bwCloud: cross-site server virtualization

O. V. Dulov

Steinbuch Centre for Computing, Karlsruhe Institute of Technology
Hermann-von-Helmholtz-Platz 1, Building 441 / 236, 76344 Eggenstein-Leopoldshafen, Germany

E-mail: oleg.dulov@kit.edu

The paper provides a project status, system architecture, and future work for bwCloud. The purpose of the bwCloud project is to build a prototype for cross-site Infrastructure as a Service (IaaS) system, based on the set of universities in Baden-Württemberg (Germany). This differentiates from the typical IaaS systems, where sites located into one data center or into one service provider's organization. The federative approach is the core concept for the project: it starts from the system architecture, follows through the resource sharing and user management for the scientific community in the region. The concept for production-ready cross-site IaaS is another output from the project. This concept can be used by any other universities or research centers to connect to bwCloud or setup their own cross-site IaaS platform. There are some design ideas introduced and some technological stacks analyzed.

Keywords: University Cloud, Cloud computing, Virtualization, Infrastructure as a Service, cross-site IaaS, multi-region OpenStack.

1. Introduction

A Cloud Computing is the powerful paradigm in IT-area and represents the shift in providing IT as a service. In general, the main idea behind the Cloud Computing is to abstract the hardware infrastructure for users. Cloud platforms provide their resources over the Internet or Intranet with the self-service approach, where a user can interact with the infrastructure or preconfigured hosts, based on his specific requirements. Computer Centers are using the Cloud Computing, providing their services based on different models.

Traditionally, there are clear separated service models to deliver Cloud-based services [NIST]: Infrastructure as a Service (IaaS), Platform as a Service (PaaS), Software as a Service (SaaS). Sometimes these concepts are considering as the layered structure: PaaS extends IaaS and can provide SaaS. In general, it is not necessary and all three concepts can base on completely different platforms. Currently, there are more than sixty extensions [Sharma] in an as-a-Service family, such as Database as a Service (DBaaS), or HPC as a Service (HPCaaS).

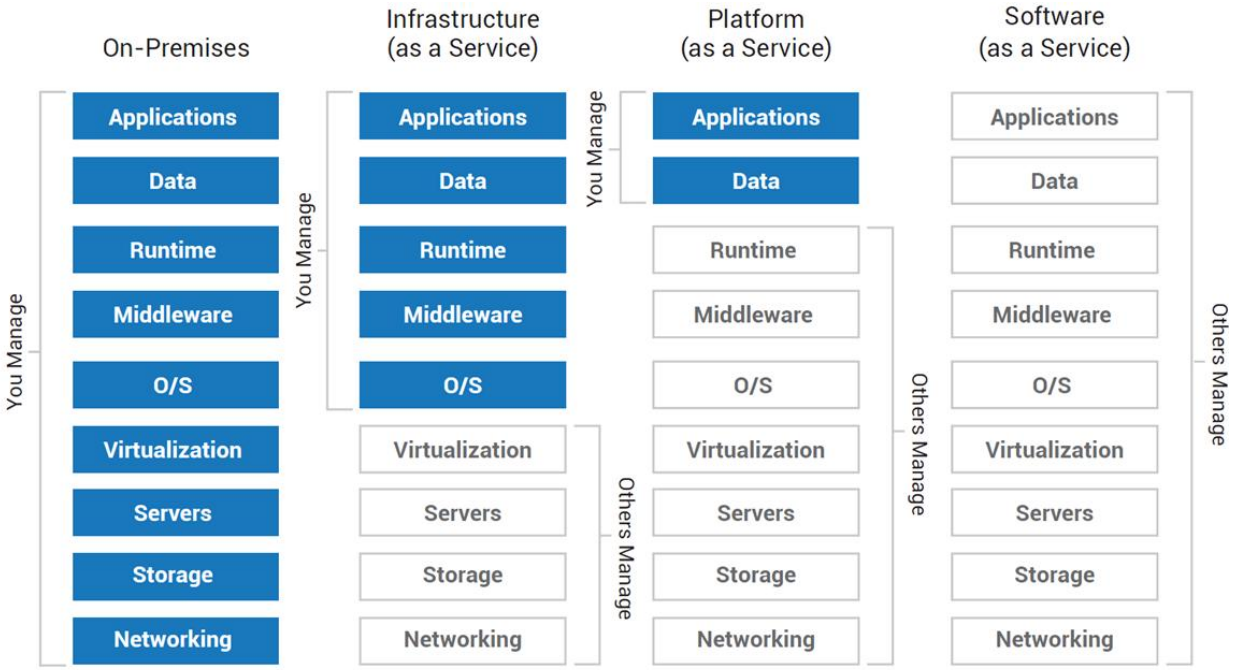

Figure 1. Three basic Cloud Service Models

IaaS gives the most level of flexibility for users, by providing highly scalable resources that can be adjusted on-demand. This makes IaaS well-suited for temporary, experimental or change unexpectedly workloads. Other characteristics of IaaS environments include the automation of administrative tasks, dynamic scaling, desktop virtualization and policy-based services. The IaaS functionality can include framework for installing additional packages during the virtual resources maintenance. Such an extended IaaS sometimes called IaaS+ concept, because it is still not quite PaaS, but already not the "classical" IaaS.

Users should monitor their virtual resources closely to avoid being charged for unauthorized services. Because IaaS providers own the infrastructure, systems management and monitoring may become more difficult for the users, and if an IaaS provider experiences downtime, users' workloads may be affected.

There are four Cloud deployment models[NIST] are considered: private, public, community and hybrid. Private cloud is a cloud infrastructure operated solely for a single organization, whether managed internally or by a third-party, and hosted either internally or externally. Often, the access to the private cloud is through an intranet. Very often, the private cloud called the Enterprise Cloud, but this is not necessarily true. A cloud is called a "public cloud" when the services are rendered over a network that is open for public use, the Internet.

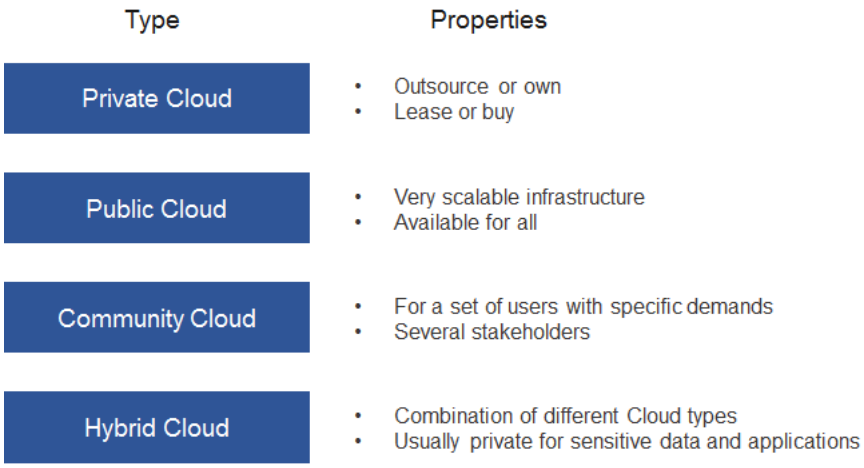

Figure 2. Cloud deployment Models

Community cloud shares infrastructure between several organizations from a specific community with common concerns (security, compliance, jurisdiction, etc.), whether managed internally or by a third-party, and either hosted internally or externally. Hybrid cloud is a composition of two or more clouds (private, community or public) that remain distinct entities but are bound together, offering the benefits of multiple deployment models.

Cloud architecture is the systems architecture of the software systems involved in the delivery of cloud computing, typically involves multiple cloud components communicating with each other over a loose coupling mechanism such as a messaging queue. Elastic provision implies intelligence in the use of tight or loose coupling as applied to mechanisms such as these and others.

The universities are providing their IT services for the students, researches and employees. As any service provider need to optimize the infrastructure costs and improve the scalability, usability and flexibility for the users. Usually, universities are installing the IaaS Platform based on their own computing resources, under one site, as a private Cloud. The local system administration team usually provides the service to users from the same university.

The idea behind bwCloud Project is to share IaaS resources between a set of universities in Baden-Württemberg: These requirements are pointing to the cross-site IaaS platform with community Cloud deployment model. The project started in December 2014 for two years with the following two aims: prototypical implementation and concept development of a "university-Cloud". The project partners are the Universities in Mannheim[1], Freiburg[2], Ulm[3], Stuttgart[4], Karlsruhe Institute of Technology[5] and BelWü[6]. Additional information about the bwCloud project is on the project's website: http://www.bw-cloud.org.

[1] https://www.uni-mannheim.de
[2] https://www.uni-freiburg.de
[3] https://www.uni-ulm.de
[4] http://www.uni-stuttgart.de
[5] https://www.kit.edu
[6] https://www.belwue.de

2. Cross-site IaaS

The main challenge for constructing IaaS infrastructure – to build the site: computing cluster with a Cloud platform on it and provide the functionality for users. In a case of bwCloud, there are more than one site and users are coming from a set of universities. Hence, the deployment model for the project is Community Cloud, where under user community considered all projects and people from the institutes into the region. There are two logical ways to construct cross-site IaaS: by setting up hybrid and non-hybrid systems.

The hybrid system differentiates between the Cloud (virtualization, containerization) platforms on every site and integrated cloud management platform (CMP) for the cross-site interconnection and management. The Cloud platform and the CMP can be completely different software stacks. In such an architecture, the Cloud management platform is communicating with the Cloud or virtual platforms and assigning them management tasks for virtual resources. The functionality split between the following:

1. Cloud management platform is independent from virtualization environment and not responsible for actual maintenance of virtual resources.
2. Virtualization platform is not responsible for managing the cloud user's rights, constructing workflows, templating and so on.

There are some potential problems with IaaS architecture based on CMP, such as incompatibility between virtual resources formats through sites and not sufficient quality and limitations of cloud management software for today. This technology is more promising for use cases, where for example, the virtualization or Cloud platform into the site cannot be changed because of one or other reason but should be included into the bwCloud infrastructure.

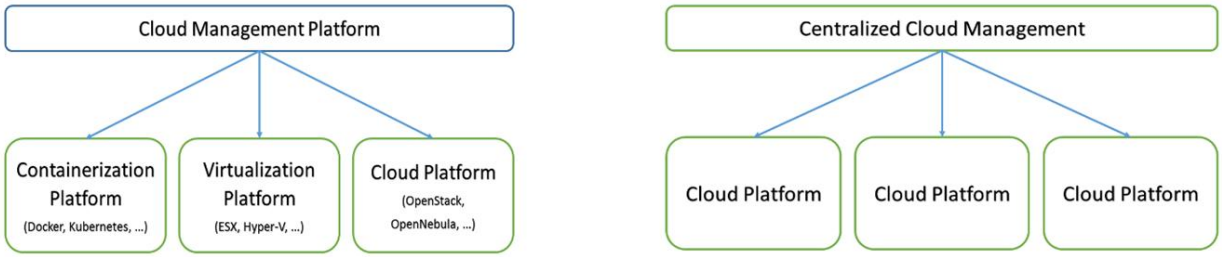

Figure 3. Hybrid and non-hybrid cloud architecture

In the case of non-hybrid architecture, every site is using the same software platform. Interconnection and management provided by the same software and somehow distributed among different site locations. There are some components for centralized management functionality.

As we can see, there are not big differences in general architectural sketches: there are two subsystems: centralized cloud management and distributed cloud platform. In one case the role of centralized management is playing by special tool – CMP, in another the same for every site part of the Cloud platform software. The hybrid model offers more flexibility to choose the Cloud platform (it can be a virtualization or containerization platform) and couple the managerial tasks, user workflows and management, but the price for this can be higher complexity and (as for today) low-quality software.

There is a list of software platforms, which can be used for constructing IaaS with or without CMP, but currently de-facto standard is the OpenStack Project[7]. OpenStack project has many activities inside, which are separated as independent projects[8] and can be grouped into the following types:

1. Core - official projects with full access to OpenStack brand and assets.
2. Incubated or optional - projects on an official track to become core projects; partial access to OpenStack brand and assets.
3. Library - projects are directly or indirectly consumed by Core projects.
4. Gating - gating projects are used as part of the continuous integration gate mechanism that protects the core projects branches from regressive changes.
5. Supporting - additional projects that are deemed necessary in producing the other official projects.
6. Related - unofficial projects with no rights to use OpenStack brand and assets or project resources.

3. The system architecture

In general, the Cloud system architecture can be seen as three-layered structure: a physical layer, a virtual resources layer, and a service layer. Between the physical and the virtual resources, it can be additional allocation sublayer to physical and virtual resources interconnection. There are a set of software stacks and technologies are available for every layer.

The physical layer for bwCloud is not determined - every site can use their own hardware system and networking components. The service layer fixed by the IaaS, hence – only the virtual resources will be provided, without any platform or software-as-a-Service components. The allocation and the virtual layers are the main challenges for the cross-site virtualization.

During the evaluation phase of the bwCloud project, two architectures were setup: non-hybrid based on OpenStack and hybrid with CMP ManageIQ[9]. The software containerisation technologies are often outside of IaaS context and were excluded from evaluation for constructing IaaS, but (as we will see later) were considered as the underlined technology to deploy the IaaS itself.

ManageIQ project integrates different types of middleware: cloud, virtualization and container platforms. The interconnection between CMP and cloud/virtualization platform made by the user with an administrative role. Current state for this project does not allow the migration or conversion the virtual resources between different underlying formats, for example, VMWare-based VM cannot be easily migrated into OpenStack and another way around.

To avoid the virtual infrastructure complexity, bwCloud decided to construct the system without mixing the virtualization, containerization, and hybrid technologies and concentrate on Cloud platform based on OpenStack multi-region setup. There are six core projects currently into OpenStack: Nova for computing tasks, Neutron for networking, Cinder as block storage service, Swift – object storage, Glance as image service, Keystone as an identity service.

All OpenStack core projects, except Swift, are included into the bwCloud prototype to build compute cluster without object storage. The supported optional OpenStack Telemetry[10] services: Ceilometer, Gnocchi and Aodh and the Horizon Dashboard added with some customizations to provide the Dashboard and the system Monitoring functionality.

[7] http://www.openstack.org/
[8] https://wiki.openstack.org/wiki/ProjectTypes
[9] http://manageiq.org/
[10] https://wiki.openstack.org/wiki/Telemetry

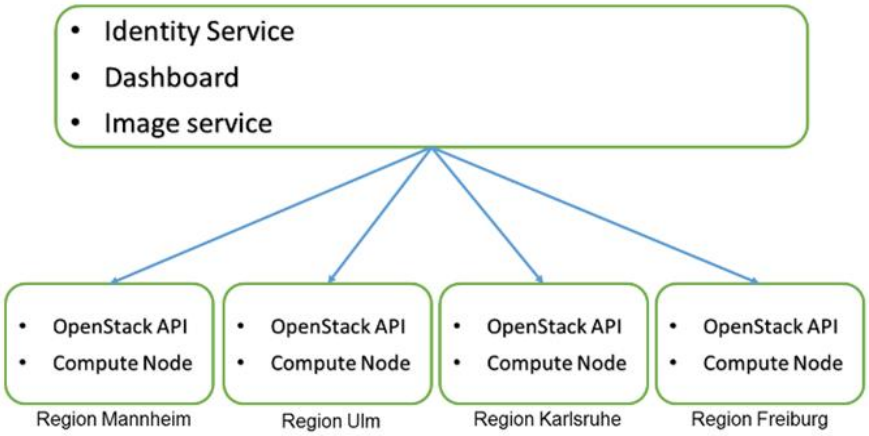

Figure 4. bwCloud OpenStack multi-region setup

All participating sites are representing themselves as regions and provides the OpenStack application programming interface (API) for inter-service communication. As we can see in Figure 3, the services are separated into two groups: installed on every site and some central services. Thus, Nova, Neutron and Cinder and their APIs are installed under every site. Two other core services (Glance, Keystone) and optional Horizon and Monitoring stack are the central bwCloud components.

Typically, the OpenStack installation based on commodity hardware: relatively inexpensive, widely available devices. The bwCloud do not restrict sites for hardware configuration: every site can choose which servers and networking components to use. It can be commodity server or blade center with attached storage area network (SAN) cluster. The sustainable APIs are getting the possibility for cross-site communication inside distributed environment. Hopefully, the OpenStack API do not change much between releases, hence it is possible to have the situation, where different sites have an even different version of OpenStack services.

Practically speaking, the OpenStack multi-region setup is not different from one region setup: but allow switching between regions. OpenStack central components cannot provide functionality to mix resources between sites – every site is independent but accessible with API. This situation makes difficult to migrate virtual resources between sites. Such a migration task can be interesting for migration of one virtual machine or the whole site (e.g. before site downtime).

In order to increase the site reliability, all Compute Nodes based on the Ceph[11] distributed object store. Ceph is open source and freely-available distributed object store and file system designed to provide good performance, reliability, and scalability.

During the bwCloud project, some extension to Horizon-based Dashboard implemented to getting the possibility for the user to make migration between two sites, getting some monitoring information and some others addons. The migration between the regions is coupled with the user quota management: the resource policy for every user.

4. User management and use cases

Before the user can start with bwCloud, he/she should register himself. User registration is done by connecting to bwIDM[12], - federated identity management platform for universities in Baden-Württemberg. The core of the bwIDM system is the Identity provider based on shibboleth Single Sign-On. The bwCloud in this structure is the shibboleth service provider with permission to communicate with Identity provider and ask the user attributes.

[11] http://ceph.com/
[12] https://www.bwidm.de/

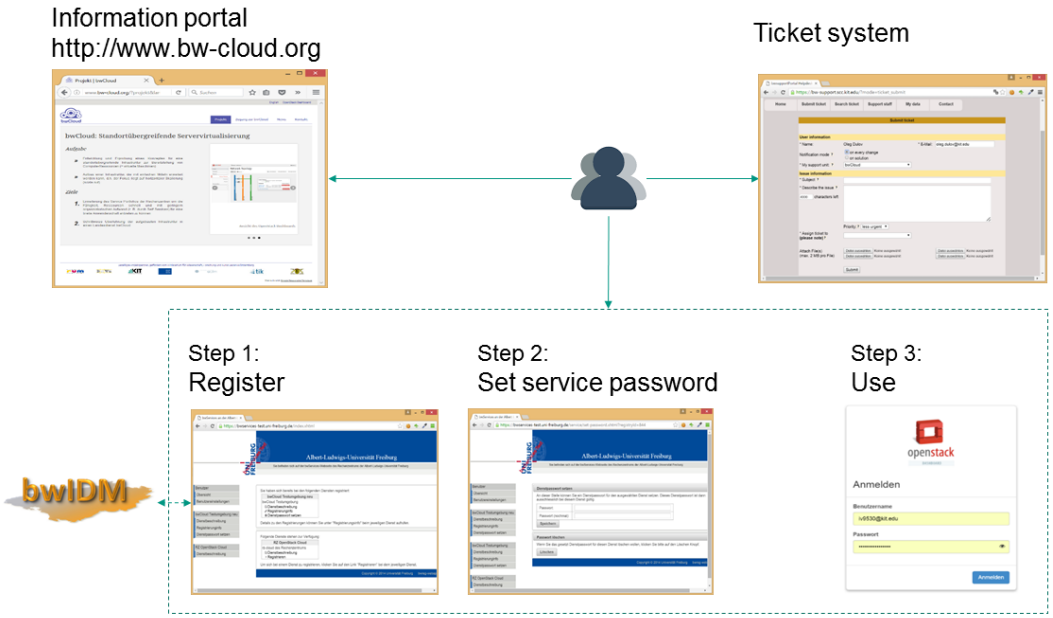

Figure 5. bwCloud user capabilities

After registration and approval, the user should setup password for bwCloud. The registration portal cannot store user credentials from the identity provider but provides the possibility set the service password into OpenStack Keystone. In this setup, any participant of bwIDM can register himself and have login credentials for bwCloud. If user does not want to use system anymore, he can deregister himself. Currently, no automatic deregistration procedure is implemented, but it can be by updating the user status from Identity Provider.

The problem to use special registration portal is the complexity not only to register the user but also deregistration process: the portal itself have no information about how the system is used by the user, only which shibboleth attributes are coming for him. The registration process can be simplified by providing the shibboleth authorization directly by the Horizon. This option is also considered, but the general workflow is approved.

The ticket system is the major part of the user support process. Currently, the bw-support portal[13] used for this purpose, based on xgus[14] platform. The xgus framework developing the Karlsruhe Institute of Technology and provides the Global Grid User Support ticketing system for Large Hadron Collider Experiment (LHC) and some others helpdesk portals.

The accounting for the user currently not provided, but only evaluated and prototypically implemented. The base for accounting metrics are the data from Nova database. This data is collected and the price function applied for used resources. The price model is simple and includes the weighted sum of the virtual CPU, Disk, and RAM usage. The price estimation calculated during the instance creation based on the flavor. In OpenStack, a flavor defines the compute, memory, and storage capacity of a virtual server, also known as an instance.

The recommended way to implement the accounting to setup the CloudKitty[15] rating-as-a-Service project. This can be combined with the telemetry measurements and provides the dashboard for customization of weights for price function and set the attributes which to include into the price model.

There are three use cases identified for the project: "Student-VM", "Institut-/Scientist-VM", "Site operation/administration". Students require the virtual resources during their study process. Here the

[13] https://bw-support.scc.kit.edu/index.php?mode=index
[14] http://xgus.scc.kit.edu/
[15] https://wiki.openstack.org/wiki/CloudKitty

resources are minimal and statistics for billing collected for the appropriate project manager or scientific employee. A scientist needs the resources for their scientific projects with more resources and storage requirements and API access.

Defined use cases are very general and do not specify if, for example, if the GPU-hardware supported or not. Providing the Cloud functionality for the special hardware is out of the project scope. It can be done by one or more sites in the future if the appropriate software (connector) available. Near the same situation is for supporting software containers: it can be done inside the Virtual Machine, but not as a part of bwCloud infrastructure.

5. Operation

Every bwCloud site is operated by a local administrator with requirements to control, allocate the resources, implementing resource migration between sites. There are many operational aspects in case of Cloud computing platform: one group of them is the administration of the Cloud infrastructure and another is the Cloud user operational tasks. In the same time, there are many software stacks to archive these goals: some of them can combine these two groups of interest, others provide only for one group.

Historically, the system administration can be done by scripting the tasks, by using the configuration management tools, by using the software containers or a combination of them. In a case of the Cloud provisioning, there is one more option: to use special provisioning system, which can deploy and manage the maintenance specifically for the Cloud platform. For example, OpenStack has the project TripleO[16] to solve this kind of questions.

The project evaluates the Ansible[17] configuration management tool for provisioning with playbooks, written for bwCloud. The RDO packages deployed into the Centos-based Linux cluster with the Ceph storage backend. The OpenStack Services are using Ceph cluster as a storage backend. Currently, there are no needs to use the special bare-metal provisioning (or Metal as a Service) projects like TripleO, but it can be used as well.

There is an alternative way do not use the RDO packages, but the source code from the official repositories. This also evaluated during the bwCloud, indirectly, by using the Docker-based containers from OpenStack Kolla[18] project. Such a setup is relatively uncommon in our days, but the technology is very promising for dynamically allocating the resources for Cloud Platform with the possibility to separate different environments for development and production.

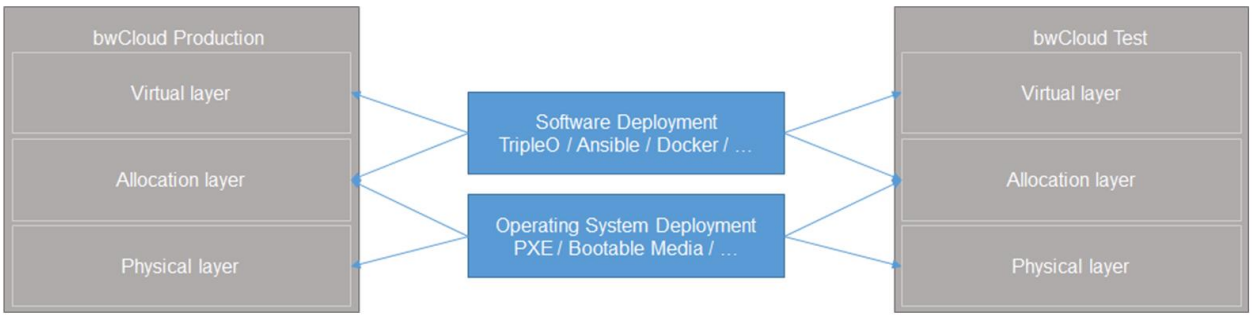

Figure 6. Systems for Deployment and Release management

If consider the release deployment cycle, the TripleO, Ansible, Docker or by using the systems like the Foreman[19], provides the opportunity for transparent migration the system from test or devel-

[16] https://wiki.openstack.org/wiki/TripleO
[17] https://www.ansible.com/
[18] https://wiki.openstack.org/wiki/Kolla
[19] https://theforeman.org/

opment environment into the production environment. Because of the specific of OpenStack software in our days, not all tools are providing the same level of complexity for this tasks. In most cases, it depends on the method how the cluster was installed. The current OpenStack architecture in some cases has dependent services, what is not good during the update process. In the case of Docker containers, this question disappears, because from the beginning all services are separated into own containers.

The Cloud users actually can use any provisioning and deployment tools, which they want inside the Virtual Machine. Currently, one operational interface should be provided from the Cloud Platform to the user: the virtual resource Monitoring. In general, the Cloud monitoring is a big challenge, because of complexity and integration into the Cloud Platform. There are many different types of events are flowing into the monitoring system, and API access must be provided from the monitoring server not only for the operational team but also for the Cloud users.

Usually, the availability and performance monitoring are considered and monitoring system consists of minimum three subsystems: checks handling, notifications and alarming, and dashboard. If the Monitoring system considering the system under monitoring as a "Black Box", we named it as a "black box", else the "white box" monitoring system.

The bwCloud benchmark is a part of black-box monitoring together with the availability checks. This done centrally, based on the OpenStack Rally[20] project, where the internal functionality for every site benchmarked and the results are regularly published as reports for every bwCloud region. Availability checks are not centralized and done by site's local monitoring system (e.g. Nagios, Icinga, Zabbix etc.).

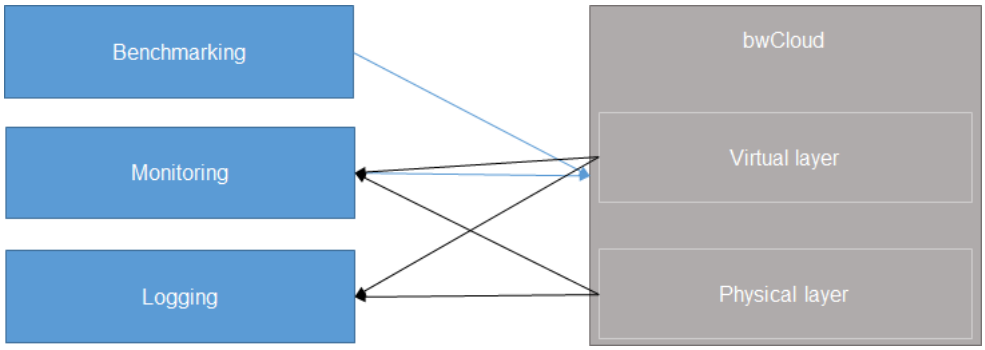

Figure 7.The bwCloud monitoring, logging and benchmarking

There are three white box monitoring models are evaluated. The first centralize dashboard and leave performance checks and notification under the site. System statistics are collected by the collectd[21] tool and stored into influxdb[22] time-series database. Notification is done by collectd or any site monitoring platform (e.g. Nagios, Icing, Zabbix, etc.). Grafana[23]- based dashboard connects to the InfluxDB and publishes the statistics for every site. The Cloud user can be connected into appropriate scripted Grafana dashboard for his virtual resources. In the case of Docker-based deployment, existing software containers monitoring tools can be used.

The second model centralized not only the monitoring dashboard but also the notification. It assumes that statistics should be provided to the central monitoring message queue and handled by the central monitoring server. The concept includes interconnection between collectd and Sensu[24] monitoring project under site hosts and setup central Sensu server. Sensu components can be combined

[20] https://wiki.openstack.org/wiki/Rally
[21] https://collectd.org/
[22] https://www.influxdata.com/
[23] http://grafana.org/
[24] https://sensuapp.org/

with open-source automation platform StackStorm[25] for automation purposes. Unfortunately, the Sensu handler for open source version is not working stable under high load and this scenario should be combined with other monitoring solutions, which makes not much sense.
The previous two models do provide the Cloud monitoring under the system level and without OpenStack monitoring projects. The third is about how to construct the OpenStack-based Monitoring as a service for every Cloud user. Monitoring-as-a-service solutions, such as Amazon's CloudWatch[26], are not open-source. There is Monasca[27] project, which addressed the open-source solution for OpenStack and can be integrated into Horizon. The metrics can be accessed also by using the OpenStack Telemetry.
Additionally, the dashboard should be integrated for the user virtual resources. As the prototype, the Horizon dashboard was implemented which generate the list of virtual machines pro user and create the link to the Grafana server. The dashboards are generated under Grafana-side for the appropriated virtual resource id. These dashboards can be not only linked, but also integrated into Horizon directly (with HTML iframe tag or directly into Horizon by python or JavaScript coding).
The question about centralizing logging for the bwCloud split into two models: central log server is located on every site and the log server is located centrally for all sites. On both cases, the logging messages are processed by ELK Stack: logs are proceed by Logstash[28] and stored into Elasticsearch[29] Database. Kibana[30] does dashboard with the search option. The same monitoring Grafana dashboard connects to Elasticsearch and publish logs information together with performance monitoring data.
There are many other operational questions should be answered: bwCloud security, release deployment process, checking the VM Image functionality before publishing, delisting the user resources, and some others. Unfortunately, the OpenStack architecture is changed very often between their releases and currently very difficult to note the way to unify all these activities. The very good constructed monitoring for the Cloud system can solve not only a big part of this but also the user accounting questions.

6. Conclusions and future work

The bwCloud project archives the goals to create a prototype and describes the concept for "university-Cloud" for a set of universities in the region. In this article, the current state is presented and the major architectural components are noted. Of course, not all project activities are described, for example, complex and time-consuming evaluation process in order to make a decision which technology to use. During the process some valuable companies (RedHat, IBM, Mirantis etc.) was consulted, different webinars participated.

In particular, the hard questions addressed to the hybrid platforms: it is clear the benefit in general - the level of flexibility for sites will be much higher, but choosing the stable technology with clear advantages against the OpenStack is the good question. Another point is commercial or open source free and non-free software and product support. Because of mixing different types of virtualization platform, the hybrid Cloud functionality on OpenStack can be an overhead.

One more hybrid aspect is the underlying hardware infrastructure, which can be hetero- or homogeneous. Currently, the hybrid system configuration is considered, what means in simple words: "every site has own decision about the hardware". But this does mean it can have different views, like specialized hardware for one site, and the VM with requirements for this hardware type can be allo-

[25] https://stackstorm.com/
[26] https://aws.amazon.com/cloudwatch/?nc1=h_ls
[27] https://wiki.openstack.org/wiki/Monasca
[28] https://www.elastic.co/products/logstash
[29] https://www.elastic.co/products/elasticsearch
[30] https://www.elastic.co/products/kibana

cated to this site. In order to unify and to avoid the complexity, the homogeneous systems can be considered, where every site has the same hardware systems.

While the bwCloud project is successful, the system will be improved by the next project, bwCloud SCOPE, which will start from 2017 for three years. During this time, the bwCloud prototype should be transformed into production platform. During this transformation, it will be some changes in the architecture, for example adding the Object storage, long-term user data archiving services, and/or data analytics use cases. The major point for bwCloud SCOPE project will be operation improvement for providing production-ready service for the bwCloud users.

References

National Institute of Standards and Technology, U.S (2011). The NIST Definition of Cloud Computing. [Electronic resource] (Last accessed on October 10, 2016) http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf

Sugam Sharma (2015). Evolution of as-a-Service Era in Cloud. [Electronic resource] (Last accessed on October 10, 2016) https://www.researchgate.net/publication/279784427_Evolution_of_as-a-Service_Era_in_Cloud