumption of resources (e.g. database size, bandwidth) per hour. In any case, these cloud applications are related to a Windows environment.

Heroku is a solution that we inserted in this list as it was chosen by Facebook as cloud PaaS provider. When creating a new Facebook application, developers [23] can choose Heroku as web hosting developer provider. Furthermore, it is one of the first PaaS cloud platforms developed. Originally born as a Ruby environment, it now allows the deployment of applications written in other programming languages such as node.js, Java, and Pyhton. Heroku offers the highest number of NoSQL solutions in the market (e.g., Cloudant, Membase, MongoDB and Redis), but also an SQL-based approach with the PostgreSQL database. As a PaaS cloud, it requires the installation of local development environment (the Heroku toolkit) that allows the build and deploy of the application in the Heroku platform. Once deployed, it is possible to control the application workload and lifecycle directly. The codes can be deployed and completely managed also by adding or

removing resources on the fly via add-ons. So applications can be scaled. It allows operations by using a command-line interface web console or full REST API. Heroku will run the application within the dynos that are isolated virtualized Unix containers that provide the environment required to run the application. Since the dynos are the unit of computer powers, they are one of the resources that are calculated in the definition of the price of use of the platform. There is a starting free plan at the start, then the price scales with dynos, database size and other resources requested. In any case, this is a pure PaaS solution. Finally, the last PaaS solution examined in this list, is Cloud Foundry [14]. It was set in this list, since developed initially by WMware as the industry's first open PaaS, but now is delivered as an open PaaS made available by Pivotal and partners. It represents an ecosystem, since it allows to use cloud services of its partners. As a PaaS it supports applications written with several programming languages (Java, Ruby, Node.js, Scala, PHP and Python) through various industry standard frameworks (Spring, Ruby for Rails, Grails) via build packs. Moreover, it offers database management support through relational and no-relational databases (PostgreSQL, MySQL, MongoDB, Redis). Finally, it provides also an IaaS hosting by means of the partners, since a developer could choice the cloud for deployment across public, private and hybrid clouds provided, for example, by vSphere [24], AWS or OpenStack.

The procedure for its use as a PaaS, currently depends on the choice of the hosting in the Pivotal CF or to Cloud Foundry. In any case, it requires the installation of the software needed to interact with the deployed platform (e.g., the Cloud Foundry CLI) together with getting an account to the available platform (i.e., run.pivotal.io). Then there are a set of steps needed to configure the environment for the selected programming language and framework. These offerings have a variable cost, since related to different partners. At the beginning, all services have a free plan, then according the different service providers there are different costs and prices. Currently this solution seems to be changing so maybe it could be used for a beta application. However the use of such platform requires knowledge of the underlying architecture and the environment that being related to different platforms seems not so simple.

4.2 PaaS solutions that declare to be opens

The second list includes PaaS solutions that declare to be based on an open stack. We analyzed, for example, Appfog [25], AppScale [26], Cloudify and Openshift [27]. Appfog is a cloud platform for web applications builds on CloudFoundry, which is an open PaaS project. It supports PHP, Ruby, Node.js and Java application that could be deployed in the cloud platform provided by its offering. The access is through a web console interface (App or Framework Jumpstarts), or through the command line interface that is locally installed with the AppFog (AF) command line tool. The App or Framework Jumpstarts is an easy way to use a PaaS platform. It allows to choose the type of application and the environment before to start with the build and deploy. A developer could choice an application from a list (e.g. including languages like Java or PHP, frameworks like Ruby on Rails or Python Django and CMS like PHP Wordpress or PHP Drupal), then an infrastructure vendor (e.g., from a list of vendors provided by AWS and HP in a beta service), and finally a sub-domain where to deploy and access the application. AppFog offers details pricing plans based on the used resources such as number of server instances, database storage size and SSL endpoints. The quote is based on a fee per month. There is also a starting free plan that allows to deploy unlimited apps within 2GB of RAM, 100MB of database storage (both SQL-based or NoSQL based), 5GB of data transfer per month and 100 requests per second. It seems to be a good solution for testing purpose and probably for small apps.

AppScale, instead, is defined as an open source, GAE compatible, private clouds. It implements Google's Cloud Platform in an open source. AppScale according Tim O'Reilly [28] could make part of a new stack-acronym as LEAP that stands for Linux, Eucalyptus [29] that is an open source private cloud software compatible with AWS, and Python. It shows an open stack architecture based on a variety of open source technologies. These technologies provide traditional web hosting and serving (e.g., nginx web server and HAProxy with Hadoop [30]) as well as supporting different datastore (i.e., Cassandra, MySQL, MongoDB). The limitation is the supported programming language that is only Python. AppScale software stack is available as a single virtual machine image, with different formats for private clouds (e.g., VirtualBox, OpenStack, Eucaplyptus), and public clouds (e.g., AWS and Rackspace [31]). It. declares to be devoted to business critical applications, since deployed in a private cloud and able to provide fault tolerance and elastic automatically by adding a virtual machine.

PaaS solutions that seems to be «open»

Name	Features	Link
Cloudyfy	Open source PaaS stack –that sits between the application and the chosen cloud. Supports any application stack and database store (relational and non-relational such as Apache Cassandra). Support public clouds (amazon E2c, windows azure, rackspace) and private clouds (openstack, cloud.com, vmware vcenter, citrix xenserver)	www.cloudifysource.org
AppScale	Open source framework that support Google App Engine apps. It allows users to upload users to upload numerous apps to the cloud and support multiple distributed backends (MySQL cluster, Hbase, Hypertable, Apache Cassandra) and Python, Go and Java apps.	appscale.cs.ucsb.edu/
AppFog	Open source Paas solution (located in Portland, Oregon, US) that allows developers to build and deploy apps in seconds	http://appfog.com/
Openshift	Openshift is a free, autoscaling PaaS for building apps. A subsidiary of Redhat, it manages the stack, which gives developers the ability to focus all their attention on coding	https://openshift.redhat.com/app/
dotCloud	Allows users to build and deploy any app on the cloud, while having the ability to manage it all in one place. Therefore users can mix and match languages, database and message components witout the need to purchase additional servers.	https://www.dotcloud.com/
Slipstream	Engineering Paas fully automated and on-demand. It allows uses to create multi-machine runtime env. It provides an easy to use interface.	sixsq.com/products/slipstream.html
ConPaaS	Allows to easily run apps in the cloud using a runtime env. Services are elastic and can be scaled up or down with a click	
Engine Yard	Reliable and secure Paas solution for any type of web or enterprise app. Includes the Ruby on rails stack, automated load balancing and monitoring	http://www.engineyard.com/

Fig 7 List of cloud offerings declared as based open solution

As the other platforms, it provides a set of tools called AppScale Tools that allow handling and managing with the application locally or directly on a cloud platform. A solution of this kind is useful, even if there is no support for other languages.

Another PaaS solution is Cloudify that declares to be the open PaaS stack that brings DevOps automation to the world of PaaS. It enables the deployment, management and scaling of all types of applications on any type of cloud with no changes to the code or architecture. The Cloudify approach is based on a recipe that is a container for all useful information about the application (the application descriptor file, interdependencies between tiers, deployment process, and so on) that can be ported to other cloud allowing the scaling. Each recipe is uploaded on a virtual machine and processed by Cloudify agents that install also the application tiers and integration software by using Chef software. The recipes include also web server (nginx) and datastore (PostgreSQL, CouchDB, Redis). The application can be installed with a single shell or web services based command. Next to the virtual image, it is

required the installation of the Cloudify shell that, implementing a local cloud, is an emulation environment. Cloudify shell allows to run Cloudify management and applications services on a single machine. With this tool, it is possible to get the available recipe where to deploy the application and thus monitoring the application with a web management console. Apps are considered as a set of services and a service is composed of a least one service instance, each one on one host, whereas more instances can span multiple machines. Once the recipe is created, it can be run on any cloud, and any middleware platform. Depending on the cloud platform selected (Azure, OpenStack, HP Cloud [32], RackSpace), it is necessary a custom configuration. The limitation in this software is the supported language that in this case is the Java language.

Finally Openshift, should be inserted in the first list as related to Red Hat, but declares to be an open hybrid cloud application platform. The solution is offered as a public PaaS (Openshift online), a private PaaS (Openshift enterprise) and community PaaS (Openshift origin). Focusing on this last solution, Openshift origin is a free and open source PaaS that are available as a downloading image suitable for running on a virtual machine with different formats. Moreover, it is possible to build an own machine using Puppet. It supports a wide variety of language runtimes (i.e., Java, Ruby, PHP, Python, Perl) and data layers (MySQL and PostgreSQL). It is founded on RedHat Enterprise Linux (RHEL), and a set of instances of RHEL called Openshift nodes are where user applications reside. Multiple nodes are managed by brokers. A developer can try the software installed in Redhat hosted public PaaS requiring an account.

As seen, the different solutions manifest various features including the use on public, private or hybrid cloud. The first list offers PaaS solutions mainly on public cloud, while the second in a private cloud.

Other examples of PaaS offerings in a private cloud are reported in Fig. 8.

Private cloud vendors	features
Apprenda	Enterprise-friendly provider of private PaaS for .NET, SQL Server, and IIS applications. It offers a deploy-anywhere multi-tenant platform and application multi-tenancy.
Ubuntu private cloud	Provides a development platform, mainly based on Python even if supports other languages (C++, Java, Ruby, PHP).
Red Hat Openshift	PaaS software layer that Red Hat runs and manages on top of third-party providers (e.g. including Amazon) It is focused on the runtime level, rather than being framework-oriented.
Platform ISF	Offers management software for building private IaaS clouds.
Longjump	Provides a development environment entirely based on open software that could introduce a cost for control and governance. Its business application platform is available as customer-installable licensed software.
Amazon virtual private cloud	It allows users to set up and access virtual servers via a simple Web interface. EC2 is designed to work in conjunction with Amazon's other cloud services.

Fig 8 List of private cloud offering

Our categorizations are however limited since some offerings manifest intersections. For example, most private cloud providers deliver PaaS in a public context, which means that they operate the platform as a third party. In the perspective of an open solution, OpenStack is positioned as the cloud operating system.

4.3 PaaS solutions and the cloud operating system

Openstack is an open source software for buildings private and public clouds. It is defined as cloud operating systems that control large pools of compute, storage and networking resources throughout a datacenter managed with a dashboard. View. It could be used to configure a private cloud by means of the installation and configuration of software packages on different operating system Linux-based supported. It is possible to star with this complex software using public clouds thanks to cloud vendors (e.g., HP, Rackspace) that, however, require a sign up and the payment of a fee or in a local environment with devstack.org. Devstack is a shell script that allows for a build a complete Openstack development environment that is based on Linux operating system (e.g., Ubuntu or fedora distributions). Devstack is available also in a virtual machine format. However, this type of installation requires a series of tasks very onerous together with high technical skills to use the entire infrastructure.

5. CONSIDERATIONS

According the classification made before about PaaS service providers and Paas stack providers, we cloud put GAE and Heroku on the first category as PaaS service providers and Cloudify, Cloud Foundry or OpenStack in the second one as PaaS stack providers.

A further sub-category of classification would be on the level of openness and control, i.e. open source, language support, cloud environment support, DevOps support and such.

With these types of deployments, there are, however, an interaction with the server at the level of the operating system. While this gives tremendous power and flexibility to tune the server, manage security and access control, and configure clusters, it also requires some amount of an initial configuration and ongoing maintenance (typically carried out by a system administrator or the sysop).

One of the key differentiators for PaaS-style hosting as compared to traditional virtual or hardware server hosting is the deployment process. Many developers have embraced git as their version control system of choice, and many of the PaaS offerings out there have taken advantage of git's popularity, since the deployment process is transformed into a simple git workflow.

In any case, it is very difficult to analyze and compare the available solutions without a real testing of the platforms. And a real testing of the platform in a public cloud, it is possible only afterregistration and a test of the environment that in the free plan could be only limited. Usually developers do not deploy new code changes or bug fixes directly to the live site or production server, but they test any changes locally and then push the code to a shared integration or testing server for verification. Also for these reasons many PaaS providers offer low-end hosting options for free that are suitable for a number of purposes (e.g., a shared environment for verifying bug fixes, an area to develop rapid prototypes, sites without specific performance or scaling needs). Afterwards as an application's needs expand beyond what the free offering provides, most providers structure their fees based on performance needs as well as data and bandwidth usage that the application incurs.

The pricing structure for PaaS offerings tends to be based on what you need to support your application. This includes items such as:

- Number of web hosts (or instances, or dynos) to support the requirements;

- Number of background processing instances for bulk processing;

- Number of database instances as well as data storage needs;

- Add-on services such as caching, hosted error reporting, email delivery, etc;

- Bandwidth usage.

In addition, many PaaS solutions offer options for having either shared or dedicated resources for the web application.

Probably the best thing is to consider the type of the application to migrate in a cloud environment and platforms that we know or that are adopted in a similar context. Then, it would be possible to decide according the application type the selection of a PaaS service provider or a PaaS stack provider. For example if our application is a website implemented as a CMS software, we could use GAE, Azure or Appfog that provide CMS support in an easy way. For example, as the Fig. 9 shows, AppFog solution provide a free plan, and we could choose Drupal or Wordpress as Web CMS framework when choosing from a web interface the type of applications that we would like to deploy on the PaaS platform.

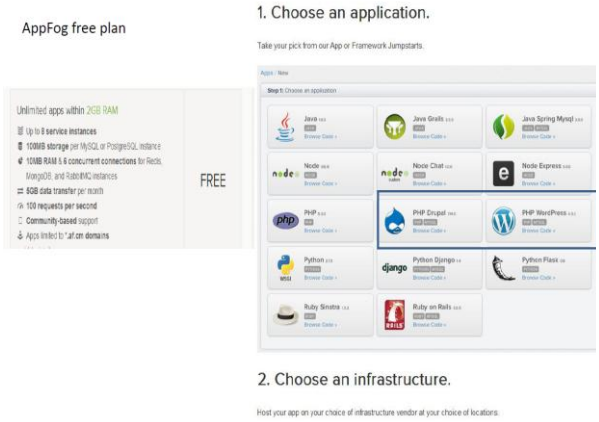

Fig. 9 The AppFog free plan and the type of application and frameworks supported.

Probably with the trial version of Azure and the free plan of Appfog, we can make a real test of these platforms and have data for comparison between different solutions.

Another approach that we need to consider is the implementation of the Openstack software considering its use in specific educational and research environments. This software has been adopted by CERN [33], a research institute focused on scientific disciplines, to implement a private cloud environment. Moreover, the European Grid Infrastructure that implements the grid computing distributed model [30] is studying the way to offer its computing resources in a cloud perspective. And working towards the adoptions of an open stack is seeing at solutions like Openstack.

6. CONCLUSIONS

We are approaching cloud offerings from the web developers' perspective, since these environment could provide faster development and easy management. PaaS combines both the server that is hosting an application with a desired stack of data repository and web application technologies to create a platform on which you can deploy an application that uses the given technologies. Typically a PaaS offering will provide a suite of options for various layers of the application stack, for instance, by offering both MySQL for a relational data store, and MongoDB for a more document-oriented data store. These storage options could be combined with a model-view-controller style web framework such as Ruby on Rails, Groovy on Grails, Spring MVC on Java, or a multitude of other technologies. There is a plethora of PaaS offerings for a variety of technologies. We have tried to categorize such offering basing on In general PaaS aims to simplify the effort required to provision, maintain, and deploy a high-performance web hosting environment, however, this simplification can come at a cost. For security and maintainability PaaS providers often lock down their service to a specific set of software versions (typically the most recent versions in any software stack with up to date security patches). This behavior could not be useful for many developers that are managing legacy applications that rely on specific older versions of web technologies. PaaS tends to favor popular open source tools; therefore, it generally isn't possible to host commercial applications with PaaS providers. In our context, we have decided to approach these offering with two perspective: the adoption of a PaaS solution where to deploy an application like a web CMS, and choose an open stack cloud solution that could allow to implement a private cloud.

7. REFERENCES

[1] Buyya Rajkumar, Broberg James, Goscinski Anrzej, 2011. Cloud Computing: Principles and Paradigms, Wiley & Sons.

[2] Governor James, Hinchcliffe Dion, Nickull Duane, 2009. Web 2.0 architectures, O' Reilly.

[3] Sanders Bill, 2010. Smashing HTML5, Wiley.

[4] Dropbox website, Avaliable at: http://www.dropbox.com.

[5] James Murty, 2008. Programming Amazon Web Services (AWS): S3, EC2, SQS, FPS and SimpleDB

[6] Humble Jez, David Farley, et. al., 2010. Continuous Delivery: Reliable software releases through build, test, and deployment automation, Kindle version, Addison-Wesley Professional.

[7] Seifeddine Mohamed, 2009. Introduction to Groovy and Grails. Opensourceconnections.com.

[8] Viral Patel, 2010. Introduction to Spring 3 MVC Framework. Available at: viralpatel.net blog.

[9] Tom Hughes-Croucher, Mike Wilson, 2012. Node: Up and Running, O'Reilly media.

[10] Serena Pastore, 2013. Web-oriented data formats and their management in the mobile era, Mobile Computing Journal, Volume 2, Issue 2, May 2013, pp.29-42. ISSN: 2325-7423, 23257431

[11] Heroku website, Available at: https://www.heroku.com

[12] Sandersin Dan, 2012. Programming Google App Engine, Second Edition, O'Reilly.

[13] Roberto Brunetti, 2011. Windows Azure Step by Step, Microsoft Press.

[14] Cloud Foundry portal. Available at: http://www.cloudfoundry.com

[15] Derrick Harris, 2013. Why 2013 il the year of 'NoOps'for programmers, Gigaom, Available at: http://gigaom.com/2012/01/31/why-2013-is-the-year-of-noops-for-programmers-infographic/32

[16] Scott Chacon, 2009. Pro git. Apress.

[17] Cloudify website. Available at: www.cloudifysource.org

[18] Kevin Jackson, 2012. OpenStack Cloud Computing Cookbook, Packpub.

[19] Evernote website. Avalilable at: https://evernote.com

[20] The Go programming language. Available at: http://golang.org

[21] The Unofficial guide to migrating off of google app engine. 2011. Available at: http://www-cs-students.stanford.edu/~silver/gae.html

[22] S. Pastore, 2012. Website development and web standards in the ubiquitous world: where are we going? WSEAS Transactions on Computers, Issue 9, Volume 11, September 2012. pp. 309-318. E-ISSN: 224-2872Jesse Feiler, How to do everything: Facebook applications, Mc-GrawHill, 2008.

[23] WMware vSphere. Available at: www.wmware.com

[24] Barb Darrow, 2002. AppFog lets you pick your cloud, (almost) any cloud, Gigaom.com blog.

[25] AppScale website. Available at: www.appscale.com

[26] OpenShift by redhat website. Available at: http://www.openshift.com

[27] Prabhakar Chaganti, 2010. Cloud services for your virtual infrastructure, Part2: Platform as a Service (PaaS) and AppScale. IBM developerworks library.

[28] Eucalyptus Open source. Available at: http://www.eucapyptus.com

[29] Josette Rigsby, 2012. HBase, Node.js, nginx, Hadoop make big enterprise inroads. CMSWire web magazine.

[30] Rackspace website. Available at: www.rackspace.com

[31] HPCloud services. Available at: http://www.hp.com

[32] Archana Venkatraman, Case Study: CERN adopts OpenStack private cloud to solve big data challenges, ComputerWeekly.com, 2012

[33] Pastore Serena, 2001. Distributed computing platforms like clouds and web standards: what could be the solution in an open environment?, Proc. Of the WSEAS International Conference on Recent Research in Applied Computer and Applied Computational Science (ACACOS 2011), 8-10 March 2011, pp. 195-200. ISSN: 1792-8559