Grant Agreement Number: 257528

KHRESMOI

www.khresmoi.eu

Prototype and evaluation of the "Full Cloud Infrastructure"

Deliverable number	*D6.4.3*
Dissemination level	*Public*
Delivery date	*August 2013*
Status	*Final*
Author(s)	*Ivan Martinez Rodriguez, Miguel Angel Tinte Garcia*

This project is supported by the European Commission under the Information and Communication Technologies (ICT) Theme of the 7th Framework Programme for Research and Technological Development.

1 Executive summary

This document aims to report on the final assessment of the Khresmoi *Full Cloud Infrastructure*, which is directly aligned with previous work described in Deliverable 6.4.2, in which the *Early Cloud Prototype* was presented.

This final stage of cloud development has been implemented over a new hardware resources environment. The new hardware resources have been deployed in CUNI labs and ATOS has been mainly involved in the installation of the software in charge of managing the cloud (Open Nebula) and the hypervisor to virtualise the resources, as well as the deployment of the virtual machines (VMs). The main characteristic of the new cloud hardware resources is the acquisition of nine powerful servers that will be managed as different nodes to allow virtualisation of resources into different virtual machines to fulfil the requirements of every Khresmoi component.

Besides the description of the Full Cloud architecture, this document includes important chapters related to evaluation, as a final step of the deployment. The result of this evaluation process relies on obtaining useful quantitative metrics that indicate the architecture adequacy and performance. Therefore, this document describes the assessment performed over the final stage of the cloud architecture deployment with the main goal to ensure that it satisfies to the Khresmoi requirements in terms of performance and scalability.

Table of Contents

2 List of abbreviations

ART	Average Response Time
CPU	Central Processing Unit
DoW	Description of Work
ECP	Early Cloud Prototype
FCP	Full Cloud Prototype
JSON	JavaScript Object Notation
KMI	Khresmoi Mimir Interface
QMS	Query Mapping Service
SCA	Service Component Architecture
SOA	Service Oriented Architecture
SOC	Service Oriented Cloud
SSH	Secure SHell
URI	Uniform Resource Identifier
URL	Universal Resource Locator
VLAN	Virtual Local Area Network
VM	Virtual Machine

Table 1. Abbreviations and acronyms.

3 List of figures

4 List of tables

5 Introduction

5.1 Introductory Explanation of the Deliverable

The main objective of this document is to describe the Full Cloud prototype, in terms of hardware and software resources, as well as architecture structure. This prototype represents the final task of cloud deployment, as planned in T6.3 System scaling, which started with the specification of the Early Cloud Prototype and finishes with tasks T6.3.6 and T6.3.7, also described in this deliverable.

This final stage of development must be also tested exhaustively in order to check its reliability and scalability before being released as a final version. This document describes this process and the results associated to cloud metrics are included in Chapter 10 Results report.

5.2 Purpose and Audience

5.2.1 Purpose

The purpose of this deliverable is to present the final state and assessment process of the Cloud deployment that will finally be hosting the Khresmoi platform. The assessment will be performed based on cloud metrics described in D6.4.2 [1] and consists of a quantitative approach to system scalability, reliability and other features the system must take into account in order to provide good performance.

5.2.2 Audience

This deliverable is relevant to all technical work packages in Khresmoi (WP1-WP9). The target audience includes component providers, users, and any person inside or outside of the Khresmoi project interested in learning about the internal processing of the Khresmoi Cloud Infrastructure.

Specifically, this document concerns all partners developing a functional component within Khresmoi workflows because the Full Cloud prototype is aimed to host every components and all the necessary data related to them.

5.3 Structure of the Document

This document has the same structure as D6.4.2. [1], it follows the same approach, first describing the Cloud prototype and then introducing Coud facets and metrics with the results obtained. Hence, the first part focuses on describing theoretical concepts, such as software and hardware definitions, the second part describes the software election, as well as the architecture model chosen and finally it focuses on evaluation results description in terms of several numeric metrics that are submitted. Concretely, Section 6 focuses on defining the new hardware and software requirements specified for the Full Cloud Prototype. Section 7

describes the Full Cloud environment, mainly the OpenNebula[1] cloud manager framework. Section 8 mentions the evaluation approach followed during the evaluation phase and Section 9 defines the Test Scenarios that will be achieved. Section 10 is dedicated to present the results obtained during the evaluation part, while Section 11 reports on the Facet Linked report analysis. Finally, Section 12 presents the final conclusions concerning all the work performed

[1] http://opennebula.org/

6 Full Cloud Final Requirements

The Full Cloud prototype must provide enough hardware (HD) and software (SW) resources to offer a good performance in terms of CPU, Memory, Processor speed, etc. These requirements have been specified as a result of an information collection process among the partners that has been performed during the past year, when the Early Cloud Prototype was deployed. This period has been devoted to identify new needs or features to add on future iterations. For instance, some of these features are those related to concurrent access to the system in order to avoid bottlenecks identified in the last evaluation process. Table 2 shows the new requirements in terms of HD and SW resources that will be needed for deploying the Khresmoi Full Cloud.

Partner	Component	VMs	RAM	HDD	Cores	Extra storage	Bandwidth	Hypervisor	Monitoring	Software requirements
UNI-DUE	**ezDL Web**	1	4GB	100GB	1	Possible	No special configuration required	Xen	CPU load, free HDD, thread count, DB connections, IO load, NW load	Sun Java 1.7 and Mysql 5.X. Debian Linux preferred
	ezDL Swing	1	4GB	100GB	1	Possible	No special configuration required	Xen	CPU load, free HDD, thread count, DB connections, IO load, NW load	Sun Java 1.7 and Mysql 5.X. Debian Linux preferred
HES	**ParaDISE**	1/2.	30GB	1-5TB	2	Possible	No Special configuration required	Xen	Seems like they like to monitor some parameters	Java + Linux
ONTO TEXT	**BigOWLIM, Dissambiguator (Development and Production enviroments)**	2	15GB min. 30GB max.	600-700GB	8 (4x2)	Highly probable	No special configuration required	Xen	Webserver	Sun Java, Servlet container and Khresmoi integration system. GUI (Xwindows) and they prefer to use Ubuntu rather than Debian
USFD	**GATE/Mimir**	1	8GB	100GB	2	Possible	No special configuration required	Xen	Not required	Sun Java, Tomcat
CUNI	**MOSES**	8	32GB	200 GB by each VM	64 (8x8)	Possible	No special configuration required	Xen	Not required	Debian preferred, Python, PERL, GCC
MUW	**MUW 3D**	more than 4x3.0 GHZ CPUs	74GB	1TB+ and fast file system (SAN)		Easy if SAN	No special configuration required	No preferences	Standard logs	WebApp Server and Matlab components
HON	**Spell Checker, Search Framework**	1	8GB	1.5TB	1		No special configuration required	No preferences	Standard logs	Tomcat, Java 1.7 update 1
	Crawling, Annotation and Indexing	1	16GB	5.5TB	24		No special configuration required	No preferences	Standard logs	Tomcat, Java 1.7 update 1
ATOS	**SCA Components**	8	4GB per VM	100GB	4	Possible	No special configuration required	Xen	Standard logs	Tomcat, Java 1.7, Apache Tuscany
ATOS	**SCA Composites**	2	4GB per VM	100GB	4	Possible	No special configuration required	Xen	Standard logs	Tomcat, Java 1.7 , Apache Tuscany
ATOS	**Cloud Manager**	1	4GB per VM	200 GB	4	Possible	No special configuration required		Application Manager or Nagios required.	Open Nebula, CentOS (Linux distribution)

Table 2. Full Cloud hardware and software requirements

This chapter describes the hardware resources acquired, as well as the planned virtualisation tasks that were applied to it to cover all the requirements in terms of VMs. The main characteristics of the hardware resources are:

- 9 servers:
  - 1x server: 1x Intel 4-core 2,4GHz, 2x 1TB WD RE4 HDD (mirror), 32GB RAM DDR3
  - 8x server: 2x Intel 6-core 2.0GHz, 2x 3TB HDD SATA, 128GB RAM DDR3
  - Six of them have 2x 1TB SATA HDD, 2 of them have 2x 3TB SATA HDD.
- 9 nodes:
  - 1 Management node: Node 0, Open Nebula cloud manager: node0-khresmoi.ms.mff.cuni.cz
  - 8 virtualization nodes: node1..node8
- 20 Virtual Machines to deploy Khresmoi components with datastores, interfaces, etc.

This new environment aims to fulfil the Khresmoi performance requirements, as well as to show good scalability to ensure multi-access availability of the Khresmoi platform. The tests reported in the next chapters focus on assessing these features. Figure 1 shows the final diagram with the Khresmoi Full Cloud hardware and the virtualised resources.

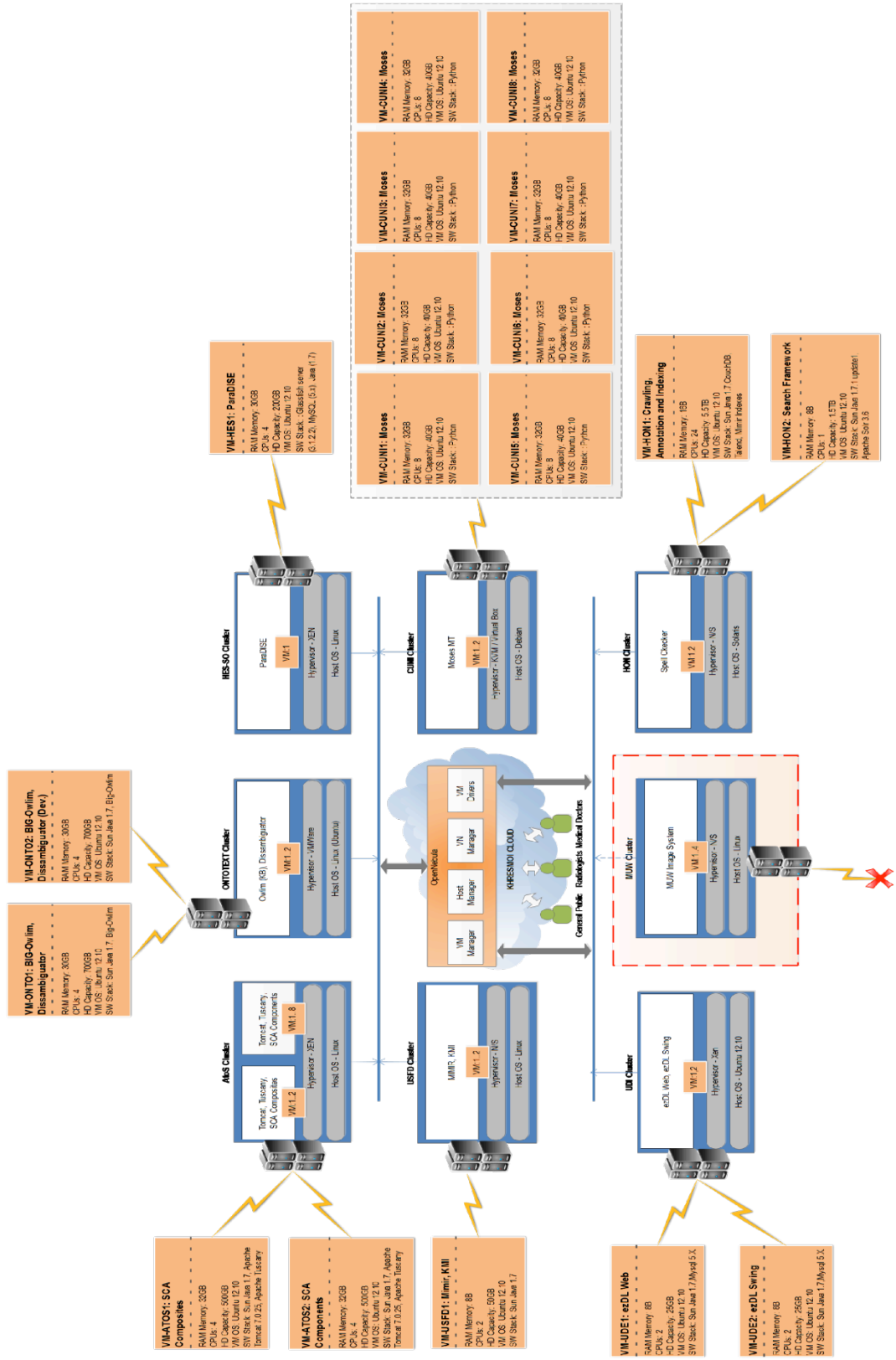

Figure 1. Khresmoi Full Cloud hardware and virtualised resources

The previous figure depicts the different nodes forming in the Full Cloud as well as the VMs deployed in each one of them. The list of VMs assigned to each node is:

Nodes	VMs	Component
Node1	VM-UDE1, VM-UDE2, VM-USFD1	ezDL + Mimir and KMI Components
Node2	VM-ONTO1	Dissambiguator Component
Node3	VM-ATOS1, VM-ATOS2	SCA Integration layer
Node4	VM-ONTO2, VM-HES1	Dissambiguator + ParaDISE Components
Node5	VM-CUNI1, VM-CUNI2, VM-CUNI3, VM-CUNI4	Multilingual Translator Component
Node6	VM-CUNI5, VM-CUNI6, VM-CUNI7, VM-CUNI8	Multilingual Translator Component
Node7	VM-HON1	Solr Search Framework
Node8	VM-HON2	Crawling

Table 3. List of Nodes and VMs deployed

7 Full Cloud Environment Description

The development of a production cloud environment requires several analysis tasks before starting with the deployment. These analysis tasks involve mainly technical or exploitation decisions that will determine the software and hardware environment. In the Khresmoi project, different aspects have been taken into account, such as:

- Nature of the data in order to choose a private/public cloud.
- Economic resources for choosing an open source/proprietary cloud manager.
- Technical requirements as hypervisor supporting in order to provide flexibility and easy installation characteristics.

These features have been evaluated to make a decision about hardware and software to be used for the Full Cloud prototype. In the following subsections these resources are fully described.

7.1 OpenNebula Overview

The Full Cloud prototype has been deployed using OpenNebula[2] software management cloud. This software was already used in the Early Cloud Prototype and supports the easy integration of any Virtual Machines images and several hypervisors. In the following paragraphs, we describe some of the key features that Open Nebula provides.

Figure 2 shows the Open Nebula infrastructure, which joins the front-end node with the rest of the hosts where the VMs are deployed through a physical network, such as the VLAN network, as well as other possible datastore nodes. This infrastructure is similar to a cluster architecture where hosts are accessible via the front-end interface and a distributed file system is accessible over the hosts.

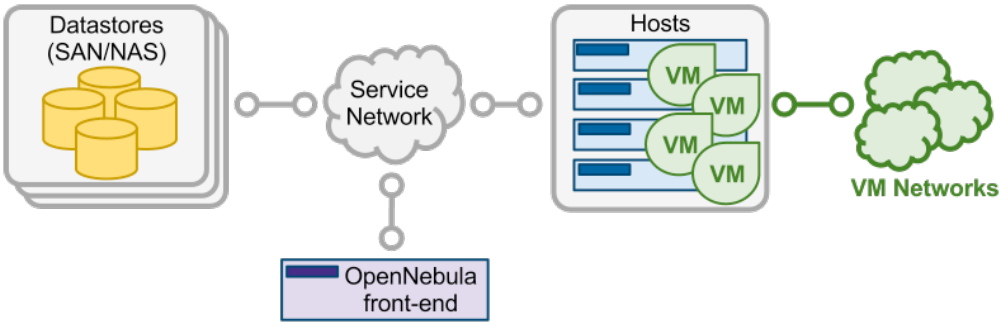

Figure 2. OpenNebula overview[3]

[2] http://opennebula.org/about:about

[3] http://opennebula.org/documentation:rel4.2:intro

The main components that appear in the OpenNebula infrastructure are:

- **Front-end**: is the machine in charge to access all hosts, VMs, virtual networks, etc.
- **Hosts**: physical machines or nodes where the VMs are deployed
- **Datastores**: filesystem paths where VM images are stored
- **Service Network**: physical network that allow connection between different components
- **Virtual Networks**: VLAN networks working on the VMs

7.1.1 Front-End

The front-end is the machine hosted in the cloud that allows the managing of all cloud resources in a centralised way. Therefore, this machine should have access to datastores and network connectivity to hosts and VMs. Moreover, an important feature of the front-end machine is the capacity to access with no-password authentication to the rest of cloud elements (hosts, VMs, etc.).

7.1.2 Hosts

The physical machines where VMs are created are named hosts. In our case, the hosts are the nodes1-8, dedicated to virtualise resources and shared by different Khresmoi components. These hosts must include, besides the VMs, an own Virtual Network and habilitate an SSH protocol communication in order to allow remote access to them.

These hosts can also be monitored through the OpenNebula software and other tools allowing the cloud administrators to trace its performance, connectivity, status, etc.

7.1.3 Storage

Another required feature for the OpenNebula installation and working is to place physically the VM disk images by using datastores. These datastores must be accessible from outside in order to allow transferring disk images to the hosts during VMs deployment process. There are different transferring methods, however, in this prototype we have used real transfer in order to optimise disk space in datastores.

These images can be located also in the management node, but the ideal architecture suggests separating them in the aforementioned datastores. To use file-based disk images and a shared file system of any kind to transfer the images allows taking full advantage of the hypervisor capabilities and typically offers better VM deployment times.

7.1.4 Networking

Networking specific configuration is required by the OpenNebula front-end services in order to access the hosts to transfer image files, as well as to manage and monitor the VMs. To do so, it is necessary to use a bridge in the physical host, that is, the management node, in order to redirect network traffic from outside to hosts and then to VMs.

In the following code extract, we show the node0 bridge specification in which the gateway and network range where VMs addresses will be defined is declared.

- `cat /etc/sysconfig/network-scripts/ifcfg-br0:`

```
DEVICE=br0
BOOTPROTO=none
HWADDR=00:25:90:88:21:CE
NM_CONTROLLED=yes
ONBOOT=yes
TYPE=Bridge
UUID=eded243e-55f4-4e09-9617-343dd07bea6b
NETWORK=195.113.21.0
NETMASK=255.255.255.0
IPADDR=195.113.21.3
GATEWAY=195.113.XX.X
```

7.2 Multi-hypervisor support

One of the main advantages of the OpenNebula cloud manager selection is its capacity of multi-hypervisor support. This is important because it provides out-of-the-box compatibility with different VMs from partners in case of integration or migration.

Despite this feature, all VMs deployed use the Xen[4] hypervisor, because of its integration with Open Nebula Sunstone[5] interface, as well as its easy command console interface.

7.3 Open nebula filesystem datastore

One of the main features required at the time of starting any virtualisation is the way to store all Virtual Machine information required for its proper operation. The Open Nebula filesystem, such as other distributed filesystems (HDFS, etc.), allows easy storing of VM images in different file formats depending on the Hypervisor used to create the image.

This filesystem datastore uses the standard console commands (ls, cd, etc.), so no extra installation or configuration is required to use it. Also, the filesystem store can work with shared or SSH transfer drivers at the time of replicating the files from management node to hosts. SSH option has been chosen for Full Cloud deployment in order to optimise the disk space capacity.

[4] http://www.xenproject.org/

7.4 Using the SSH Transfer Driver

The main disadvantage of using SSH transfer is that it may take a considerable amount of time when VM image sizes increase, introducing some delay in the system.

The structure of datastore files transferred during the process is the following:

- SSH copy process replicates in the nodes the folder structure, with an assigned number to each VM, under the path specified
- This structure, includes disk.0 and disk.1 files, that are respectively used to boot the VM from the OS image or from the installation already performed

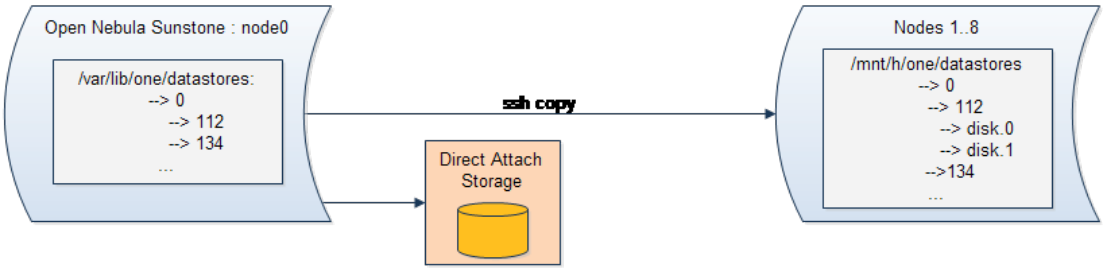

Figure 3. Open Nebula ssh transfer

7.5 Cloud Management

OpenNebula Sunstone[6] is the web service interface in charge of monitoring cloud elements and showing real-time information. This tool is basic, allowing to display live cloud components status via a web browser. The main parts of the sunstone web service interface are:

- Dashboard: this is the initial screen of the sunstone web service. It shows a summary status of the main components (hosts, virtual machines), as well as useful metrics about the main resources (CPU, Bandwidth consumed, etc.).
- System: this tab shows information about users, groups, access control lists, etc.
- Virtual resources: in this tab a list of all VMs deployed appears with the status, IP address and other useful information. Also, a list of templates and images used to create the VMs is shown and it is possible to check its configuration and edit some parameters if necessary.
- Infrastructure: this tab shows information related to physical resources, such as physical hosts, datastores, virtual networks, etc.
- Marketplace: the OpenNebula marketplace is a plugin search engine that allows easy installation of extra functionalities for a standard cloud deployment.

Figure 4 shows a screenshot of the Khresmoi Full Cloud prototype status via sunstone interface (dashboard).

[6] http://opennebula.org/documentation:archives:rel2.2:sunstone

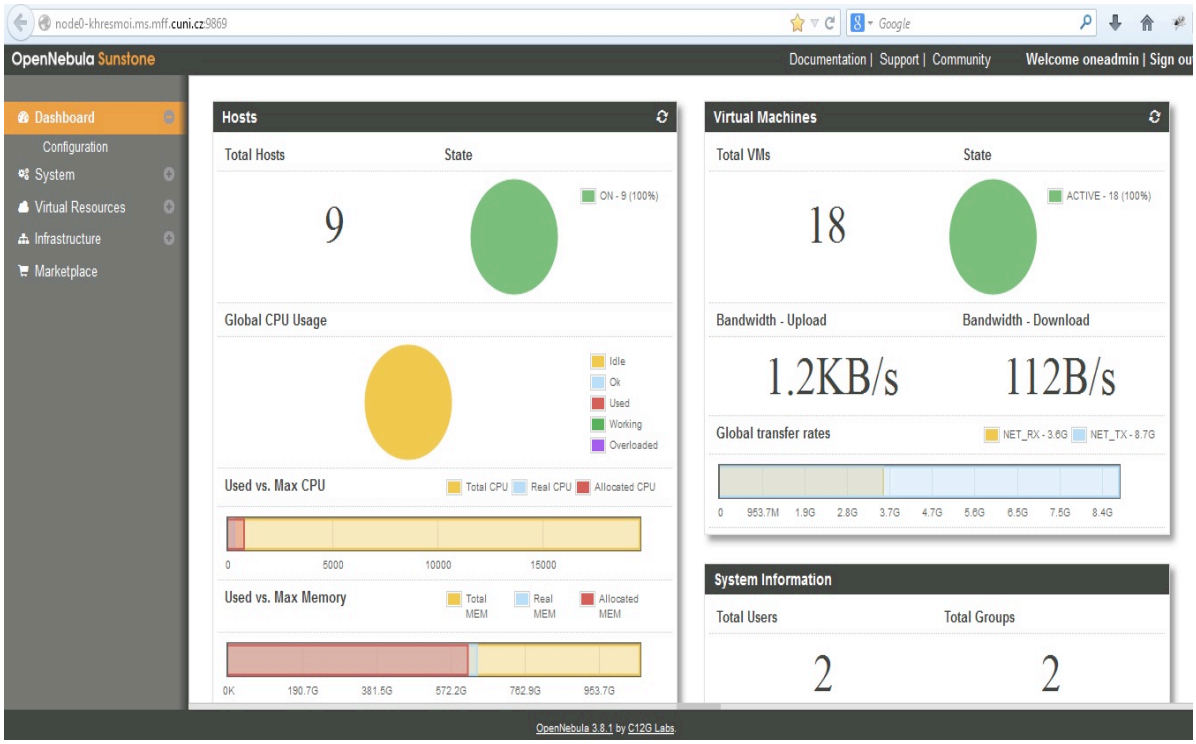

Figure 4. Sunstone web interface

7.6 Cloud Monitoring

Once the deployment is finished, a key task to ensure the maintenance of the Cloud is monitoring all hosts and services hosted on it. To do so, we need an Open Source monitoring solution that satisfies all monitoring requirements to keep us aware about the Khresmoi cloud status. Nagios[7] is a free open source tool to monitor cloud, clusters and other IT infrastructures remotely. Nagios allows defining hosts and services, as well as grouping them according to preferences. It offers by default different web services monitor, such as HTTP request, PING, etc. that can extended or modified easily.

Figure 5 shows the service overview for the Khresmoi Full Cloud using the Nagios tool.

[7] http://www.nagios.org/

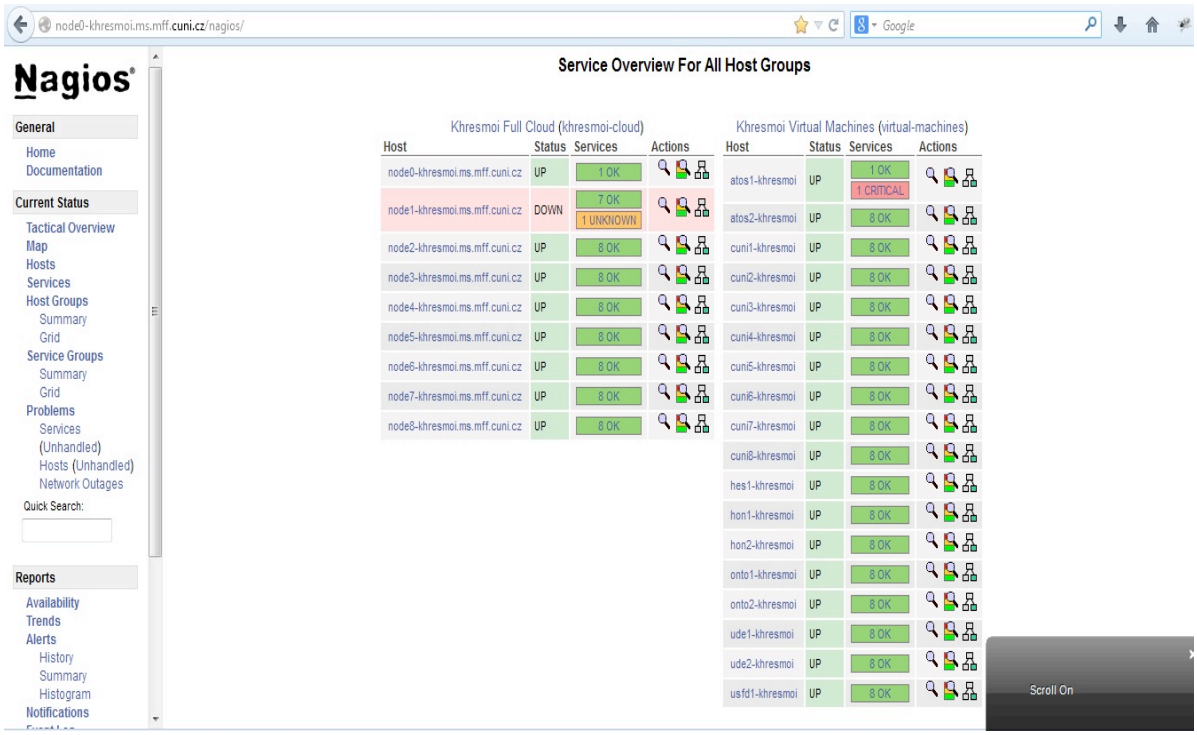

Figure 5. Nagios monitoring tool

8 Evaluation Approach

The work already performed in D6.4.2 has been continued as an iterative testing approach in order to assess the final Cloud deployment. Alike previous iterations, cloud testing evaluation approach is based on five steps shown in Figure 6.

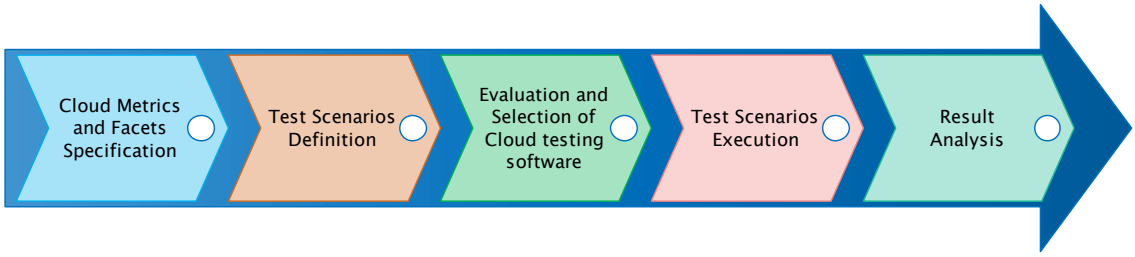

Figure 6. Evaluation Approach

Following this approach, the first step about Cloud Metrics and Facets Specification remains similar to the one followed for the previous iteration, because we aim to assess the same cloud features. The test scenarios also remain the same, as the main changes have been performed over hardware and software resources that are fully explained in chapter 6. Regarding evaluation and software selection, we use the same tool than for the Early Cloud prototype, SOASTA Cloud Lite Software[8], as it continues satisfying all the monitoring requirements for the Full Cloud Prototype.

Regarding the load for tests, we assume the same average users and queries per user as described in D6.4.2, because Khresmoi end-users remain the same as planned initially. Therefore, we will use for assessment: **60 virtual users with 4 queries per user in 5 minutes intervals**. We need to consider that load could increase in future productions environments, so Khresmoi platform should be sufficiently robust for higher amount of users and concurrent queries.

Finally, test scenarios execution and result analysis are deeply explained in chapter 9.

[8] http://www.soasta.com/products/cloudtest-lite/

9 Test Scenarios Definition

This deliverable focuses on the main Khresmoi workflows scenarios that have been achieved so far in every evaluation process as showed in D6.4.2.[1]: Textual Search, 2D Image Search, 3D Image Search and Multilingual Search. These scenarios have been continuously updated and improved iteratively since the project started and some of their components are likely to change since the last evaluation, as well as some steps of the workflow. Subsequently, these scenarios are presented with the main updates or changes that have been achieved in the last development iterations.

9.1 Textual Search Scenario

Textual Search scenario represents the same workflow than in the previous evaluation despite the fact that some of the components have suffered significant changes. In this final iteration, QMS Component has been replaced by KMI Component, which aims to simplify the process of query construction before querying over Khresmoi Mimir indexes. The main changes performed over Textual Search workflow are highlighted below:

Prototype 1	**Textual search**	
Components	ezDL (UDE)	ezDL is a multi-agent search system for heterogeneous data sources and a tool-set for building search user interfaces to support complex tasks
	Speller (HON)	HON's medical multilingual spell checking service
	Multilingual Translator (CUNI)	CUNI Multilingual translator service
	Disambiguator (ONTO)	ONTO Disambiguation service
	KMI (USFD)	KMI Service allows an XML file will be posted to the service, which will result in a Mimir search result being returned. The response from this service will be the same as a standard `postQuery` call to the Mimir web service API.
	Mimir (USFD)	Mímir search Web Service
	TextManager (AtoS)	This component manages the complete Textual Search Workflow that integrates the other components and its iterations
Scenario	Step1 : search(keywords)	The user introduces a list of keywords through the UI in order to obtain a proper answer. In this version, the system allows also to add

		some constraints in order to adjust the search.
	Step2 : improve query	The user obtains some improvements to the query keywords performed by the system: spelling correction, possible language translation and disambiguation.
	Step3 : perform search of final query	The query is transformed by KMI Component into a query_id that will be used to perform the search over the index
	Step4 : return list of Documents	The user receives a list of documents with hits found as the answer of the search
	Step5 : view document	The user can see and translate any document of the list returned from the search
Main functionalities to be integrated	Functionalities used by User: from ezDL	search(List<Keywords>)
	Functionalities used by User: from ezDL	viewDoc(docURI)
	Functionalities used by ezDL: from Speller	List<Suggestion> :getSpelling(keywords,lang)
	Functionalities used by ezDL: from MT	translateQuery(text,docType,sourceLang,targetLang,profile)
	Functionalities used by ezDL: from MT	translateDoc(docURI)
	Functionalities used by ezDL: from Disambiguation	List<Label> :getDissambiguation(keywords)
	Functionalities used by ezDL: from TextManager	searchByText(userProfile,keywords)
	Functionalities used by TextManager: from KMI	query_id :getQueryId(kmi-config,kmi-query)
	Functionalities used by TextManager: from Mimir	hitCount(index_id,query_id) hits(index_id,query_id,start_index,count) docMetadata(index_id,query_id,doc_id) docText(index_id,query_id,rank,term_pos)

Table 4. Textual Search Scenario Definition.

9.2 2D Image Search Scenario

Both Image Search workflows described in D6.4.2 [1] remain the same than in the last iteration. Despite this, some minor changes have been done over each workflow. 2D Image Search workflow changes are highlighted in the table below:

Prototype 1	**2D Image search**	
Components	ezDL (UDE)	ezDL is a multi-agent search system for heterogeneous data sources and a tool-set for building search user interfaces to support complex tasks
	ParaDISE (HEVS)	MedGIFT's new image search prototype Web Service called ParaDISE (Parallel Distributed Image Search Engine)
	SCA-ParaDISE (AtoS)	SCA Component for ParaDISE
	Repository	Image repository
	ImageManager (AtoS)	This component manages the complete Image Search Workflow (2D+3D) that integrates the other components and its iterations
Scenario	Step1 : search(images)	The user introduces a list of images through the UI in order to obtain a proper answer
	Step2 : the user obtains a list of images	The user obtains a list of images after the search processing ranked by a predefined score
Main functionalities to be integrated	Functionalities used by User: from ezDL	search(List<Image>)
	Functionalities used by ezDL: from ImageManager	search2DImages(List<ImageScore>,List<ImageCollection>, userProfile)
	Functionalities used by ImageManager: from SCA ParaDISE	searchImages(captionQuery, relevantImages, irrelevantImages,modalityFilter, booleanOperator, presets)

Table 5. 2D Image Search workflow updated

The new searchImages() method returns a JSON with images URLs and other information. Also, one can set different search options through the "presets" parameter. The different types allowed are improved (by default), standard and big data:

- Standard: It does not support multiple boolean operators, does not use the images' captions in the relevance feedback and does not perform any modality filtering.
- Improved: allows using the latest features implemented in the system.
- Bigdata: allow accessing a different image database than the one that is currently used.

9.3 3D Image Search Scenario

3D Image Search scenario has not been updated since the last iteration. Therefore, the full workflow description is fully explained in the previous deliverable D6.4.2 [1].

Prototype 1	**3D Image search**	
Components	ezDL (UDE)	ezDL is a multi-agent search system for heterogeneous data sources and a tool-set for building search user interfaces to support complex tasks
	MUW (MUW)	MUW 3D image retrieval web service
	SCA-MUW (AtoS)	SCA Component for MUW
	ImageManager (AtoS)	This component manages the complete Image Search Workflow that integrates the other components and its iterations
Scenario	Step1 : search(images)	The user introduces a list of images through the UI in order to obtain a proper answer
	Step2 : the user obtains a list of thumbnails from images	The user obtains a list of thumbnails from images similar to search performed
	Step3 : the user selects an image	The user selects an image thumbnail in order to obtain the full image
Main functionalities to be integrated	Functionalities used by User: from ezDL	search(Image,Text)
	Functionalities used by ezDL: from ImageManager	3DsimilarImages(image,text)

	Functionalities used by ImageManager from SCA MUW getImage(String id)

Table 6. 3D Image Search Scenario Definition.

9.4 Multilingual Textual Search Scenario

Multilingual Search scenario is a special case of Textual Search Scenario where input query from ezDL is translated dynamically. This workflow has already been evaluated in previous iterations and remains the same for the Full Cloud prototype. Despite the Multilingual Translator service being improved since last evaluation, the workflow has not suffered changes and can be checked out in previous deliverable D6.4.2 [1].

Prototype 1	**Multilingual query translation**	
Components	ezDL (UDE)	ezDL is a multi-agent search system for heterogeneous data sources and a tool-set for building search user interfaces to support complex tasks
	Multilingual Translator (CUNI)	CUNI Multilingual translator service
	SCA-MT (AtoS)	SCA Component for Multilingual Translator
Scenario	Step1 : translate(keyword)	The user introduces keyword through the UI in order to obtain a proper answer
	Step2 : the user obtains dynamically a list of possible translations	The user obtains a list of translations in a different language for the word introduced
	Step3 : the user selects a word	The user selects a word as a translation of the keyword entry
	Step4: Textual Search workflow	The rest of the scenario is the same as Textual Search workflow
Main functionalities to be integrated	Functionalities used by User: from ezDL	search(Image, Text)
	Functionalities used by ezDL: from SCA-MT	Translate(keyword)
	Functionalities used by SCA-MT: from Multilingual Translator	getTranslation(action, sourceLang, targetLang, text)

Table 7. Multilingual Textual Search scenario

10 Results Report

This chapter aims to describe a quantitative assessment approach by means of some useful cloud metrics and facets that allow us to evaluate the Khresmoi Cloud performance and quality. These main features have been already used in previous evaluations so they are summarised in the next subchapters.

10.1 Cloud facets and metrics

Cloud facets can be defined as the main features or characteristics that the Khresmoi system cloud should have. Such type of complex architecture should perform optimally in relation to three main aspects that were already defined in D6.4.2 [1]: **scalability, elasticity and availability.**

The Cloud Metrics most interesting for the assessment phase are those that allow testing hardware and software infrastructure in terms of performance. During the evaluation described in D6.4.2 [1], we used mainly cloud metrics referred to: **Fundamentals measures**, **CPU Usage, Memory Usage, Average Response Time and Bandwidth Usage**. These metrics were already defined in the mentioned deliverable, so they are not defined again in this document.

10.2 Metrics results

As for previous evaluation deliverables [1] [2] [3], the metrics results obtained during the evaluation are useful to show an overview of the Full Cloud prototype performance. These results have been obtained using the SOASTA Cloud Lite software[9] that has been already used in previous evaluations and which allows production environments simulations to assess software architecture. This tool also provides metric diagrams about the simulation performed, as well as the numeric values that can be compared over different simulations in order to extract useful conclusions about architecture performance.

In the coming sections, different scenario simulations and executions are described along with the diagrams and numeric results obtained. Finally, these simulations will be interpreted in order to understand the main goals achieved or issues to improve for future architecture refinements.

The following subsections show the results for different scenarios defined previously, as well as the diagrams obtained from SOASTA Cloud Lite test software.

10.2.1 Textual Search Scenario Results

The Textual Search scenario assessment can be performed as in previous deliverables [1], performing an HTTP GET request to the Khresmoi-textual-search SCA Composite URL that manages the scenario workflow:

http://atos1-khresmoi.ms.mff.cuni.cz:8080/khresmoi-textual-search/rest/documents?query=diabetes

This SCA Composite has been deployed in the new Full Cloud infrastructure, in one of the dedicated ATOS machines, as can be observed in the URL. The main goal for this scenario is to demonstrate that

[9] http://www.soasta.com/products/cloudtest-lite/

the system has improved its response to the same query related to "diabetes" search, which has been tested in previous evaluations. Also, the simulation includes an increasing load in number of users and requests in order to test simultaneous users' requests over the system.

However, final releases of KMI and Mimir have not been deployed. Therefore, textual search has been tested over a development version that soon will be finished with complete indexes from crawled sources.

Below, the main metrics results and diagrams are presented as well as comparative interpretations regarding the previous deployment.

10.2.1.1 Fundamentals

The Fundamentals box summary shows the main metrics about the simulation executed. In this case, the following information is observed:

- Elapsed time: refers to the full time required to finish the simulation (can be previously set)
- Messages/Actions completed: the number of messages delivered
- Composition status: the status of the simulation (should be completed if finished)
- Name, Composition and start: just some logging information about execution
- Finally, the most important metrics showed in Fundamentals are:
    - **Average response time:** shows the average time needed to deliver the messages or complete the actions simulated. In this case, we observe a value that is clearly lower than in previous evaluation [1]. This means, that the same query over the Full Cloud prototype improved by almost 50% the Average response time metric.
    - **Effective throughput:** this metric displays the number of messages per second successfully delivered. In our case, we obtain an effective throughput of 1 that can be considered as an optimum value. Hence, we can conclude that system efficiency has been maintained with respect to previous evaluations.

Below, it is shown in Figure 7, the Fundamentals box returned by SOASTA Cloud Lite tool [10] with all the information previously explained:

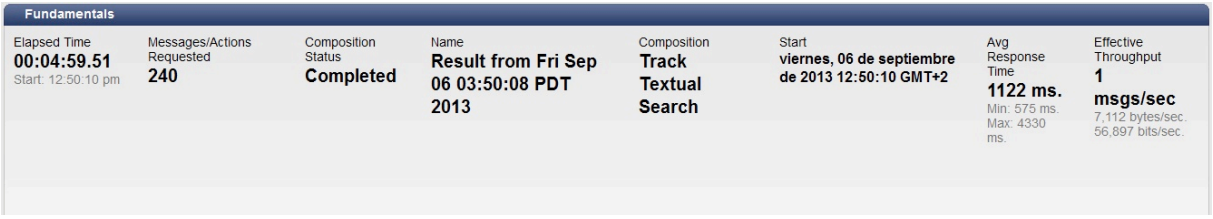

Figure 7: Fundamentals Textual Search scenario

10.2.1.2 CPU Usage

CPU Usage is a critical metric to assess the well-operating of the Hardware infrastructure. In this phase, hardware resources have increased in order to ensure a good performance of the system. In this context, CPU Usage should improve with respect to the previous evaluation, in terms of percentage of use. According to this hypothesis, we ascertain the following statement as true: "CPU Usage highest value in this phase reaches 39 % whereas in previous evaluation [1] it reaches 100% in several occasions". This means, that the CPU Usage now is lower due to hardware resources increment

[10] http://www.soasta.com/products/cloudtest-lite/

performed for the Full Cloud deployment. These new resources should ensure a good performance of the system in terms of CPU consumption, as checked in this case.

Figure 8 below shows the CPU Percentage diagram related to Textual Search scenario composition:

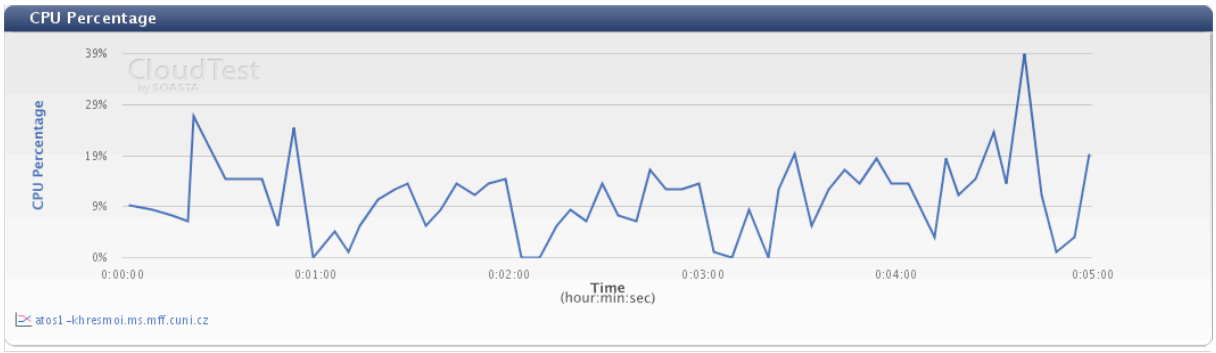

Figure 8: CPU Percentage for Textual Search scenario

10.2.1.3 Memory Usage

The Memory Usage metric shows RAM memory consumption during composition execution. As for CPU Usage, RAM Memory resources have been increased for the Full Cloud deployment, so memory consumption should improve performance in terms of Memory consumption. In this context, lower memory consumption with respect to the previous evaluation [1] is observed, reducing consumption from 1.4 Gb to 500 Mb. Figure 9 shows the diagram of Memory Usage during the composition execution:

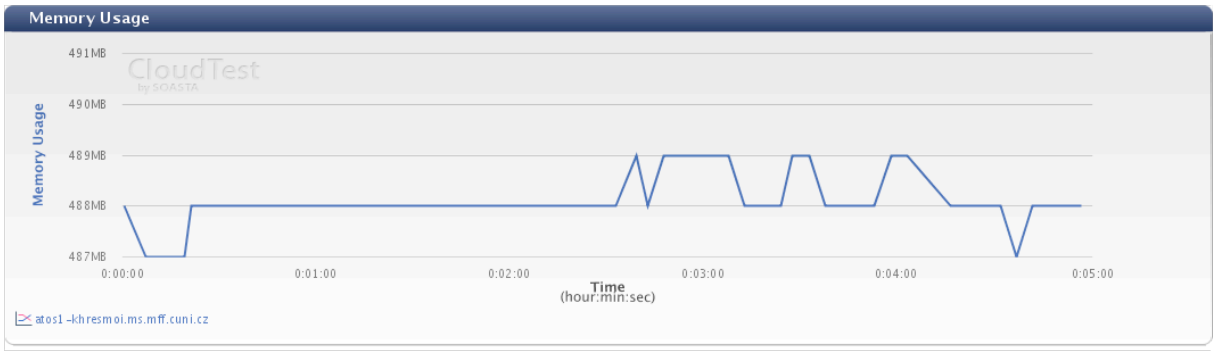

Figure 9: Memory Usage for Textual Search composition

10.2.1.4 Average Response Time

This metric is one of the most important in order to assess the performance, because its value affects directly end-users when retrieving information. Figure 10 shows generally low response values, except a few peaks where response time is between 3-4 seconds. These can be caused by simultaneous requests to the service, but it is an exception because it doesn't really affect the average response time. The value obtained now improves by almost 50% with respect to the previous evaluation [1] so we can conclude that an optimal improvement was achieved during the Full Cloud deployment.

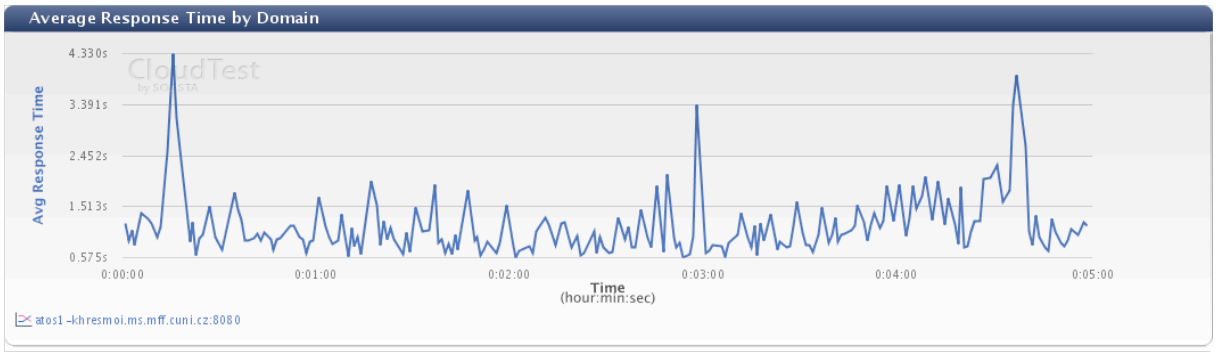

Figure 10: Average response time for Textual Search composition

10.2.1.5 Network Bandwidth

The network bandwidth required for Textual Search workflow is displayed in Figure 11: Network bandwidth Textual SearchFigure 11. We can observe that consumption is quite stable except for some peaks at the beginning of the execution. Future work must check these peaks because they could behave like bottlenecks in extreme situations.

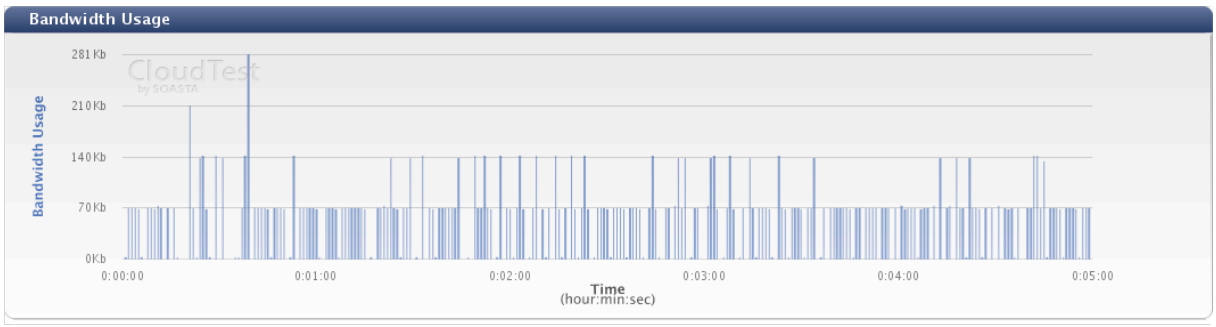

Figure 11: Network bandwidth Textual Search

10.2.1.6 Virtual Users

Virtual users' metric allows testing the concurrent access to the infrastructure in order to assess its capacity to deal with several simultaneous requests. These virtual users are set in order to perform a concrete number of requests in an incremental mode. This way, access to infrastructure should group different requests at the end of the execution. Figure 12 shows how two Virtual Users overlap at some points of the simulation, although the overlap can be considered as minimal. Therefore, the Full cloud prototype can be considered as able to deal with the pre-set number of simultaneous users and requests as described in Evaluation Approach 8.

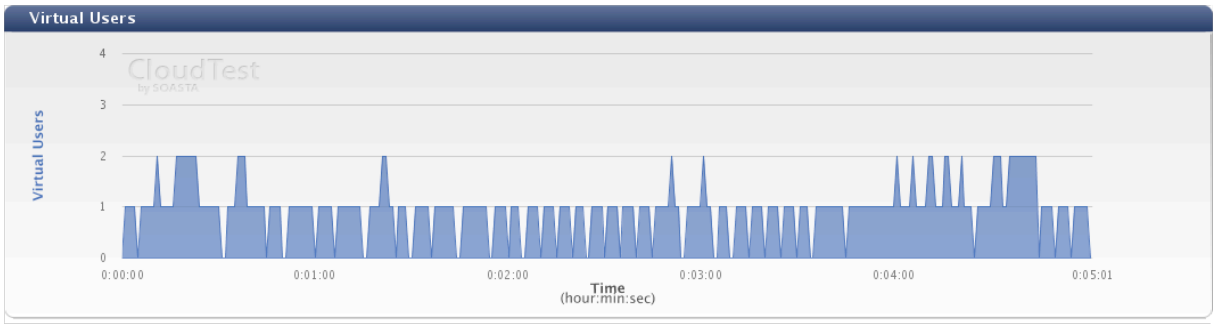

Figure 12: Virtual Users Textual Search

10.2.2 2D Image Search Scenario Results

This scenario carries on performing the evaluation already started in the previous deliverable [1] for the Early Cloud Prototype. Regarding the Full Cloud evaluation, it is assumed that the same configuration for 2D Image search is kept. In this context, this workflow is also launched by a GET Method with the following parameters:

- caption-query = text queried over images repositories. In this example, set as "diabetes".

Additionally, two new parameters have been added:

- Profile = filter the dataset used. For example, "big-data" value.
- Maximum-results= to limit results per query (5 by default)

The new URL that supports the SCA Composite in Full Cloud prototype is shown below:

http://atos1-khresmoi.ms.mff.cuni.cz:8080/khresmoi-image-search/rest/images/2D/searchImages?caption-query=diabetes&profile=bigdata&maximum-results=5

This method returns a JSON response with the IDs of images matched, therefore this infrastructure deployed must be able to deal with several requests in a good response time. The composition assumes an increasing load over the system during the simulation, so it is interesting to check how the system responds under this situation.

The different metrics results related to 2D Image Search are described in the following subsections.

10.2.2.1 Fundamentals

Fundamental metrics show, as in the previous scenarios, the main numeric information values after the simulation. As mentioned before, the most important items in this summary are the Average Response Time and Effective Throughput. In this phase, we notice how average response time has decreased by almost 50 % with respect to the previous evaluation [1]. Moreover, effective throughput maintains its value of 1 and can thus be considered as optimum. Therefore, Fundamentals metrics (Figure 13) reveal that the Full Cloud prototype basically improves the 2D Image Search workflow performance.

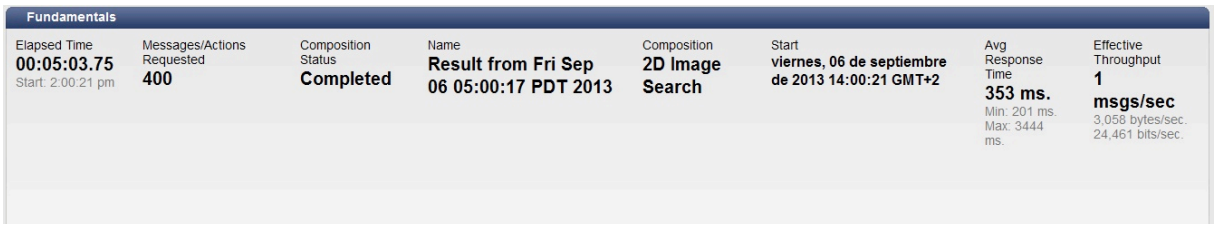

Figure 13: Fundamentals 2D Image Search

10.2.2.2 CPU Usage

The CPU Usage metric describes the percentage of process capacity consumed during 2D Image search simulation. Similar to the Textual Search scenario, the new Full Cloud infrastructure should fulfil all hardware resources requirements for 2D Image Search. In Figure 14 below, it is shown how CPU consumption percentage highest value is 20% at its peak, whereas it is usually lower during the rest of the execution. This means that hardware resources are capable of dealing with test simulation performed for 2D Image search and it is possible to infer a correct performance of the infrastructure in similar production environments.

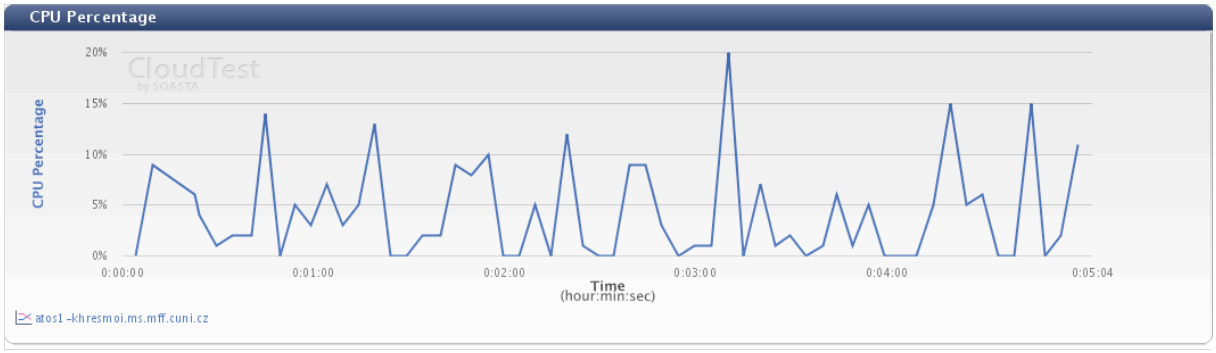

Figure 14: CPU Usage 2D Image Search

10.2.2.3 Memory Usage

The Memory Usage metric describes memory requirements consumed during 2D Image search composition in order to deal with requests performed. In this case, Figure 15 shows a stable consumption of about 500 Mb RAM, which represents a small fraction of the memory available in the Full Cloud prototype. According to this, it is possible to infer that the current infrastructure can deal with 2D Image search scenario in a real production environment.

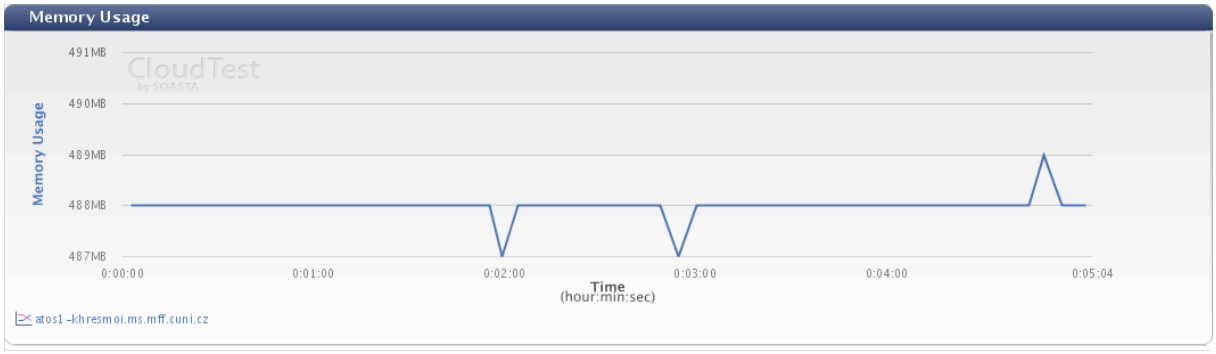

Figure 15: Memory Usage 2D Image Search

10.2.2.4 Average Response Time

Average response time can help us to determine how fast the system is dealing with the incoming requests. For the 2D Image search scenario, the infrastructure appears to be quite fast and improves the value by almost 50% with respect to the previous evaluation [1]. Despite this, it is possible to observe some exceptionally long peaks in responses, although they are not really critical, as long as the system is able to deliver the rest of the requests in an efficient way. Figure 16 shows the average response time for complete 2D Image search composition:

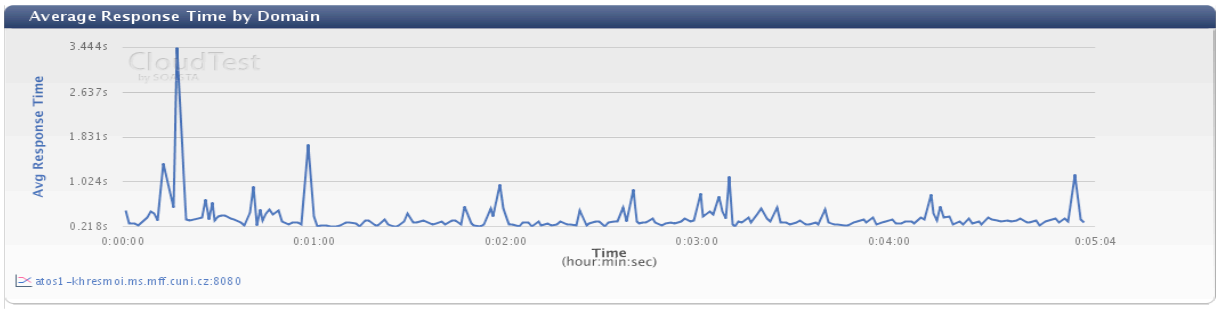

Figure 16: Average response time for 2D Image Search

10.2.2.5 Network Bandwidth

The Network bandwidth metric can be used to identify possible bottlenecks in the network while composition execution. Figure 17 shows network performance during 2D Image search execution and we can appreciate how the network is barely used simultaneously by several requests that can be interpreted as an optimal bandwidth capacity. This feature is also observed in low Kb traffic per query that may affect positively the overall system performance.

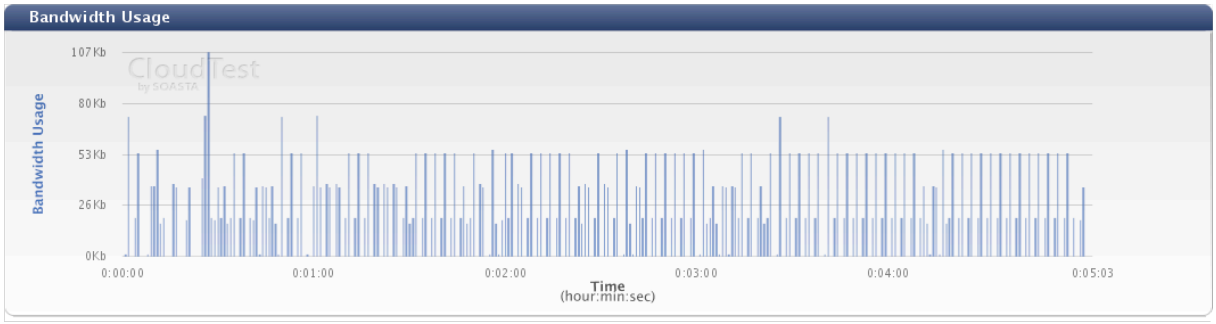

Figure 17: Network bandwidth 2D Image Search

10.2.2.6 Virtual Users

Similar to the previous scenario, virtual users have been scheduled via Cloud testing software in order to test overlapping user requests. Despite this premise, Figure 18 shows a few overlapping requests due to optimum infrastructure capacity to manage requests very fast and therefore only 2 or 3 simultaneous users in a concrete moment are observed. This feature is a good approach for future productions where several real end-users may access simultaneously the system and the system must be able to respond satisfactorily.

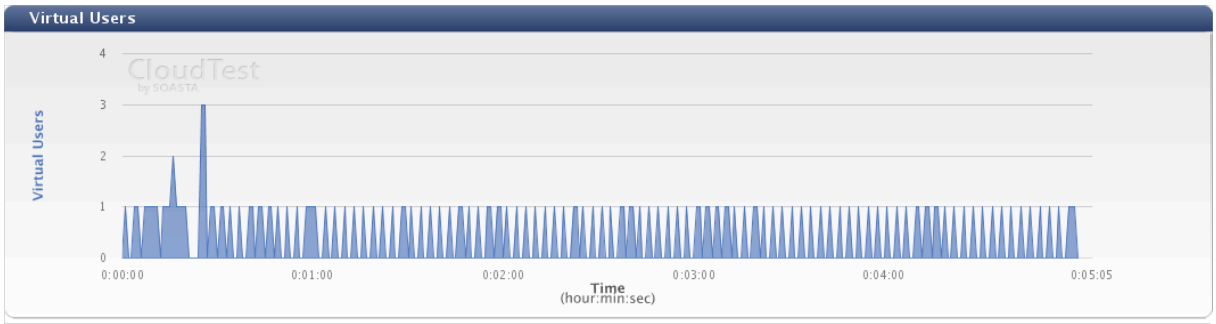

Figure 18: Virtual Users for 2D Image Search

10.2.3 3D Image Search Scenario Results

The 3D Image Search workflow assessment has not been performed for the evaluation reported in this document. This circumstance is due to the fact that so far the MUW SCA Component has not been integrated in the Full Cloud prototype. Therefore, it has not been possible to evaluate its operation within it.

Consequently, the 3D Image Search scenario evaluation has been postponed until future evaluations in case the SCA MUW Component is finally integrated into the Full Cloud infrastructure.

10.2.4 Multilingual Textual Search Scenario Results

Similar to previous evaluations, the Multilingual Textual Search scenario runs a special use case of Textual Search where end-user queries are translated dynamically. The main changes in this iteration are that both services (SCA MT and SCA Textual Search) are deployed into the Full Cloud Prototype and therefore the complete workflow is executed into it. The new service URLs are:

- MT Service: http://atos2-khresmoi.ms.mff.cuni.cz:8080/khresmoi-MT/rest/MT/translator/get/simple?action=%22translate%22&sourceLang=de&targetLang=en&text=Zuckerkrankheit
- Textual Search: http://atos1-khresmoi.ms.mff.cuni.cz:8080/khresmoi-textual-search/rest/documents?query=diabetes

The test performed is the same as in the previous evaluation 45[1]:

As in the previous evaluation [1], this scenario simulates a query performed by an end-user, for instance "diabetes", the same as in the previous testing to allow comparing the results. This workflow is composed by two main steps:

- Translation: the query is dynamically translated to the target language. The same text query "Zuckerkrankheit" is used, which means diabetes, as in the previous evaluation in order to compare both composition results.
- Searching: search is performed using the query translated in the previous step, launching the usual Textual Search workflow.

10.2.4.1 Fundamentals

Fundamentals results display an overview of composition execution, with regard to the number of complete messages delivered, as well as the time of execution. Also, two main metrics in Figure 19: Fundamentals MT Textual SearchFigure 19 offer some new values:

- Average response time (ART): here the average responding time of two requests that compose the execution is displayed. Therefore, the ART final value is obtained after multiplying x2 the value obtained, due to the fact that it should be ART first request + ART second request.
- Effective throughput: for the same reason, the effective throughput increases to 2 messages per second, because this time the composition must deliver two request instead of one in order to complete the workflow. In this context, it is a good approach, as long as ideally the system performance should not be affected by the workflow complexity to respond properly.

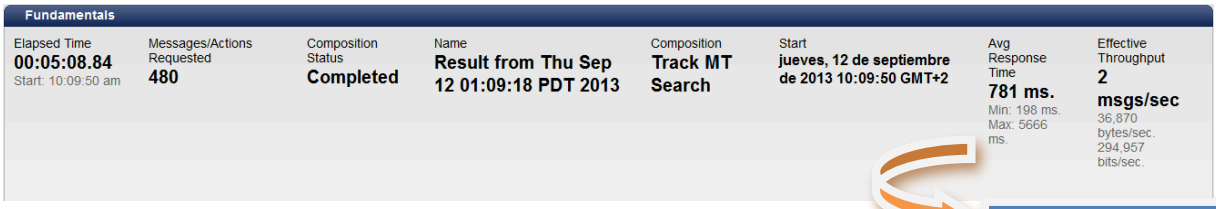

Figure 19: Fundamentals MT Textual Search

10.2.4.2 CPU Usage

The CPU Usage represented in Figure 20 below displays higher processing resources consumption compared with the simple Textual Search. However, the CPU consumption highest value reaches 40%, which is not a critical value in any case. Therefore, we can infer that complex workflows such as Multilingual Textual Search are able to run in the Full Cloud Prototype properly.

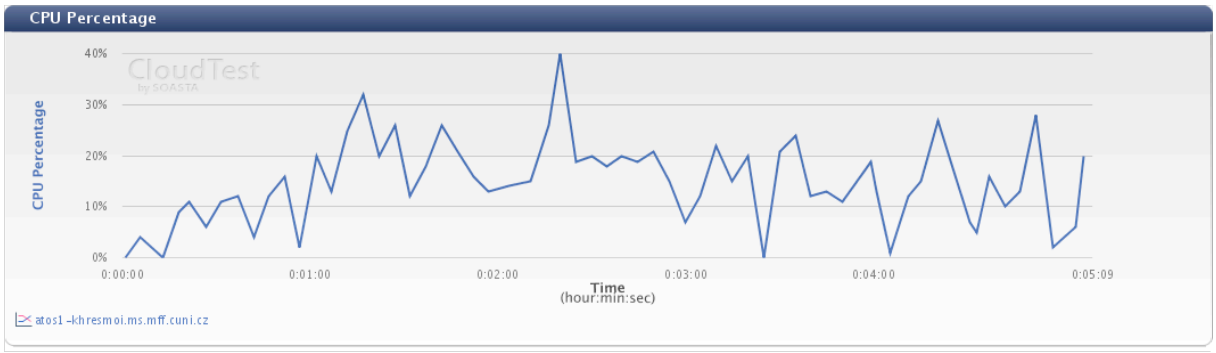

Figure 20: CPU Usage MT Textual Search

10.2.4.3 Memory Usage

Memory Usage is shown in Figure 21, which reflects a quite stable consumption and low values. Hence, this workflow does not require consuming much more memory than simple workflows and full cloud resources currently satisfy both types of executions.

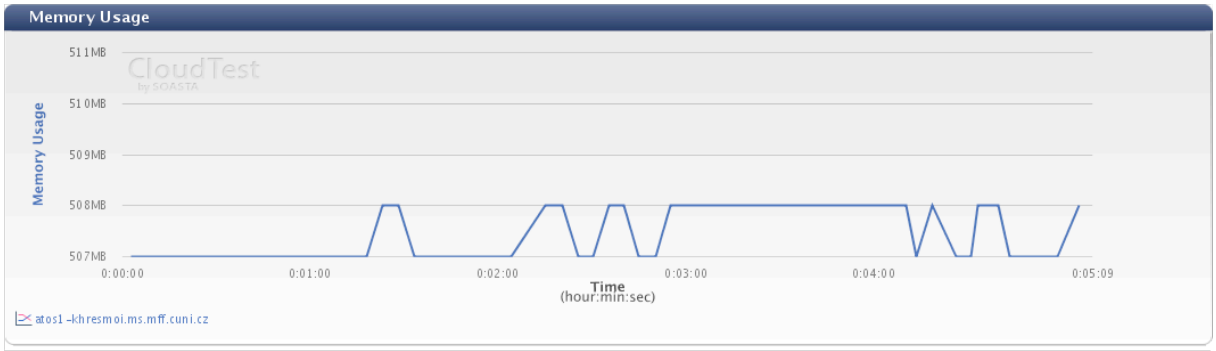

Figure 21: Memory Usage MT Textual Search

10.2.4.4 Average Response Time

Figure 22 below shows the average response time for different requests that are required for the workflow. Therefore, we can check the two different SCA requests:

- SCA Multilingual translator request: represented by the blue line, it displays the lowest values of execution except in a peak produced by the time ramp function. However, this load increase at the end of execution does not affect the overall performance.

- SCA Textual Search: represented by the green line, obviously shows the highest response values due to the complexity of its workflow. Despite this, it is observed how most part of the execution response times are below of 1,5 seconds, except some minor peaks where response times increase considerably. These types of malfunctioning behaviours will be observed and improved as much as possible during next year's infrastructure refinements.

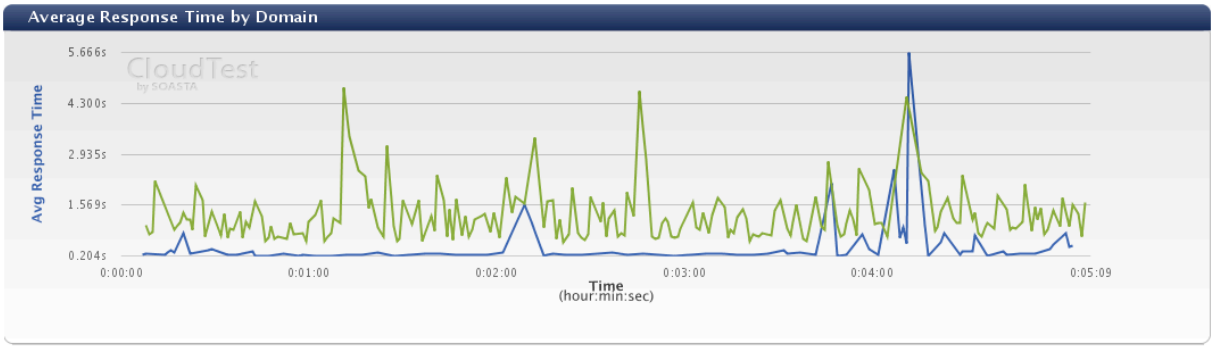

Figure 22: Average Response Time MT Textual Search

10.2.4.5 Network Bandwidth

The Network bandwidth consumption represented in Figure 23 below displays the network traffic during multilingual workflow execution. As can be observed, traffic is quite variable, being mostly of about 500 Kb per request. However, some peaks in network traffic may be caused by simultaneous requests. These peaks must be controlled for future infrastructure refinement during KHRESMOI last year in order to avoid performance bottlenecks.

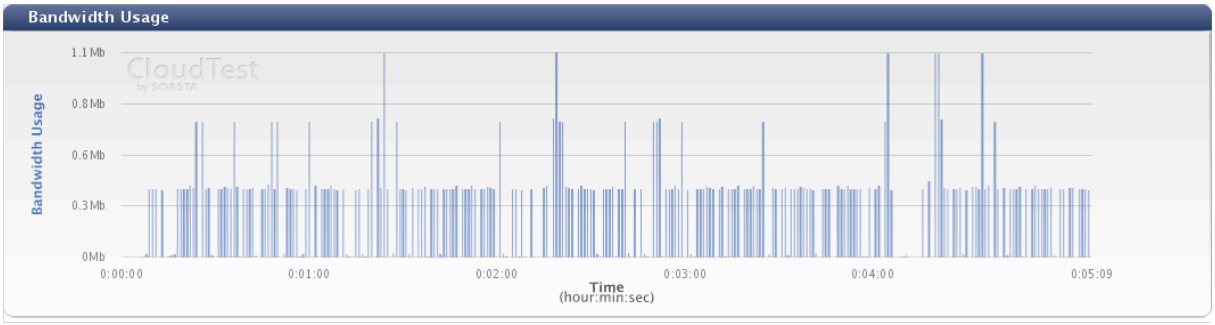

Figure 23: Network Bandwidth MT Textual Search

10.2.4.6 Virtual Users

The Virtual Users data, gathered during the multilingual scenario workflow, show in Figure 24 an increasing access to the platform. This concurrent access is simulated in order to test the system capacity to deal with simultaneous requests. In this context, the highest value observed is of 3 concurrent users requesting over multilingual scenario. The peaks observed in the Network Bandwidth and Average Response Times diagrams are related with virtual users overlapping shown below:

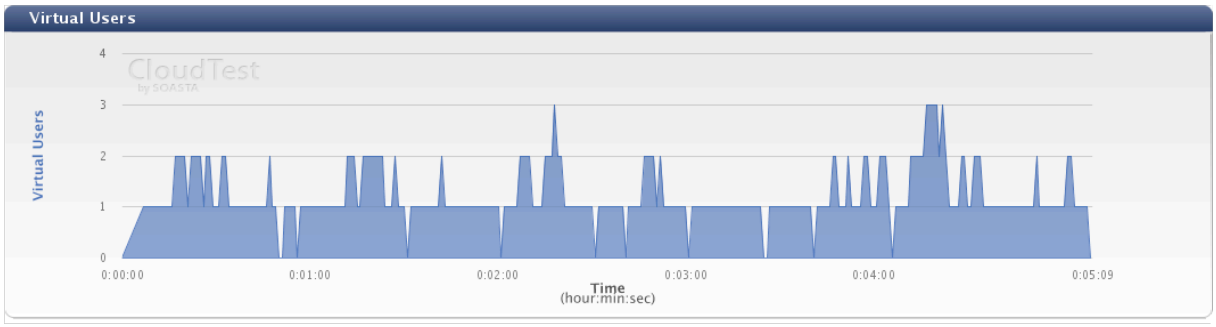

Figure 24: Virtual Users MT Textual Search

11 Facet-Linked Result Analysis

The Facet-linked result analysis aims to extract valuable conclusions about infrastructure relating numeric results with the main KHRESMOI Cloud facets already defined in subchapter 10.1 and the previous deliverable [1]. These facets (scalability, elasticity and availability) are used to create the radar chart diagrams used to compare the results that will be showed in the next subchapters. These radar charts are linked to test scenarios defined in chapter 9 and results reports in chapter 0.

Subsequently, the facets according to the metrics results for each test scenario performed over the Full Cloud Prototype (FCP) are described, as well as compared to the previous Early Cloud Prototype (ECP) evaluation [1], where first facets-linked results were introduced.

11.1 CPU Usage

Figure 25 below displays the comparative results of the CPU percentage consumption for the Early Cloud prototype (ECP) and the Full Cloud prototype (FCP). This diagram shows clearly an important decreasing of CPU consumption between both phases in different scenarios, which could be caused by the increasing of hardware resources used for the FCP. This improvement aims to increase infrastructure capacity to manage several requests for the future production environment deployments. In this context, the main facets related to this change may be affected in the following way:

- Scalability: scalability is a critical facet, as it allows to the system adjust its resources to the dynamically changing needs. This facet has not been tested exhaustively due to the ample capacity of the system to deal with the requests so far.
- Elasticity: this facet allows us to check whether the system is able to adapt to different environments and different load scenarios. Currently, as shown in Figure 25, the CPU elasticity has not been properly tested, as the capacity of the system has not required testing it so far with more resources than available right now.
- Availability: contrarily to the Early Cloud prototype, where the high CPU consumption puts the availability of the system at risk, the current prototype shows a much lower CPU consumption, as can be observed below in Figure 25. Hence, so far the CPU consumption allows us to ensure the availability of the system for similar load tests.

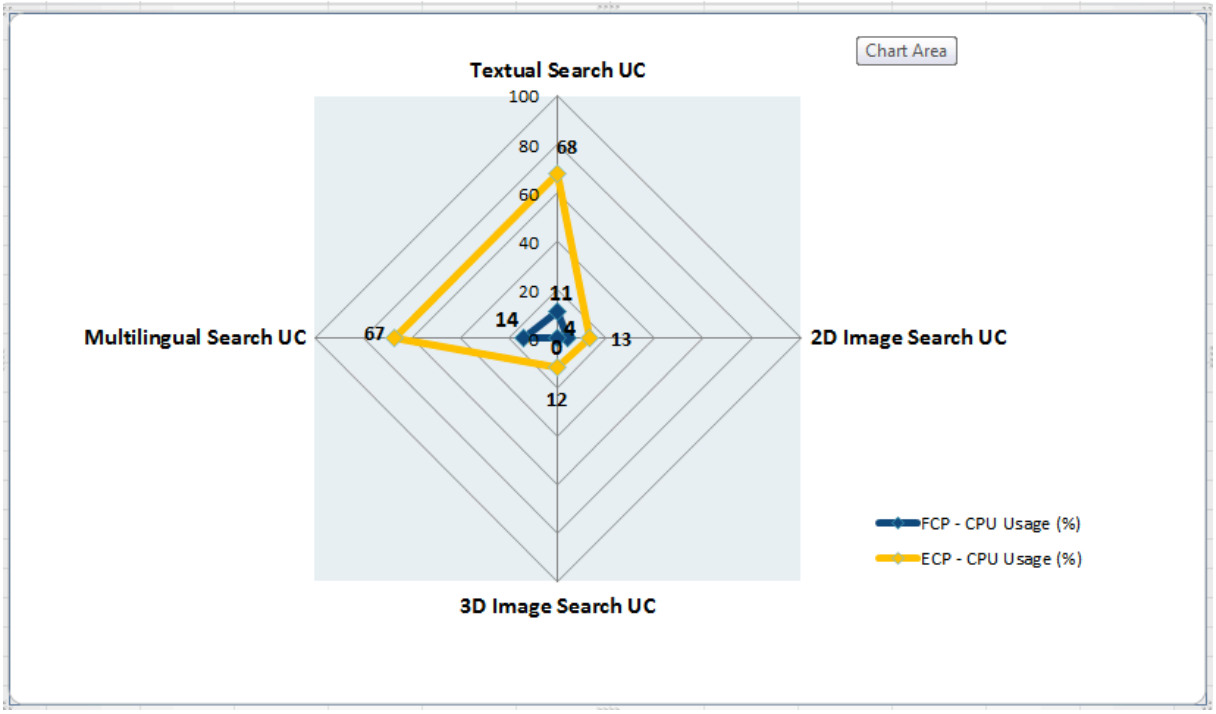

Figure 25: CPU Usage Radar chart diagram

11.2 Memory Usage

The Memory Usage radar chart in Figure 26 allows comparing the different memory consumption in both prototypes: Early Cloud Prototype (ECP) and Full Cloud Prototype (FCP). Similar to the previous ECP evaluation [1] stable memory consumption is observed. In the same way, this consumption can be caused by constant memory requirements to launch the SCA services within the Tomcat application server. Moreover, the lower values observed for FCP may be justified because the new hardware resources have more capacity for processing requests dynamically and need less RAM memory to deal with same test scenarios compositions. Regarding the main cloud facets, the Memory Usage could affect them as following:

- Scalability: this facet is ensured regarding memory dynamic needs because the percentage required in the test compositions is very low. We can infer that future and more exhaustive tests will be able to run without difficulties due to good scalability of the system.
- Elasticity: this facet relies on the system capacity to adapt memory consumption depending of the workflow in execution, number of requests, etc. According to this, it is observed how none of the workflow requires too many resources, so currently system elasticity is not a key feature although future tests and refinements may show different results.
- Availability: availability of the system regarding memory consumption is ensured with the current load. Memory needs are much lower than resources available and therefore system availability appears as a quite stable.

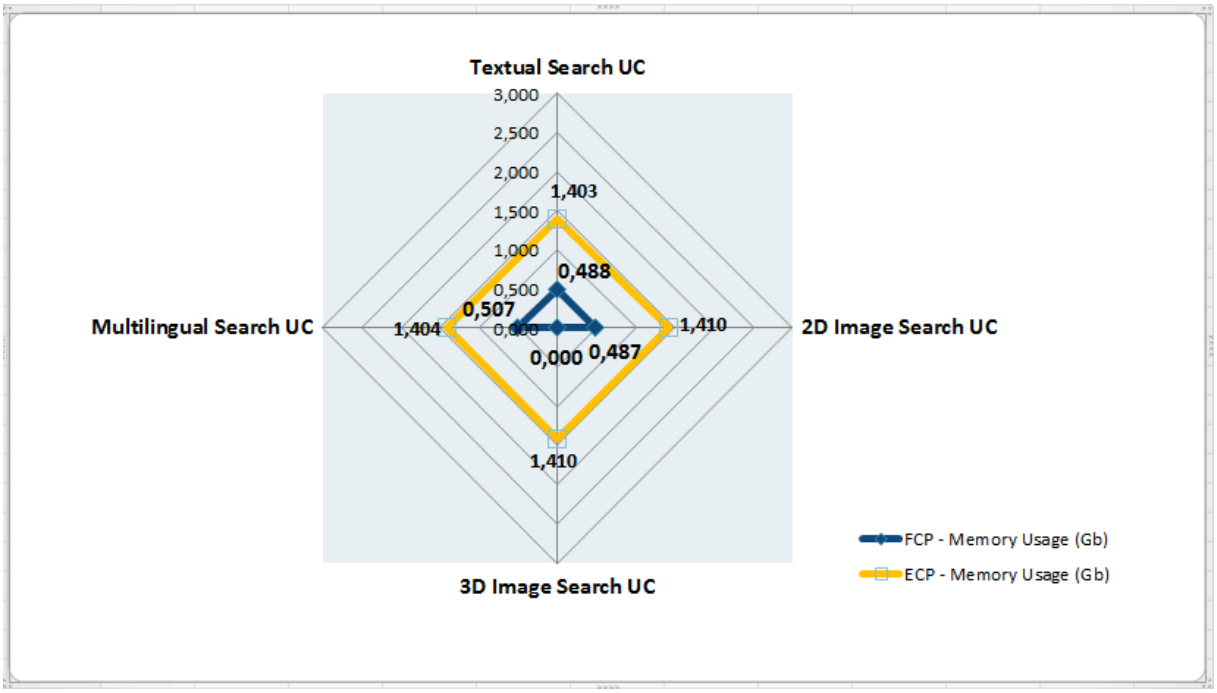

Figure 26: Memory Usage Radar chart diagram

11.3 Average Response Time

Average Response Time is displayed in Figure 27 where it is compared with ECP results. We can clearly see how FCP results are lower than in the previous evaluation. This can be justified because in ECP evaluation every component was deployed separately in each partner machine, whereas the FCP contains a Component instance deployed in the nodes of the FCP cluster. Therefore, time responses must be lower as requests can be processed in the same network without being redirected to different partner networks geographically separated. The main facets are described according to Average Response Time results below:

- Scalability: regarding time responses metric, scalability must ensure that the increasing load of requests does not create bottlenecks at the time of delivering messages. So far, tests performed have been successful although future infrastructure refinement will focus in avoiding this problem.
- Elasticity: aligned with the previous evaluation, changes performed in infrastructure resources have allowed reducing ART for every test scenario composition. Despite this, possible future refinements for production deployment must check elasticity of the system with a higher requests load.
- Availability: ART results have improved since the ECP and ensure as well the availability of the system, as there are no response times before time out. This feature is mandatory for future production deployments along with an optimum ART lower than 2 seconds.

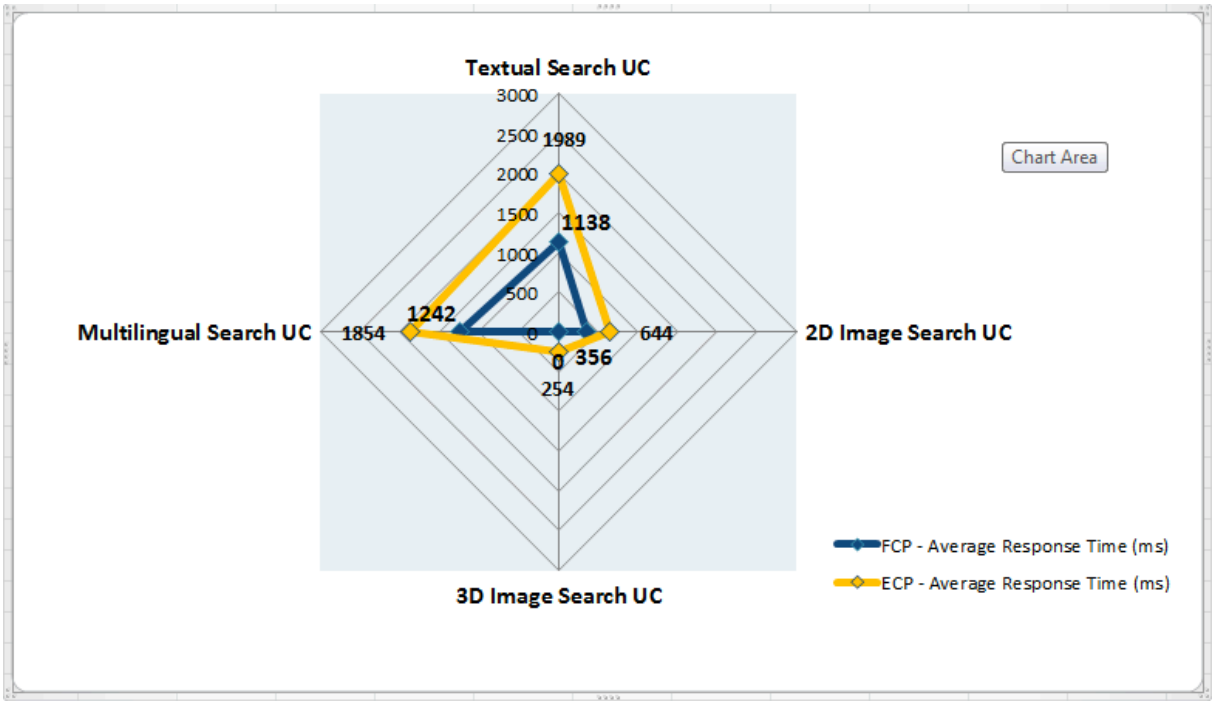

Figure 27: Average Response Time Radar chart diagram

11.4 Network Bandwidth

The Network bandwidth diagram in Figure 28 displays a comparative chart radar diagram of FCP and ECP results. A considerable reduction of network bandwidth use during FCP composition test is observed. This is due to the fact that transmission intervals of data packets also influence the bandwidth consumption: as we increase the number of kbps, there is more latency (delay) in the transmission of the data, but less consumption of bandwidth (and vice versa). Also, the volume of data sent has increased in most workflows, but instead is down in general terms the number of calls for data transmission. Both reasons coupled with an improvement in HW infrastructure have resulted in a significant reduction of bandwidth in general terms. The main facets have been affected with these changes as following:

- Scalability: this facet has been improved for FCP because currently the different scenarios workflows require less bandwidth. This affects positively the scalability of the system, avoiding bottlenecks even with the highest load of requests.
- Elasticity: elasticity has no direct effect in the current system performance due to the low bandwidth usage. Future production deployment will probably need to improve system elasticity regarding the network bandwidth.
- Availability: low network consumption ensures good network availability and avoids bottlenecks.

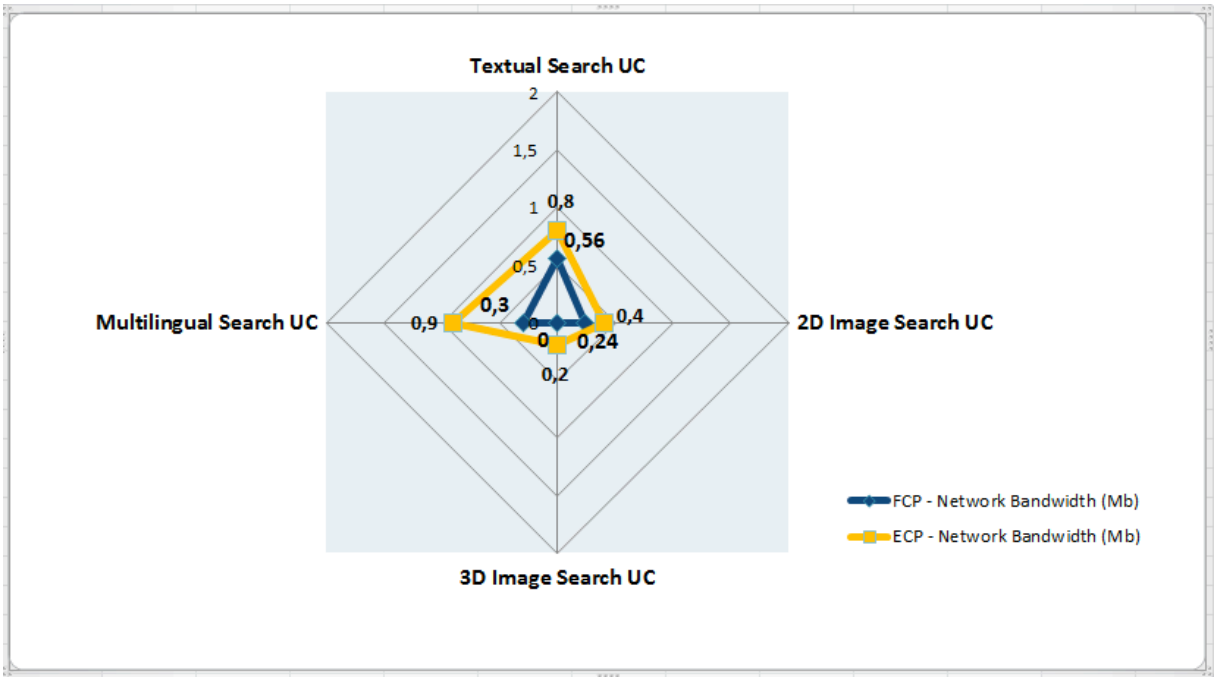

Figure 28: Network Bandwidth Radar chart diagram

12 Conclusion

In this deliverable, we have presented the final stage of Khresmoi Cloud development, as well as all the improvements performed since the last prototype. The Full Cloud Khresmoi prototype aims to fulfil all the requirements identified so far in order to offer a robust infrastructure for the production release of the Khresmoi platform. To do so, after a phase of refinement, new Hardware resources capable of deploying a Cloud solution and encompassing all Khresmoi components in a production and stable release must be determined.

Section 0 about results report aims to describe the evaluation process over the test composition, as well as to extract useful conclusions about the status of the current Khresmoi deployment. The main cloud metrics measured unveil a clear improvement in terms of performance of the main workflow use cases (Textual Search, Image search, etc.). These improvements cover mainly two aspects:

- Cloud hardware resources improvement: the hardware resources have been widely improved in order to provide a powerful cluster of machines able to manage a Khresmoi production environment.
- Common components deployment in the Khresmoi cloud: the common deployment of SCA Components and partners' components into the Cloud allow a simplified execution of workflows in the same network.

Both aspects are also presented in section 11, where facet linked results are described. This chapter focuses on three main facets to be taken into account to assess the cloud performance: scalability, elasticity and availability. It also describes the improvement of these main facets performance.

Despite these good results, during year 4 Khresmoi must focus on validating this infrastructure in different environments, along with possible refinement, in order to adapt to production deployment needs.

13 References

[1] Martinez Rodriguez, Ivan and Tinte Garcia, Miguel Angel, Deliverable D6.4.2 "Evaluation of the "Early Cloud Infrastructure" and specification refinement for the "Full Cloud infrastructure"" in KHRESMOI Project, Seventh Framework Programme.

[2] Martinez Rodriguez, Ivan and Tinte Garcia, Miguel Angel, Deliverable D6.4.1 "State of the art, concepts and specification for the "Early Cloud infrastructure" in KHRESMOI Project, Seventh Framework Programme.

[3] Martinez Rodriguez, Ivan and Tinte Garcia, Miguel Angel, Deliverable D6.3.2 "Evaluation of the 'Early software architecture' and further specification" in KHRESMOI Project, Seventh Framework Programme.

[4] Jerry Gao, Xiaoying Bai, and Wei-Tek Tsai, "Cloud Testing- Issues, Challenges, Needs and Practice" in Software Engineering: An International Journal (SEIJ), Vol. 1, No. 1, September 2011.

[5] Gurdev Singh and Rakesh Kumar, "Availability Metrics for Cloud Vibrant Behaviour with Benchmarks Influence On Diverse Facets" in International Journal of Software Engineering & Applications (IJSEA), Vol.3, No.1, January 2012

[6] Proko Eljona, Ninka Ilia, "Analysis and Strategy for the Performance Testing in Cloud Computing" in Global Journal of Computer Science and Technology Cloud & Distributed, Vol. 12 Issue 10 Version 1.0 July 2012