Survey on Open Source Platform-as-a-Service Solutions for Education

Pavel Kriz
Faculty of Informatics and Management
University of Hradec Kralove
Rokitanskeho 62
500 03 Hradec Kralove
Czech Republic
pavel.kriz@uhk.cz

ABSTRACT

While the cloud computing becomes popular in the industry and companies take advantages of Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) as well as Software-as-a-Service (SaaS) solutions, education is sometimes one step behind. SaaS cloud solutions are commonly used in education since they provide well-established services like email, instant messaging, shared calendar. But IaaS and PaaS are rarely taken into account while legacy hosting services are still popular with students and lecturers of computer science. Majority of these users will actually not benefit from IaaS as they simply want to run their own or existing applications. This survey tries to answer the question which open-source PaaS solutions should be taken into account for demonstrations and practice in courses focused on web applications. Suitable PaaS platform can also be deployed at the university's IT infrastructure and used to host projects of students, researchers and lecturers. Google App Engine, OpenShift, Cloud Foundry and Heroku platforms are being investigated and their features and pricing are being discussed in the context of use in education. This article summarizes supported languages and services, porting-related issues and platforms' restrictions, pricing (focusing on low-cost plans) and private cloud installation options and complexity. The results show that there are differences that make some of the platforms more suitable, particularly because of different associated costs and different complexity of porting existing applications to the cloud.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Cloud Computing—*open-source Platform-as-a-Service solutions*

Keywords

Platform as a service, Cloudware, Open source, Education

IDEAS'14 July 07-09 2014, Porto, Portugal

http://dx.doi.org/10.1145/2628194.2628245.

1. INTRODUCTION

Cloud computing solutions become a common approach to provide necessary services to the users. Three main types of services are provided by the cloud: Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), Infrastructure-as-a-Service (IaaS) [17]. Some authors also define the fourth type: Hardware-as-a-Service (HaaS) [16].

SaaS offers existing ready-to-use applications accessible via the Internet. Google Apps and Microsoft Office 365 are typical SaaS solutions widely used in education providing email, instant messaging, (shared) calendar and documents. Though SaaS is excellent for these well-established services, it is not much flexible and customizable. It will not allow creative students or researchers to face new challenges and find unique solutions.

IaaS provides remotely delivered (via the Internet) computer infrastructure, typically virtual computers (Virtual Machines, VMs). Users of IaaS are required to know the full software stack to run their own applications. IaaS is suitable for projects requiring a special deployment setup or for building a PaaS cloud on top of the IaaS cloud. Majority of students' projects and a great part of the researchers' and lecturers' projects do not benefit from IaaS. IaaS solutions also require regular maintenance operations like applying security updates to the underlying operating system and application's runtime environment. The application may become vulnerable when the maintenance is omitted. Unfortunately not all IaaS users in education are rigorous enough to carefully maintain their IaaS VMs.

PaaS is the missing link here between IaaS and SaaS. It provides runtime platform and all required services (e.g. a database) for custom and pre-configured existing applications while maintaining the underlying operating system and runtime environment. PaaS solutions also support vertical scaling capabilities by increasing or decreasing resources (RAM, CPUs, etc.) granted to a single instance. Vertical scaling may also be provided by underlying IaaS. Several PaaS platforms also provide horizontal scaling capabilities by adding or removing additional instances to a distributed application.

Figure 1 shows simplified structure of the users of cloud computing services at the typical university. While it is easier to provide users with VMs (manually managed or using private IaaS) today, PaaS cloud will probably be better adopted by regular users at the university meeting bigger success in solving problems.

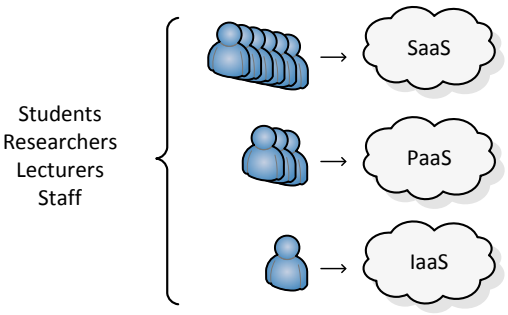

Figure 1: Simplified structure of the users of cloud computing services at the typical university

The goal of this survey is to find a suitable open-source PaaS solution for education. The analysis of challenges is mainly based on experience of Faculty of Informatics and Management, University of Hradec Kralove. This survey focuses on requirements of ongoing research and courses on programming at the university.

In contrast to other papers, we focus on aspects that are important when using PaaS in education. These mean the availability of the source code for the study purposes, easy installation of a local private cloud, affordable plans for a public cloud, etc. Furthermore, we reveal relations between commercial products and open-source products. In some cases, a part or a whole commercial solution's code is open while sometimes open-source community creates alternatives to commercial solutions.

This paper is organized as follows: Section 2 introduces challenges we face when choosing a PaaS solution for use in education. Section 3 sums up related surveys and papers. Section 4 introduces the main open-source PaaS cloud providers and technologies. In section 5, we discuss common and different features of several PaaS solutions, highlight the key issues when porting applications for PaaS cloud and give a short recommendation regarding the choice of a solution for education. Finally, a conclusion is presented in Section 6.

2. CHALLENGES IN EDUCATION

According to the web statistics [12], the most popular programming languages are C, Java and PHP. While C is a language primarily used for system programming, Java and PHP are popular with students when they develop desktop or web applications. Java is an obvious choice as students gain knowledge of Java language during main programming courses at the Faculty of Informatics and Management. PHP is a popular language for developing custom web applications and many existing applications are already available (e.g. WordPress) and may be easily customized as students gradually increase their programming skills. Therefore, discussed PaaS platforms should support both Java and PHP.

The expected ways of using PaaS cloud in education are:

- Teaching computer-science students how to develop applications for clouds and use cloud-specific technologies.
- Hosting students' projects developed during programming courses, for use in the theses or as a part of research.
- Hosting students' projects based on pre-configured applications (hosting WordPress or other popular applications, providing near-SaaS experience with options to fine-tune or easily enhance or customize the application).
- Hosting applications developed by researchers, lecturers or faculty staff.
- Teaching students how to use cloud solutions in bussiness projects.

The following features of the PaaS cloud are required according to the use-cases mentioned above:

- Public and private cloud solutions should be provided in order to run private cloud at the organization using its resources and IT infrastructure as well as to give an option to run applications in the public cloud when necessary.
- PaaS platform should be open-sourced allowing a free installation, customization, and study.
- Free or low-cost plan should be available for testing and academic use of public cloud.
- Free or low-cost plan should be available for non-academic projects of students making the transfer of their knowledge into the business easier.
- Platform should support Java and PHP as these are popular languages for web application development.
- Required modifications of existing applications should not be too difficult. Difficult optional modifications (in order to benefit from cloud-specific technologies) are acceptable.
- Platform should support a relational database as a service, because existing applications usually rely on it.
- Platform should provide a mechanism to limit and manage resources consumed by applications to avoid overloading of private cloud.
- The installation of the platform should be well documented and straightforward.
- Micro-cloud solution for local (offline) testing and experimenting should be available.

Several PaaS providers and technologies are described later in this article with respect to these requirements. PaaS clouds that do not meet more of these requirements are not taken into account.

3. RELATED WORK

Several comparisons of cloud platforms have already been made. However, many of them are focused on IaaS (including open-source platforms [7]), SaaS [17] or sum up a whole cloud industry including all three types of cloud services [16]. Differences in methodology are also described as there are *service feature-emphasized*, *metrics-emphasized*, *benchmark-emphasized* and *experiment-emphasized* studies [13]. Some

authors focus on PaaS platforms [15] although not in context of education. Evaluation of the use of IaaS and PaaS clouds in higher education [18] shows significant benefits however only one IaaS and one PaaS public clouds are taken into account in this work. Other authors [2] focus on the importance of virtualization in the teaching of computer networks.

This paper discusses key aspects of PaaS cloud platforms that are worth taking into account when choosing one for education rather than trying to do an exact quantitative multi-criteria comparison.

4. PAAS PROVIDERS AND TECHNOLOGIES

As the cloud computing has grown in recent years, several established providers and technologies exist. We focus on open-source solutions in this article and sum up their features, application porting issues and restrictions, pricing and private installation process. Products offering both public cloud service and option to install private open-source cloud were taken into account.

4.1 Google App Engine

One of the oldest cloud platforms is Google App Engine[1] (GAE), released in 2008. GAE provides a runtime environment for web applications in Google-managed data centres.

4.1.1 Languages and Services

GAE currently supports Java, Python, PHP and Go. Java and Python are officially supported (with some restrictions discussed later) while native PHP and Go support are experimental features. PHP was and still is also indirectly supported by Quercus runtime on top of Java Virtual Machine (JVM) similarly to other JVM-based languages such as Groovy, Scala, etc. GAE currently runs Java applications in Java 7 JVM utilizing Jetty servlet/web container.

Several methods for storing persistent data are provided. *Cloud Datastore* is fully scalable NoSQL data store based on *BigTable* distributed storage system [4]. *Google Cloud Storage* is scalable RESTful online web storage for *objects*, usually files. These objects are organized into *buckets* and the permissions may be defined in access control lists (ACLs). *Cloud SQL* is a highly-available relational database based on MySQL with replication support.

4.1.2 Porting and Restrictions

There are significant restrictions enforced by the GAE's runtime sandbox that are reasonable in cloud environment however they may complicate the porting of an existing applications to GAE.

First, only a subset (specified by so-called *JRE whitelist*) of the standard JRE classes is available. Applications using other classes need to be rewritten and recompiled. It is fully understandable that some classes related to low-level functionality irrelevant to cloud environment are missing. However, many developers complain about the missing classes from `java.awt` package. Consequently, image generation, PDF files generation and use of server-side visualization libraries is difficult with libraries incompatible with GAE.

Second, the request must be handled by the application within 60 seconds. Once this deadline has been reached, the request handler is terminated. Application may react to the exceeded deadline but it is highly recommended to ensure that the request is completed well before the limit (application is able to get the remaining time in milliseconds using GAE's API).

Third, application cannot write to local filesystem and needs to be rewritten to use Google Cloud Storage buckets, Datastore or another proprietary storage technology provided by GAE. This may be a serious issue when porting existing PHP applications to GAE as it is a common approach to write local files by PHP applications (e.g. content management systems) in order to save their configuration, implement caching and store uploaded assets as images. Typical installation scripts of existing applications may not work and a manual configuration is required. Porting process differs depending on the PHP runtime. When using native PHP runtime, the persistence code has to be rewritten to use the "gs://" protocol wrapper which is provided by the GAE to ease the adoption of Google Cloud Storage technology by PHP applications. Obviously the modification of the source code will result in issues during the application updates from original sources. When using Quercus PHP runtime, servlet-based request handler can be configured to implicitly wrap all file operations and use Google Cloud Storage buckets which results into smooth integration between existing PHP application and GAE's storage technology without the need of source code modification.

Fourth, although Cloud SQL uses MySQL and one can run applications depending on MySQL, there are some limitations [9]. File-related SQL statements (e.g. LOAD DATA INFILE) are not supported because of local filesystem access restrictions discussed earlier. User defined functions are not supported which may be an issue for applications with thick database layer implementing business logic and/or integrity constraints.

4.1.3 Pricing

Developers running applications on GAE are billed according to the resources their applications consume. Default (smallest) frontend instance has 128MB RAM and application gets one 5GB Google Cloud Storage bucket. Google has a very fine-grained pricing where particular resources are billed per gigabyte, per instance hour, per gigabytes per month, etc. depending on the nature of the resource (network traffic, running instance, data storage, etc.). It is also possible to run the application free of charge when no billing is set. However the application cannot exceed specified free quotas on particular resources. When billing is set, the free quotas are raised to "billing enabled limits" and a developer can raise particular limits above these free limits according to the needs of the application and his or her budget. Only resources exceeding the free limits are charged.

But Cloud SQL (MySQL) is not included in free quotas and it is always a paid service. Thus an existing application depending on a relational database will not effectively run free of charge on GAE. 0.36 USD per day (10.8 USD per month) is charged for small instance (500MB) of Cloud SQL.

4.1.4 Private Installation

While Google App Engine is a commercial, not an open-source product, an effort to create an open-source alternative implementation of GAE began at the University of California. The result of this effort is *AppScale*[2] [6], the platform that allows developers to deploy and run applications built

[1] `https://developers.google.com/appengine/`

[2] `http://www.appscale.com/`

for GAE anywhere outside of the Google's ecosystem, e.g. on own (virtual) servers, on virtual machines provided by third-party providers like Amazon or on virtual machines managed by open technologies like OpenStack or Eucalyptus. AppScale uses some parts of GAE available in Google App Engine Software Development Kit (SDK). This SDK is provided by Google allowing developers to test and run their GAE applications locally without actually deploying them to cloud. Besides GAE SDK, AppScale is built on top of existing open-source projects. More than ten storage technologies are supported [3], e.g. Cassandra [11] (NoSQL database) and MySQL cluster (relational database) but also MongoDB [5], HBase [8] and others. ACID-compliant transactions are implemented using ZooKeeper [10].

Several installation methods are described on AppScale's web site and wiki. Vagrant file with corresponding VM image is provided for quick setup of a single development or testing AppScale instance running inside VirtualBox. There are also prepared images for Amazon EC2, KVM-based VM and others. Bootstrap shell script for Debian/Ubuntu-based systems is provided as a generic installation method using no prepared image. Multi-node deployment is described in AppScale's wiki.

4.2 OpenShift

OpenShift[3] is a platform-as-a-service solution developed by Red Hat, initially released in 2011. There are three editions: *OpenShift Origin* is an open-source foundation for the other editions, *OpenShift Online* is commercial public cloud service and *OpenShift Enterprise* can be deployed into company's datacentre or a private cloud.

4.2.1 Languages and Services

OpenShift supports many languages, databases, services and binary applications that are able to run on underlying Red Hat Linux. Support for particular technology is always encapsulated into a *Cartridge*. Several cartridges are prepared for Java runtime (JBoss, Tomcat), PHP runtime (Apache with mod_php), relational databases (MySQL, PostgreSQL), NoSQL databases (MongoDB), services (Cron, Jenkins), etc. Existing applications can also be packed into prepared cartridges allowing non-developers to easily start favourite applications like WordPress on OpenShift platform (only few configuration steps are required). One can also create his or her own custom cartridges, though many additional cartridges have already been made by the community.

In contrast to platforms tightly coupled with particular technologies (like Google App Engine), OpenShift is literally open to any technology. With openness strong security must be applied. Application (with necessary cartridges) is therefore deployed into so-called *Gear*. It is an isolated environment built on top of several directories, Linux *cgroups* (implementing resource limits) and *SELinux* policies (enhancing security and isolation). At the beginning an empty gear provides no functionality except the isolation and resource limiting sandbox.

4.2.2 Porting and Restrictions

Since OpenShift Gear is actually fully functional sandboxed GNU/Linux environment, there are only few restrictions. On the other hand there are no "advanced" cloud solutions for horizontal scaling of file storage and/or relational database directly available because of such a simple architecture.

Even though it is allowed to write to gear's local filesystem, only OPENSHIFT_DATA_DIR directory is guaranteed to be persistent across several application's lifecycle operations. Thus all application's persistent data files should be stored there. But data directory is not replicated to other gears when application is set to be scalable. A new gear will have the empty data directory. Application's code is always obtained from its *git* repository.

When porting an existing non-scalable application to OpenShift, it must be modified to store all data files to the data directory. One can also deploy its database into the same gear. This is the simplest deployment scenario yet not scalable.

In contrast to Google App Engine, OpenShift has no direct support for MySQL scaling/replication. Scalable applications should use a shared database deployed to another gear according to the official documentation. This database-tier gear should be powerful enough to avoid a database bottleneck. But community-maintained cartridges exist supporting master-slave MySQL replication. Application-tier gears should also take advantage of some caching mechanism like *memcache* to avoid unnecessary database roundtrips.

There is no shared file storage available; the use of third-party solution like Amazon S3 is recommended. One can also implement custom solution for synchronizing files across multiple instances using *rsync* or similar utility.

4.2.3 Pricing

OpenShift distinguishes three "sizes" of gears with different pricing: small, medium and large. There are currently three plans for OpenShift Online public cloud platform hosting. "Free plan" allows developers to run their applications on 3 small gears (512MB RAM and 1GB of storage each) for free. "Bronze plan" is similar but user can order extra gears (small, medium or large) and additional storage, billed on per hour or per gigabyte per month basis respectively. "Silver plan" adds extra base storage for a monthly fee, allows to order more than 16 gears and provides ticket-based and phone support.

Pricing for OpenShift Enterprise, a fully supported private cloud solution, is not publicly available.

4.2.4 Private Installation

OpenShift Origin is the open source upstream of OpenShift Enterprise and OpenShift Online. Red Hat traditionally gives its development-stage upstream products to the open-source community. The relation between OpenShift Origin and OpenShift Enterprise is very similar to Red Hat Fedora and Red Hat Enterprise Linux. Though all editions should share the same source code, Red Hat's documentation claims different versions of supported runtimes and services in different editions.

Red Hat offers prepared images suitable for VirtualBox and KVM allowing user to quickly setup a development or to test OpenShift node running inside a single virtual machine. Bootstrap shell script for Red Hat-based systems is provided as a generic installation method without any prepared image. Puppet [14] script for Red Hat-based systems provides a fully automated Puppet-based installation. Though CentOS is also based on Red Hat Enterprise Linux (RHEL), instal-

[3] `https://www.openshift.com/`

lation scripts support only Fedora and RHEL distributions. CentOS system needs additional configuration in order to be prepared for OpenShift installation. An administrator has to add the *EPEL (Extra Packages for Enterprise Linux)* repository and install essential prerequisites to the operating system. The installation script tests automatically whether the system is ready to install OpenShift, but one has to prepare CentOS manually. Both script-based methods are also suitable for deployment to IaaS cloud. Finally, OpenShift's documentation also describes a fully manual installation for Red Hat-based systems (as this method uses repository with precompiled RPM files).

4.3 Cloud Foundry

Cloud Foundry[4] (CF) is an open source cloud platform as a service developed by VMware and initially released in 2011. *Pivotal* offers Cloud Foundry-based services on commercial basis and other companies build their solutions on Cloud Foundry too.

4.3.1 Languages and Services

Originally, Java, Scala, Ruby and Node.js were supported. In version two a new feature supporting any runtime was introduced. This feature uses Heroku *Buildpacks* to automate the deployment of an application along with its runtime environment. Community CF buildpack for PHP support exists and unmodified Heroku PHP buildpacks are also known to work with CF.

Cloud Foundry distinguishes two types of services: User Provided Services and Managed Services. *User Provided Service* allows developers to expose an existing external service (e.g. on premise deployed Oracle database) to the application in CF. *Managed Services* are services managed by and deployed to the CF platform. Some relational databases (MySQL, PostgreSQL), NoSQL databases (MongoDB) or other services (Memcached) are supported out of the box. Highly-available MySQL instance is provided by a third-party service called ClearDB. Custom Managed Services have to implement Service Broker API; they are then stored in the cloud controller database and prepared to be bound to the application. One can also make the Managed Service public allowing other platform users to provision own instances of the service.

As Cloud Foundry v2 is open to user-provided runtimes and services, strong security must be enforced by the platform. CF uses *Warden*, an abstraction layer managing container-based isolation and resource limits of applications deployed to the same CF instance (machine or VM). Although Warden is similar to *Linux Containers* (LXC), it is developed to be multi-platform by design. However, the reference implementation supports only GNU/Linux operating system using Linux *cgroups*.

4.3.2 Porting and Restrictions

Since Cloud Foundry uses similar sandbox model to OpenShift, they share similar restrictions regarding the storage tier.

Even though it is allowed to write to instance's local filesystem, the files are not guaranteed to be persistent across several application's lifecycle operations. The platform does not either replicate file modifications to other instances. Thus all application's persistent data should be stored in a shared relational or NoSQL database. One can also use a third-party cloud storage such as Amazon S3 or Google Cloud Storage.

Cloud Foundry makes porting of existing Spring Framework applications easier (as Spring is a part of VMware's portfolio) by intercepting the deployment process. CF detects datasource declared in Spring's application context and replaces it with the datasource pointing out to the bound database in the cloud e.g. MySQL managed service. Thus existing Spring applications require no modifications, even when using database.

4.3.3 Pricing

Since there are more public cloud providers using Cloud Foundry platform, prices differ across them.

The first provider, *Pivotal*, is directly related to Cloud Foundry as its services were originally offered on `cloudfoudry.com` until moved to `run.pivotal.io`. Pivotal does not offer any free plan, they charge 0.015 USD per hour per instance with 512MB RAM (10.80 USD per month). Other resources are free of charge (with limits) and a small instance (5MB) of ClearDB (MySQL) is also available without extra costs while large instances are charged.

The second provider is *AppFog* that was very popular because of its free plan offering 2GB of RAM, 8 service instances and unlimited number of application instances. But since 2014 they no longer offer free plan to new customers and they decreased limits for existing customers with free plan. The current smallest plan costs 20 USD per month and offers 2GB RAM and small instance (200MB) of relational database (MySQL or PostgreSQL). AppFog is built on top of Cloud Foundry v1 which does not support buildpacks so only preconfigured runtime environments are available to the user. AppFog also offers existing popular applications like WordPress or phpMyAdmin ready to deploy.

The third provider, *Anynines*, offers beta-stage service while pricing is already known. The smallest instance costs 2.50 EUR per month and offers only 64MB RAM and 10MB storage for relational database (MySQL or PostgreSQL) though such a small amount of RAM is not reasonable for Java applications.

4.3.4 Private Installation

Cloud Foundry platform provides its own installation and provisioning tools: BOSH and BOSH-lite. BOSH is fully featured tool for production deployments that automates provisioning to IaaS; VMware vSphere, VMware vCloud Director, OpenStack and AWS are supported IaaS providers while CentOS or Ubuntu may be installed in the VMs. BOSH-lite allows to deploy an instance of Cloud Foundry into a single machine or VM, typically for development, testing and evaluation purposes; VMware Fusion, Virtual Box and AWS are supported VM providers. BOSH-lite uses Vagrant to manage VM's lifecycle and the resulting VM runs on Ubuntu Linux.

Since authors of Cloud Foundry provide no prepared VM images for local provisiong of CF v2 (e.g. to Virtual Box VM), testing applications developed for CF v2 locally (offline) is difficult and requires multi-step build process including BOSH-lite-based deployment. Furthermore there are several issues related to the build process on Windows requiring modifications of the BOSH's source code [1]. Local

[4] `http://www.cloudfoundry.org/`

development and deployment were better supported by CF v1 because VMware Player image of so-called *Micro Cloud Foundry* was provided. However CF v1 is maintained no more.

4.4 Heroku

Heroku[5] was originally created as platform-as-a-service for Ruby in 2007. Heroku is currently developed by Salesforce.com. Heroku itself offers only public PaaS on commercial basis but there is community's effort to create an open-source platform compatible with and similar to Heroku.

4.4.1 Languages and Services

In addition to Ruby, Heroku added Node.js (JavaScript), Python, and Java (including JVM-based languages like Scala or Clojure) later. PHP and Perl are not officially supported but known to work. Runtime environments for other languages are supported by so-called *Buildpacks*. Buildpack encapsulates the necessary operations for detection of application's language, compilation and run. Buildpack is also responsible for installation of necessary run-time and build-time tools (including binaries) to the isolated operating system based on Ubuntu.

Services (e.g. relational database) are called *Add-ons* in Heroku. There are many existing add-ons ready to bind to the deployed applications. Besides other services, relational databases such as PostgreSQL and MySQL are supported. Heroku's PostgreSQL is ready to scale by adding read-only replicas (called *followers* of the *master*). Highly-available MySQL instance is provided by ClearDB implementation.

Heroku's user can also deploy prepared applications called *slugs*, that are already precompiled and bundled with all required dependencies.

4.4.2 Porting and Restrictions

While original Heroku implementation based on so-called "Bamboo" runtime stack provided read-only filesystem (with exceptions: *tmp* and *log* directories), new implementation based on "Cedar" runtime stack allows to write to the local filesystem though it is not guaranteed to preserve changes across several application's lifecycle operations and the platform does not replicate file modifications to other instances. Persistent data should be stored in a shared database or in a third-party cloud storage such as Amazon S3 or Google Cloud Storage.

4.4.3 Pricing

A free plan is available to run application using one "1X" instance (512MB RAM) and a small relational PostgreSQL database (up to ten thousands of rows). Additional instances are billed on per hour basis while price also depends on the size of the instance (RAM and CPUs). Database can be scaled up and is billed depending on the available storage, RAM and number of parallel connections. Additional services, like MySQL-based ClearDB, usually offer a free plan (storage up to 5MB in ClearDB) while additional resources are billed.

4.4.4 Private Installation

Despite the fact that Heroku is not fully open-sourced, it is built on top of many open-source technologies and tools with some of them developed specifically for Heroku.

Foreman is the official tool to test and deploy Heroku's application locally during the development, although without the support of isolation (which is not an issue in single-tenant environment). Foreman is also able to run the application on a local server taking care of the application's lifecycle with the aid of existing service-management solutions (like *upstart* or legacy *init* for Linux/Unix systems).

Dokku[6] goes further; it is a lightweight open-source alternative to Heroku built on top of Docker and Buildstep (it builds applications using Heroku's buildpacks). Docker is a container runtime for Linux using Linux Containers (LXC) to isolate environments running in the same physical machine or VM in a similar way to original Heroku. Dokku is intended for deployment to a single (virtual) machine and lacks multi-tenant features. Though Dokku has no web-based GUI by default, project *Quais*[7] is a simple web-based management GUI allowing to display application containers, start and stop them and display application images.

Dokku works well within a local Linux operating system, local VM and IaaS VM. For example, Dokku is directly supported by DigitalOcean, an IaaS provider supplying a Dokku image that turns its IaaS into a lightweight PaaS (note that horizontal scaling and multi-tenancy are not supported).

5. DISCUSSION

5.1 Comparison

Key parameters of described platforms are summarized in Table 1. These parameters are based on the requirements listed in Section 2. Providers do not distinguish between private and academic use of public cloud thus terms of use (limits, price) are the same in both cases. PaaS clouds vary in many ways. The critical issues will be highlighted here. Only OpenShift and Cloud Foundry are fully open-sourced. Open-source community-driven alternatives exist for GAE and Heroku, but they (of course) differ slightly from the original solutions. GAE does not fully support Java and Heroku supports PHP unofficially. The complexity of porting existing applications to GAE is high because of special handling of storage and only a subset of supported Java classes. Free plan is not available for Cloud Foundry public cloud and GAE does not provide relational database in free plan. Private installation and local micro-cloud deployment of Cloud Foundry v2 is complex and requires multi-step process using several tools and there are known issues with this process in Windows. Private installation of Dokku (also called "mini-Heroku") is not multi-tenant and does not support horizontal scaling. These issues make some PaaS platforms difficult to use in education.

According to the observed parameters, OpenShift is a promising PaaS cloud for education, providing the fully open-sourced platform and supporting many customizable runtime environments (including third-party runtimes). Porting of existing applications to OpenShift requires minimal changes in the code and many existing applications are already prepared for deployment to OpenShift. Installation of private OpenShift cloud and local instalation (for offline development and testing) are simple tasks thanks to prepared VM images. Public OpenShift cloud offers three reasonable instances for free, so students may deploy their (future)

[5] `https://www.heroku.com/`

[6] `https://github.com/progrium/dokku`

[7] `https://github.com/adrien-f/quais`

Table 1: Comparison of PaaS platforms

	Google App Engine	OpenShift	Cloud Foundry v2	Heroku
Open-source	No, open-source alternative AppScale	Yes	Yes	Partially, open-source lightweight alternative Dokku
Java support	Only subset of JRE classes	Yes	Yes	Yes
PHP support	Yes	Yes	Yes	Yes (unofficial)
Third-party runtimes	No	Yes	Yes	Yes
Complexity of porting existing web application	High (Storage, Java not fully supported)	Low	Medium (storage)	Medium (storage)
Free plan limits	1 instance, 128MB RAM, 5GB storage, 1GB NoSQL database, no relational database	3 instances, 512MB RAM each, 1GB storage each, any database sharing the storage capacity	N/A	1 instance, 512MB RAM, available storage depending on the selected storage service, relational database (10k rows for PostgreSQL)
Low-cost plan price	No plans, charged on per-resource basis, relational database 10.8 USD per month (500MB)	No low-cost plan, charged on per-resource basis	Pivotal: 10.8 USD Anynines: 2.5 EUR	No plans, charged on per-resource basis
Low-cost plan limits	Relational database 500MB	N/A	Pivotal: 1 instance, 512MB RAM, 1GB storage, 5-20MB relational database (MySQL-PostgreSQL); Anynines: 64MB RAM, 10MB storage for relational database	N/A
Private installation complexity	AppScale: Low (VM images and scripts prepared)	Low (VM images and scripts prepared)	High (multi-step build process)	Dokku: Medium (several steps)
Multi-tenant private installation	AppScale: Yes	Yes	Yes	Dokku: No
Local (desktop) micro-cloud deployment complexity	AppScale: Low (VM image) GAE: Low (SDK)	Low (VM image)	High (multi-step build process)	Medium (several steps)

business-related projects here.

5.2 Key Concepts

The key concepts that users (students) should learn when developing for clouds are briefly covered by `12factor.net` web site. The emphasis in courses should be given to the way of working with data storage. The difficulties in porting applications are often caused by an ephemeral filesystem. The code has to be modified in order to use a "storage as a service" solution: relational database, NoSQL database or some kind of cloud file storage.

5.3 Container-based Virtualization

There are many features that are common to all open-source PaaS clouds. They always use some kind of isolation among runtime environments sharing the same machine. While IaaS clouds are usually built on top of hardware virtualization, PaaS clouds prefer lightweight form of operating system virtualization based on containers (though several vendors use different terms for this type of virtualization). Containers provide isolated user-space environments (sharing the same o.s. kernel) inside a real operating system. Each container's resources may be managed by setting desired quotas in order to avoid overloading and to limit users according to the terms of use. Low overheads and suitability for use inside hardware-virtualized machine are main advantages of containers. Figure 2 shows a common PaaS deployment scenario. Individual nodes are managed by PaaS cloud provider and are usually based on hardware-virtualized VMs (e.g. provided by IaaS cloud) though real hardware is another option. When application is deployed to PaaS cloud, required number of containers is set up according to desired factor of horizontal scaling. Containers may share the same node and also be spread across multiple nodes. Application instances are run inside these containers while some PaaS clouds share the same approach to run additional services (e.g. relational database). Containers allow scaling application horizontally by increasing or decreasing the number of instances (containers). They may also support migration of containers between two nodes allowing to balance load among several nodes and to guarantee high availability.

5.4 Trends

Opening of PaaS cloud platforms to many runtime environments and services is an obvious trend of recent years. For example, Cloud Foundry v1 supported only predefined runtimes while Cloud Foundry v2 allows user to deploy his or her application along with required third-party or custom runtime environment. Thus the runtime environment is always encapsulated into a package. The way this encap-

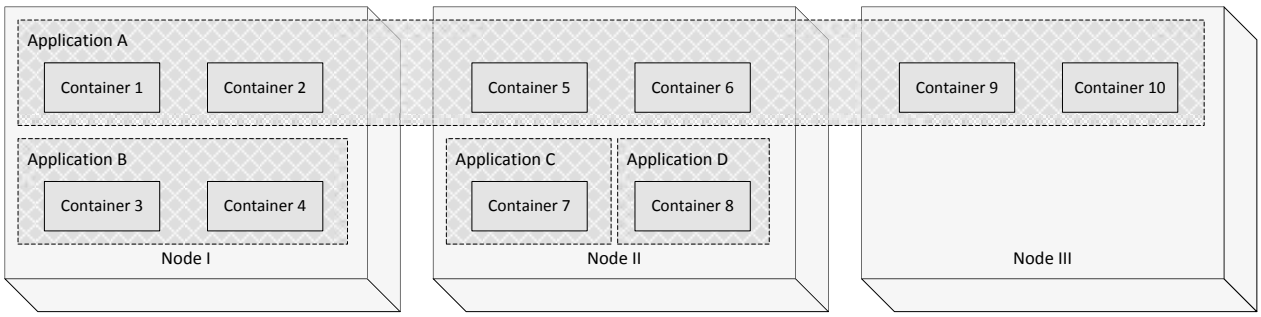

Figure 2: Common PaaS deployment scenario

sulation is done depends on particular PaaS cloud; CloudFoundy v2 and Heroku use *Buildpacks* (originally developed for Heroku) while OpenShift has *Cartridges*. There is also an effort of *Cloud Application Management for Platforms (CAMP)* group in OASIS[8] to standardize the packaging format and API.

The next trend in clouds is *convergence*. The differences among cloud providers of various types (IaaS, PaaS, SaaS) are getting smaller. Many providers offer more types of cloud services, e.g. IaaS and PaaS, while others prepare installation guides helping users with installation of PaaS on provider's IaaS and some PaaS clouds are not far from SaaS as they allow to install existing applications (like WordPress) by clicking in the web management UI. DigitalOcean (IaaS provider) supports this "shift" from IaaS to SaaS by providing Dokku VM images and Dokku is ready to host an existing application using Buildpack, e.g. `heroku-buildpack-wordpress`. Thus the convergence of cloud types and smoother shift from one type of cloud to another is expected in next years.

6. CONCLUSIONS

This paper presents an overview of PaaS cloud platforms; some of them are fully open-sourced (Cloud Foundry and OpenShift) while the others are partially open-sourced having a community-developed parts in order to provide a fully functional open-source alternative (Google App Engine/AppScale and Heroku/Dokku). Several aspects such as availability of a free plan in public cloud, complexity of porting-related issues or complexity of private cloud installation are important for education and were taken into account. According to the observed parameters, OpenShift is a promising PaaS cloud for education, supporting many programming languages and services, offering reasonably powerful instances in public cloud for free and providing straightforward installation methods. In this article, we did not cover the issue of custom runtime environment encapsulation because we assumed the use of already encapsulated ones. However, the analysis of the encapsulation process complexity (i.e. creating custom Buildpacks or Catridges) should be carried out as it is important for applications that require a special runtime environment. Although choosing the right PaaS cloud is important, the current trend of opening PaaS platforms for different runtime environments and technologies shows that it will be possible to switch easily between different providers and platforms in the future. Convergence of different types of clouds (IaaS, PaaS and SaaS) also promises a comfortable transition among these types, e.g. from SaaS to PaaS when customization of an application is necessary.

[8] `https://www.oasis-open.org/committees/camp`

7. REFERENCES

[1] W. Ali. Installing cloud foundry on Vagrant using Bosh-Lite on Windows with Cygwin, January 2014. http://aliwahaj.blogspot.de/2014/01/installing-cloud-foundry-on-vagrant.html.

[2] A. Bodnárová, M. Hátaš, K. Olševičová, V. Soběslav, and J. Štefan. Virtual and virtualization technologies in computer networks education. In *Proceedings of the European Conference of Systems, and European Conference of Circuits Technology and Devices, and European Conference of Communications, and European Conference on Computer Science*, ECS'10/ECCTD'10/ECCOM'10/ECCS'10, pages 281–285, Stevens Point, Wisconsin, USA, 2010. World Scientific and Engineering Academy and Society (WSEAS).

[3] C. Bunch, N. Chohan, C. Krintz, J. Chohan, J. Kupferman, P. Lakhina, Y. Li, and Y. Nomura. An evaluation of distributed datastores using the AppScale cloud platform. In *Proceedings of the 2010 IEEE 3rd International Conference on Cloud Computing*, CLOUD '10, pages 305–312, Washington, DC, USA, 2010. IEEE Computer Society.

[4] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, T. Chandra, A. Fikes, and R. E. Gruber. Bigtable: A distributed storage system for structured data. *ACM Trans. Comput. Syst.*, 26(2):4:1–4:26, June 2008.

[5] K. Chodorow. *MongoDB: the definitive guide*. O'Reilly Media, Inc., 2013.

[6] N. Chohan, C. Bunch, S. Pang, C. Krintz, N. Mostafa, S. Soman, and R. Wolski. Appscale: Scalable and open AppEngine application development and deployment. In D. Avresky, M. Diaz, A. Bode, B. Ciciani, and E. Dekel, editors, *Cloud Computing*, volume 34 of *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and*

Telecommunications Engineering, pages 57–70. Springer Berlin Heidelberg, 2010.

[7] P. T. Endo, G. E. Gonçalves, J. Kelner, and D. Sadok. A survey on open-source cloud computing solutions. In *VIII Workshop em Clouds, Grids e Aplicações*, pages 3–16, 2010.

[8] L. George. *HBase: the definitive guide*. O'Reilly Media, Inc., 2011.

[9] Google Inc. Cloud SQL frequently asked questions, March 2014. https://developers.google.com/cloud-sql/faq.

[10] P. Hunt, M. Konar, F. P. Junqueira, and B. Reed. Zookeeper: Wait-free coordination for internet-scale systems. In *USENIX Annual Technical Conference*, volume 8, page 9, 2010.

[11] A. Lakshman and P. Malik. Cassandra: a decentralized structured storage system. *ACM SIGOPS Operating Systems Review*, 44(2):35–40, 2010.

[12] LangPop. Programming language popularity, October 2013. http://langpop.com/.

[13] Z. Li, H. Zhang, L. O'brien, R. Cai, and S. Flint. On evaluating commercial cloud services: A systematic review. *J. Syst. Softw.*, 86(9):2371–2393, Sept. 2013.

[14] J. Loope. *Managing Infrastructure with Puppet*. O'Reilly Media, Inc., 2011.

[15] D. Petcu and M. Rak. Open-source cloudware support for the portability of applications using cloud infrastructure services. In Z. Mahmood, editor, *Cloud Computing*, Computer Communications and Networks, pages 323–341. Springer London, 2013.

[16] B. P. Rimal, E. Choi, and I. Lumb. A taxonomy and survey of cloud computing systems. In *Proceedings of the 2009 Fifth International Joint Conference on INC, IMS and IDC*, NCM '09, pages 44–51, Washington, DC, USA, 2009. IEEE Computer Society.

[17] N. Sultan. Cloud computing for education: A new dawn? *International Journal of Information Management*, 30(2):109–116, Apr. 2010.

[18] L. M. Vaquero. Educloud: PaaS versus IaaS cloud usage for an advanced computer science course. *IEEE Trans. on Educ.*, 54(4):590–598, Nov. 2011.