CACTOS Toolkit Version 1
Project Deliverable D5.2.1

Henning Groenda, Christian Stier (FZI),
P-O Östberg, Jakub Krzywda (UMU),
James Byrne, Sergej Svorobej (DCU),
Zafeirios Papazachos (THE QUEEN’S UNIVERSI),
Craig Sheridan, Darren Whigham (Flexiant Limited)

Due date: 30/09/2014
Delivery date: 30/09/2014

This project is funded by the European Union under grant agreement no. 610711
## Dissemination Level

<table>
<thead>
<tr>
<th>X</th>
<th>PU</th>
<th>Public</th>
</tr>
</thead>
<tbody>
<tr>
<td>PP</td>
<td>Restricted to other programme participants (including the Commission Services)</td>
<td></td>
</tr>
<tr>
<td>RE</td>
<td>Restricted to a group specified by the consortium (including the Commission Services)</td>
<td></td>
</tr>
<tr>
<td>CO</td>
<td>Confidential, only for members of the consortium (including the Commission Services)</td>
<td></td>
</tr>
</tbody>
</table>
## Version History

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Change</th>
<th>Author</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>04/06/2014</td>
<td>Creation of document</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.2</td>
<td>04/07/2014</td>
<td>Document structure</td>
<td>Christian Stier, Henning Groenda (FZI)</td>
</tr>
<tr>
<td>0.3</td>
<td>24/07/2014</td>
<td>Added core sections and examples for each section</td>
<td>Christian Stier, Henning Groenda (FZI)</td>
</tr>
<tr>
<td>0.4</td>
<td>08/08/2014</td>
<td>Integrated initial versions of sections on individual toolkits written by DCU and QUB</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.5</td>
<td>14/08/2014</td>
<td>Updated open points, added tabular overview on evaluated versioning systems</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.6</td>
<td>14/08/2014</td>
<td>Added the introductory section for the example use case</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.7</td>
<td>18/08/2014</td>
<td>Added CactoSim integration section for the use case example</td>
<td>Sergej Svorobej (DCU)</td>
</tr>
<tr>
<td>0.8</td>
<td>18/08/2014</td>
<td>Added CactoOpt use case and integration descriptions</td>
<td>P-O Östberg (UmU)</td>
</tr>
<tr>
<td>0.81</td>
<td>20/08/2014</td>
<td>Incorporated initial version of section on adapting the infrastructure configuration of the data centre</td>
<td>Craig Sheridan (FLEX)</td>
</tr>
<tr>
<td>0.82</td>
<td>20/08/2014</td>
<td>Refined CactoSim integration section for the use case example and for CACTOS Tooling Overview</td>
<td>Sergej Svorobej (DCU)</td>
</tr>
<tr>
<td>0.83</td>
<td>21/08/2014</td>
<td>Added and refined the section on OptimisationPlans and triggering of optimisation operations</td>
<td>Jakub Krywda (UmU), Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.84</td>
<td>22/08/2014</td>
<td>Inserted the sections contributed by QUB, consistency corrections</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.85</td>
<td>26/08/2014</td>
<td>Preparation for the internal review: Elaborated on several sections and fixed consistency issues</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.86</td>
<td>26/08/2014</td>
<td>Internal review of document excluding pending partner contributions. Rewrite of many sections.</td>
<td>Henning Groenda (FZI)</td>
</tr>
<tr>
<td>0.87</td>
<td>27/08/2014</td>
<td>Expanded upon sections on CactoScale and the integration with the other tools</td>
<td>Zafeirios Papazachos (QUB)</td>
</tr>
<tr>
<td>0.88</td>
<td>28/08/2014</td>
<td>Further internal review, corrections and expansions of individual sections</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td></td>
<td>29/08/2014</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.9</td>
<td>29/08/2014</td>
<td>WP-internal final review</td>
<td>Henning Groenda (FZI)</td>
</tr>
<tr>
<td>0.91</td>
<td>09/09/2014</td>
<td>First internal review</td>
<td>Sergej Svorobej (DCU)</td>
</tr>
<tr>
<td>0.92</td>
<td>11/09/2014</td>
<td>Expanded upon CactoScale sections as requested in the review</td>
<td>Zafeirios Papazachos (QUB)</td>
</tr>
<tr>
<td>0.93</td>
<td>12/09/2014</td>
<td>Addressed comments from first internal review</td>
<td>Christian Stier (FZI)</td>
</tr>
<tr>
<td>0.94</td>
<td>22/09/2014</td>
<td>Second internal review</td>
<td>Zafeirios Papazachos (QUB)</td>
</tr>
<tr>
<td>1.0</td>
<td>24/09/2014</td>
<td>Final version ready for release</td>
<td>Christian Stier (FZI)</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

In Infrastructure as a Service (IaaS) cloud data centres, customers can run their software on the virtualised infrastructure of a data centre. They benefit from easy scalability and pay-as-you-go payment models and are able to request Virtual Machines (VM) with varying properties, such as processing speed or memory size. Data centre providers benefit from consolidation and economy of scales effects if several VMs are deployed on the same physical resources without Quality of Service (QoS) conflicts, e.g. because VMs often idle and rarely use all available resources. The efficient utilisation of the underlying physical infrastructure including management and topology optimisation determines the costs and ultimately the business success for data centre operators.

The CACTOS project develops an integrated solution for runtime monitoring, optimisation and predictions. The solution supports data centre providers in managing and planning data centres. CACTOS consists of two toolkits:

- The **CACTOS Runtime Toolkit** facilitates automated resource planning and optimisation for Infrastructure as a Service (IaaS) data centres.
- The **CACTOS Prediction Toolkit** enables what-if analyses including effects caused by automated resource optimisation based on existing or planned data centre topologies.

This document provides an overview on both toolkits and their interactions in the completed first iteration step. The focus of this deliverable is on describing the CACTOS Runtime Toolkit but was extended to give a holistic view and cover the CACTOS Prediction Toolkit as well. The CACTOS Runtime toolkit consists of independent tools for cloud infrastructure analytics and optimisation. This document describes the purpose and features of the tools as well as utilised base technology and provided interfaces. The analytics-oriented tool, CactoScale, provides already an automated extraction of central infrastructure information and monitoring of a running data centre. The optimisation-oriented tool, CactoOpt, can perform optimisation operations on the basis of the extracted information. However, the execution of optimisation operations on cloud middleware requires manual effort.

This document describes the provisioning of both toolkits within a data centre and enables testing and running the approach in an own data centre. Exemplary use cases show the applicability and how important tasks are realized in the toolkit.

This document delineates how the tools that were developed as part of the individual deliverables for CactoOpt (D3.1 Prototype Optimization Model), for CactoScale (D4.1 Data Collection Framework) and CactoSim (D6.1 CactoSim Simulation Framework Initial Prototype) are integrated into the CACTOS Runtime Toolkit and the CACTOS Prediction Toolkit. Based upon the integration implementation presented in this document, (D5.1 Model Integration Method and Supporting Tooling) will outline the integration methodology that is applied in the toolkits discussed by this deliverable. Exemplary use cases presented in this document were motivated by one of the CACTOS testbeds that is outlined in (D7.2.1 Physical Testbed). The current version of the CACTOS Runtime Toolkit requires some manual interaction with the data centre operator to realise the optimisations. As part of the deployment of the CACTOS Runtime Toolkit in a small-scale testbed (D5.3 Operational Small Scale Cloud Testbed Managed by the CACTOS Toolkit) the integration will be fully automated.

In line with the effort of promoting and enabling continued development of the CACTOS toolkits, we released both toolkits under the licensing terms of the Eclipse Public License Version 1. Deliverable (D5.1 Model Integration Method and Supporting Tooling) will contain an in-depth evaluation of different licensing models and the rationale for opting with for the proposed licensing model.

---

**CACTOS**

---

**D5.2.1 CACTOS Toolkit Version 1**
# TABLE OF CONTENTS

**EXECUTIVE SUMMARY**

**TABLE OF CONTENTS**

<table>
<thead>
<tr>
<th>List of Figures</th>
<th>IV</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of Tables</td>
<td>V</td>
</tr>
<tr>
<td>Abbreviations</td>
<td>VI</td>
</tr>
</tbody>
</table>

**I. INTRODUCTION**

**II. CACTOS TOOLING OVERVIEW**

1. **CACTOScale**
   - Purpose and Features
   - Utilised Base Technologies
   - Interfaces for Integration

2. **CACTOOpt**
   - Purpose and Features
   - Utilised Base Technologies
   - Interfaces for Integration

3. **CACTOSim**
   - Purpose and Features
   - Utilised Base Technologies
   - Interfaces for Integration

**III. PROVISIONING OF THE CACTOS TOOLKIT**

1. **CACTOScale**
2. **CACTOOpt**
3. **CACTOSim**

**IV. EXAMPLE USE CASE**

1. **THE CACTOS Runtime Toolkit**
   - Extracting Infrastructure Information
   - Creating Infrastructure Models
   - Triggering Optimisations
   - Adapting the Infrastructure Configuration of the Data Centre
2. **THE CACTOS Prediction Toolkit**
LIST OF FIGURES

FIGURE 1 OVERVIEW ON THE CACTOS TOOLING LANDSCAPE ........................................................................................................... 2
FIGURE 2 OVERVIEW OF THE CACTOS CLOUD TESTBED DEPLOYED AT THE UNIVERSITY OF ULM ................................................................. 12
FIGURE 3 CACTOScale ARCHITECTURE AND DEPLOYMENT .................................................................................................................. 15
FIGURE 4 FCO NODETOOL STRUCTURAL OUTPUT ........................................................................................................................... 17
FIGURE 5 THE HIERARCHY OF OPTIMISATION PLAN ENTITIES ........................................................................................................... 19
FIGURE 6 THE HIERARCHY OF VERTICAL SCALING ENTITIES ........................................................................................................... 19
LIST OF TABLES

TABLE 1 COMPONENTS PER NODE TYPE IN UULM'S CLOUD TESTBED ................................................................. 13
TABLE 2 LIST OF AVAILABLE CHUKWA TELNET COMMANDS ........................................................................... 15
# Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CACTOS</td>
<td>Context-Aware Cloud Topology Optimisation and Simulation</td>
</tr>
<tr>
<td>CI</td>
<td>Continuous Integration</td>
</tr>
<tr>
<td>COTS</td>
<td>Commercial-Off-The-Shelf</td>
</tr>
<tr>
<td>EDP2</td>
<td>Experiment Data Persistency &amp; Presentation</td>
</tr>
<tr>
<td>FCO</td>
<td>Flexiant Cloud Orchestrator</td>
</tr>
<tr>
<td>FDL</td>
<td>Flexiant Development Language</td>
</tr>
<tr>
<td>rpm</td>
<td>revolutions per minute</td>
</tr>
<tr>
<td>SLA</td>
<td>Service Level Agreement</td>
</tr>
<tr>
<td>SVN</td>
<td>Apache Subversion</td>
</tr>
<tr>
<td>OSP</td>
<td>OpenSourceProjects.eu</td>
</tr>
<tr>
<td>QoS</td>
<td>Quality of Service</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
</tbody>
</table>
I. INTRODUCTION

In Infrastructure as a Service (IaaS) cloud data centres, customers can run their software on the virtualised infrastructure of a data centre. They benefit from easy scalability and pay-as-you-go payment models and are able to request Virtual Machines (VM) with varying properties, such as processing speed or memory size. Data centre providers benefit from consolidation and economy of scales effects if several VMs are deployed on the same physical resources without breaking any Service Level Agreements (SLAs). This is possible as VMs often stay idle and rarely use all available resources. The efficient utilisation of the underlying physical infrastructure including management and topology optimisation determines the costs and ultimately the business success for data centre operators.

The CACTOS Toolkit Version 1 described in this document provides the foundation for such an automated infrastructure management. This document provides an overview on the toolkit and its structure including the composed tools CactoScale, CactoOpt and CactoSim. This document describes the purpose and main features of the tools as well as utilised base technology and provided interfaces. It provides a guideline on provisioning the toolkit in a data centre and illustrates its use with examples.

The current state of the art requires data centre operators to either manually assign VMs submitted by customers to physical hosts, or let the cloud middleware handle the assignment. However, current cloud middleware solutions only support very simplistic assignment policies. They neglect to act upon arising Quality of Service (QoS) conflicts due to the usage profile of mutual deployed VMs. Usually there is no autonomic optimisation process in place. Instead, the data centre operator manually needs to reassign affected VMs to different hosts. The CACTOS Toolkit addresses this issue by integrating optimisation algorithms for the initial placement, resource assignment and migration of VMs with the middleware of the cloud data centre. Data centre operators benefit from this automation since it reduces the involvement of personnel in the active management of data centre resources. Cloud customers profit from the decreased cost of operation and a faster elimination of insufficient QoS.

This document is structured as follows. Chapter II gives an overview on the structure of the CACTOS Toolkit Version 1 and the main parts CACTOS Runtime Toolkit and the CACTOS Prediction Toolkit. It describes their fundamental features and interactions. Chapter III elaborates on the provisioning of the individual tools of the toolkit and shows what is necessary to set them up. In chapter IV, an example use case for the CACTOS Runtime toolkit is given in order to outline its current functionality and motivate future extensions. Further, the chapter sketches functionality and interactions of the individual tools that compose the CACTOS Prediction Toolkit. An outline of interactions between the CACTOS Runtime Toolkit and CACTOS Prediction Toolkit concludes the chapter.
II. CACTOS TOOLING OVERVIEW

This section provides an overview of the CACTOS tooling. Figure 1 provides a graphical illustration of the relation of the individual CACTOS tools. CACTOS brings together analysis, optimisation and simulation of large-scale data centres, enabling data centre operators to use their infrastructure in a more efficient manner. CACTOS consists of the CACTOS Runtime Toolkit for managing real-world data centres (purple area) and the CACTOS Prediction Toolkit for simulating and analysing data centres and data centre configurations (green area). These toolkits consists of a combination of the tools: CactoScale for capturing information in real-world data centres, CactoOpt for optimising the topology of data centres, and CactoSim for simulating data centres plus a Virtualisation Middleware Integration layer which decouples technology-specific data centre cloud computing platforms, e.g. OpenStack or FCO. The CACTOS Runtime Toolkit consists of CactoScale and CactoOpt, the CACTOS Prediction Toolkit of CactoSim and CactoOpt. The three main tools CactoScale, CactoSim and CactoScale have their respective functionality described in more detail in the following.

The CACTOS Toolkit implements an automated cyclical process for the optimisation of data centres. First, CactoScale extracts information on the physical and virtual infrastructure, as well as current load information from the managed data centre. It persists this information in instances of the CACTOS Cloud Infrastructure Models. Second, CactoOpt analyses these model instances to derive a set of suggested optimisation actions. CactoOpt condenses these actions into an optimisation plan. This optimisation plan is then passed on to the Virtualisation Middleware Integration component of the CACTOS Toolkit. The Virtualisation Middleware Integration Tooling...
triggers calls on the actual cloud data centre middleware in order to carry out the optimisation actions. This process is repeated cyclically based on a configurable optimisation rate.

CactoScale collects information on the physical properties of a data centre. This includes information on the physical configuration of the infrastructure, such as the nodes and the hardware placed in each node, as well as the logical configuration of the infrastructure, such as the placement of virtual machines (VMs) on physical nodes. Additionally, CactoScale measures the load on the physical as well as the virtualised resources and regularly updates the Infrastructure Models accordingly. To summarise, CactoScale is responsible for creating and updating instances of the Infrastructure Models, namely the Physical Data Centre Model, Logical Data Centre Model, Physical Load Model and Logical Load Model.

The optimisation algorithms are developed as part of CACTOS’ CactoOpt infrastructure topology optimisation tooling. CactoOpt suggests optimisation actions based on monitoring data collected by CactoScale. Information on the current state of the data centre is persisted and communicated through instances of the Infrastructure Models. CactoOpt uses the Infrastructure Model instances as input for creating optimisation plans. Suggested optimisation plans can then be executed either in the real data centre managed by the CACTOS Runtime Toolkit or a data centre simulated by the CACTOS Prediction Toolkit. Examples for optimisations on the infrastructure are VM migrations or an increase of resources assigned to a virtual machine. CactoOpt creates an Optimisation Plan Model describing the order of all optimisation actions. The Virtualisation Middleware Integration can execute such a plan on the cloud computing platform, e.g. OpenStack or Flexiant Cloud Orchestrator (FCO).

CactoSim allows simulating whole data centres including the effect of employed optimisation algorithms. Infrastructure Models provide the input for such a simulation. Models created for real-world data centres can be used directly as input, be modified according to potential change scenarios, or purely synthetic models can be created. For example, this allows a data centre operator to estimate the QoS benefit of an expansion of the data centre hardware. CactoSim simulates such a data centre including the system load internally and provides load measurements in the same way as CactoScale. The simulated models are used by CactoOpt for creating optimisation plans and are executed on the simulated data centre using the CactoSim Integration instead of the real one. CactoSim identifies the current Physical and Logical configurations in the simulated data centre and updates the simulated models accordingly. The updated models are then passed to CactoOpt to perform optimisations for the simulated data centre. Finally, CactoSim executes the Optimisation Plan proposed by CactoOpt in the simulation.

1. **CACTOScale**

CactoScale serves multiple data capturing roles in the CACTOS project. It provides a scalable data collection framework for monitoring cloud computing data centres. Furthermore, it is designed to perform online data analytics of events and conditions within a data centre. These events will be utilised as triggers for data centre optimisations in future iterations of the CACTOS Runtime Toolkit.

CactoScale also provides offline analysis functions. In future iterations, behaviour models for virtual machines or running systems based on measurements and log data will be generated as part of the offline analysis. This section provides a quick overview on CactoScale, more detailed information is available in (D4.1 Data Collection Framework) and (D4.2 Preliminary offline trace analysis).
**a) PURPOSE AND FEATURES**

CactoScale features a data collection framework capable of monitoring large scale distributed systems and cloud computing data centres. The data collection framework is coupled with a data analytics tool that allows parallel processing of the data using the MapReduce framework. Fundamental capabilities of the toolkit include:

- Scalable data collection – The data collection framework relies on an agent-based monitoring facility and historic data is stored on a Hadoop Distributed File System which provides scalability and robustness.
- Data analytics for “Big Data” – CactoScale utilises the MapReduce framework and therefore enables parallel processing of the collected data.
- Monitoring of cloud systems and alerting – Different metrics, log files and error logs are monitored and collected. Performance analysis and anomaly detection can be carried out based on historical analysis of these metrics and logs. An alerting mechanism will allow for efficient operation and quick reaction on the data centre operator side and can serve as a trigger for optimisations in future versions of the CACTOS Runtime Toolkit.
- Filtering, clustering and correlation algorithms - Focusing on effective algorithms to achieve correlation and co-analysis of traces from different sources, e.g. to create behaviour models in the offline case.

This release of CactoScale within the CACTOS Toolkit Version 1 includes:

- Monitoring of metrics including network, CPU, memory, storage and virtualised metrics
- Customised import modules for real-world traces of the MolPro scientific application provided by UULM and the black-box virtual machines experienced at Flexiant
- Offline statistical analysis tool for imported traces

**b) UTILISED BASE TECHNOLOGIES**

CactoScale is utilising a range of existing Apache software tools - Chukwa, Hadoop, HBase and Pig. Chukwa is a data collection tool for monitoring of distributed systems, which is built on top of Hadoop Distributed File System and MapReduce framework. Chukwa is extended in order to collect data for the VMs. The design of Chukwa allows us to use the agent modules to invoke a variety of low-level monitoring tools such as sar, iostat, df and top (Sysstat). These tools are used to report information on CPU, disk, network and memory performance indicators. The functionality of these tools is further described in (D4.1 Data Collection Framework). The virt-Top tool is utilised in order to retrieve information regarding the VMs. virt-Top is part of the popular libvirt toolkit. libvirt supports a great number of hypervisors including the

- KVM/QEMU Linux hypervisor
- Xen hypervisor on Linux and Solaris hosts
- LXC Linux container system
- OpenVZ Linux container system
- User Mode Linux paravirtualized kernels
- VirtualBox hypervisor
- VMware ESX and GSX, Workstation and Player hypervisors
- Microsoft Hyper-V hypervisor
- IBM PowerVM hypervisor
- Parallels hypervisor
- Bhyve hypervisor
libvirt is also able to identify and monitor virtual networks built upon bridging, NAT, VEPA and VN-LINK techniques. In addition, it can be used to extract storage information regarding the utilised IDE/SCSI/USB disks, FibreChannel-, LVM-, iSCSI-, NFS-based storage networks and filesystems.

HBase is a NoSQL database modelled after Google's Bigtable, which provides random, real-time read/write access to Big Data. HBase comprises a set of tables. Pig, which is the last of the aforementioned tools, is a high-level platform for creating MapReduce programs used with Hadoop. CactoScale interfaces with the other CACTOS tools by a java-based infrastructure and the Infrastructure Models.

c) Interfaces for Integration

The Infrastructure Models created or updated by CactoScale are stored in a Runtime Model Storage using EMF CDO technology. See section III for information on provisioning the repository.

The identification of VM configurations and virtualisation layer is currently carried out directly using libvirt. It is planned to create an interface for identifying the logical configuration and providing this interface via the Virtualisation Middleware Integration as shown in the overview picture.

2. CactoOpt

CactoOpt is composed of a set of optimisation algorithms and tools designed to allow for data centre operator control, optimisation, and – together with CactoSim – evaluation of data centre infrastructure optimisation. Optimisation mechanisms evaluate data centre properties, e.g., virtual machine deployments and configurations, and attempt to (by ways of evaluation heuristics and optimisation computations) find ways to more efficiently (according to a given objective function) configure and execute workloads in the data centre.

This section provides a quick overview on CactoOpt, more detailed information is available in (D3.1 Prototype Optimization Model).

a) Purpose and Features

The purpose of CactoOpt is to facilitate the optimisation of data centre infrastructures with respect to the efficient provisioning of computational and storage resources to virtual machines in the context of IaaS-based cloud data centres.

CactoOpt is implemented in Java. CactoOpt follows a sensor-actuator model where infrastructure topology and load models capture a sensor view of the surrounding world (i.e. the data centre state) and CactoOpt optimisation recommendations are viewed as actuators for the optimisation core. Actuators cause an adaptation to certain observations made on the sensors. An example for this would be the redeployment of virtual machines due to high host utilisation. Heuristic functions and optimisation algorithms are developed based on this model and Optimisation Plan Models are created accordingly.

Planned and supported features of CactoOpt include the ability to

- Plan for optimised deployment of virtual machines
- Perform heuristics-based evaluation of current deployments of virtual machines
- Continuously identify opportunities for improvements in virtual machine deployment layout
- Plan for migration of virtual machines within and between clusters in data centres
- Plan for optimised dynamic reconfiguration for both virtual and physical resources within the data centre, e.g. by scaling or dynamic frequency regulation
• Plan for optimised admission control and horizontal scaling of virtual machines for cloud applications.

In the CACTOS Toolkit Version 1 release, CactoOpt supports algorithms for initial placement and scaling of virtual machines. There are built-in heuristic functions for evaluating the placement efficiency and comparison of optimisation actions. This version focuses on virtual machine state and does not consider holistic effects of individual optimisation actions on the entire data centre. Optimisation algorithms are developed in prototypical (simulated) environments and (in this version) focused on simple, greedy functions (e.g., finding the node that best matches a certain resource capacity description) and will be further refined in later versions. Further optimisation and heuristics functions will support more advanced modelling of application behaviour, virtual and physical machine load, and the prediction of application behaviour.

The following subsections outline the aforementioned capabilities and features of CactoOpt in greater detail.

**PLANNING THE OPTIMISED DEPLOYMENT OF VIRTUAL MACHINES**

CactoOpt inspects the infrastructure information in order to determine the optimal placement of arriving virtual machines. The infrastructure information includes the current state of the physical and virtualised infrastructure and load measurements on both virtual and physical resources. Optimisations are performed on the basis of a high-level objective function such as the maximization of efficiency with which a resource (e.g., memory) is used, the minimization of overall power consumption, or the maximization of server consolidation (e.g., the number of virtual machines per physical machine).

**PERFORMING HEURISTICS-BASED EVALUATIONS OF CURRENT VM DEPLOYMENTS**

CactoOpt defines heuristic functions that express the utility of arbitrary virtual machine deployments. One potential instantiation of the function is a summarized cost function for the load virtual machines place on local network segments or a predicted energy consumption cost on virtual machine, node or rack level. Deployment evaluation cost functions are then used to evaluate and compare current deployment plans against alternative plans from simulation or optimisation plans.

**OPTIMISING VM DEPLOYMENT AND MIGRATION PLANNING**

CactoOpt identifies opportunities for improvements in virtual machine deployment layouts. Based on the identified optimisation operations, a plan for migrations of virtual machine within and between clusters in data centres is set up. Key to this ability is the definition of cost functions that accurately model the cost of migration of virtual machines (not only in direct transfer costs or performance delays, but also in resulting network load and potential impact on co-located / neighbouring machines and workloads). Important use cases for this ability include not only the use of virtual machine migration as an actuator in optimisation, but also in interactive use cases where data centre operators may want to weigh the potential gains of migrating a workload against the risks and costs associated with the action. Risks in this context are cost that might occur under certain circumstances, e.g. when an initially suitable node that a VM is redeployed to suddenly becomes over-utilised during the redeployment.

**ENABLING THE DYNAMIC RECONFIGURATION OF LOGICAL AND PHYSICAL RESOURCES WITHIN THE DATA CENTRE**

In future releases of CactoOpt, the virtual resources of VMs as well as the state of physical resources may be changed to dynamically adapt to changes in incoming request rates to servers. CactoOpt will use information on planned or predicted peaks in workload to determine when such a reassignment makes sense. An example for such a logical reconfiguration is the increase of RAM and CPU cores assigned to VMs in anticipation of a load peak.
A reconfiguration of physical hardware may consist of an adjustment of CPU frequencies or power state in a
similar manner. This allows increasing the energy efficiency during periods of low CPU utilisation.

OPTIMISED ADMISSION CONTROL AND SCALING OF VIRTUAL MACHINES FOR CLOUD APPLICATIONS
This ability can be used differently in the usage context of the CACTOS Runtime Toolkit as well as the CACTOS
Prediction Toolkit. For the runtime data centre environment it supports automatic adjustment of the number of
virtual machines spawned for a particular cloud application. In predictions, it additionally serves to evaluate
alternative deployment plans for virtual machines and to predict the impact changes in the infrastructure and
usage profile have on the QoS of applications deployed in the data centre. The key to this ability is the formulation
of application workload models and load prediction techniques that can capture workload fluctuations with a
sufficiently high accuracy. Thereby, the costs of virtual machine instantiations can be weighed against the costs of
potential service-level agreement violations that would result from delays.

b) Utilised Base Technologies
CactoOpt is realized in Java and stores application and workload data in a MySQL database by default. Future
optimisation algorithms can be developed in proprietary environments, e.g., Matlab and customised C-based
solver environments. In such a case, they are wrapped in Java using Java-native bridging technologies such as the
Java Native Interface (JNI). The optimisation functions will be provided as OSGI / Equinox bundles in order to
facilitate the use within both CACTOS toolkits.

c) Interfaces for Integration
CactoOpt uses the Infrastructure Models in the Runtime Model Storage (if used within the CACTOS Runtime
Toolkit) or the ones provided by CactoSim (if used within the CACTOS Prediction Toolkit) in order to create
Optimisation Plan Models. The recommendations contained in such a plan can then be executed using the
Virtualisation Middleware Integration. The Optimisation Plan Model and Infrastructure Models are described in
detail in (D3.1 Prototype Optimization Model) and presented in context with the integration methodology in the
upcoming (D5.1 Model Integration Method and Supporting Tooling).

3. CactoSim
The aim of CactoSim as model-driven discrete event simulation is to produce accurate system behaviour forecasts,
which are essential for accurately planning and managing a data centre. By using CactoSim, experimenters are
able to validate and evaluate system configuration scenarios and obtain valuable decision support information.

This section provides a quick overview on CactoSim, more detailed information is available in (D6.1 CactoSim
Simulation Framework Initial Prototype).

a) Purpose and Features
CactoSim is a context-aware cloud topology simulation framework implemented as an Eclipse plugin. In an IaaS
scenario from a data centre operator point of view, fine-grained knowledge of system internals is usually not
available. The data centre operators do not have access to detailed information of applications that are deployed
in VMs. Thus, CactoSim abstracts from these internals and utilises a higher-level black box behaviour model.
CactoSim addresses both physical and virtualised resources; the mapping of VMs to physical resources and the
hierarchy of racks and nodes are essential characteristics of IaaS systems that are captured by the CACTOS Cloud
Infrastructure models.
The version of CactoSim released as part of the CACTOS Toolkit Version 1 provides prediction capabilities in the cloud context. The QoS predictions consider access to processor, storage and memory. In order to represent system workload for a variety of applications running in the virtual landscape we also introduce black box and grey box behaviour modelling practices. Thereby, a resource utilisation prediction is enabled without detailed software component model knowledge.

CactoSim plays a major role in the evaluation and validation of the optimisation algorithms used in CactoOpt. CactoSim uses CactoOpt and takes the recommended Optimisation Plans into account, modifying the topology. This enables a comparison of different optimisation algorithms without deploying them in a physical testbed environment.

Features available in this release are:

- **Grey box and black box behaviour models** – alternative high-level behaviour and performance prediction models for VMs where little or no knowledge on the internal behaviour of hosted applications is available.
- **Memory model** – representation of physical node memory and its throughput.

The additional features of CactoSim in its final version will be:

- **Deployment and behaviour modelling** – allows modelling the structure of IaaS cloud environments, including the hardware bound deployment and behaviour models from which predictions on the resource demands of VMs can be derived.
- **Support for self-adapting systems** – extends considering optimisation and even allows to take into account modifications on individually deployed applications within the data centre to represent reactions to system conditions i.e. QoS violations or workload changes.
- **Resource utilisation reports** – the log of the simulated system resource utilisation can be accessed and analysed right after the simulation.

**b) Utilised Base Technologies**

CactoSim is built on the two existing tools Palladio (The Palladio component model for model-driven performance prediction) and SimuLizar (Performance analysis of self-adaptive systems for requirements validation at design-time).

Palladio is a component-based architecture simulator performing model-driven QoS predictions for component-based software systems defined in the Palladio Component Model (PCM). The focus of the PCM lies on reasoning on non-functional properties of systems in the design stage and selecting the best architecture alternative. Consequently, software architects and component developers designing the architecture of the system are assumed to have insight into the assembly of and dependencies between individual software components. They model individual services and their behaviour at a white box level. Palladio models physical resources and the virtualisation layer in a very abstract manner. CactoSim will further extend Palladio to support the modelling of VMs that are deployed into the system during the simulation. Additionally, a model for predicting the performance impact of memory is added to Palladio on top of its existing models for CPU and HDD access.
SimuLizar is based on PCM but extends it for self-adaptive systems. Its ability to create self-adaptation rules will be utilised and extended as by CactoSim. This enables the realisation of the Virtualisation Middleware Integration with CactoSim and therefore the execution of optimisation plans.

**c) Interfaces for Integration**

The significant strength of CactoSim in comparison to other cloud simulation tools comes from the integration with the CACTOS Runtime toolkit. The CACTOS project enforces common standard between CactoScale, CactoOpt and CactoSim components by utilising meta-models specifically tailored for the operation of IaaS cloud data centres. Having these common standards in place across toolkits ensures that all toolkits can share information using a common abstraction.

In this release CactoSim is able to fetch the Infrastructure Models from real-word data centres located in the Runtime Model Storage. These models are then stored in the Prediction Model Storage for further modification or analyses.

The Prediction Model Storage allows to store and access different versions of a data centre model allowing version tagging, e.g. that it is initially populated from a real data centre, extended by hardware nodes. This supports the lightweight comparison of different data centre or optimisation algorithm alternatives.

During a simulation run, CactoSim keeps the Infrastructure Models with a minimal memory footprint in the main memory of the node executing the simulation. These models are directly passed to CactoOpt and the Virtualisation Middleware Integration. The Virtualisation Middleware Integration uses the VMI CactoSim adapter to modify these models according to the selected optimisation plan. In this release, the suggested optimisation plan has to be transformed and manually mapped to adaptation actions to the underlying CactoSim internal simulation models based on plain PCM. This step will be automated in future releases, starting with the release for (D5.3 Operational Small Scale Cloud Testbed Managed by the CACTOS Toolkit).

The results of a simulation are stored using the persistency framework EDP2 that is developed as part of the Palladio tooling. EDP2 persists measurement results with their unit and information on the measurement context. The files containing the results are persisted locally on the machine executing the simulation. Recorded results can be analysed using distribution and time-series plots as part of the Eclipse-based Palladio tools. Furthermore, the results can be extracted as comma-separated tables for an analysis in statistical tools, storage in databases or an analysis in other graphical user interfaces.
III. Provisioning of the CACTOS Toolkit

This chapter describes where the individual components of both the CACTOS Runtime Toolkit and the Prediction Toolkit can be retrieved. It also outlines how the components have to be configured so that they can be used as part of the respective toolkits. Full instructions on an integrated setup of the tools in a data centre will be provided as part of (D5.3 Operational Small Scale Cloud Testbed Managed by the CACTOS Toolkit).

1. CactoScale

CactoScale is composed of different components, which need to be provisioned in a data centre. The first component is Apache Hadoop which can be found on the official project website (Apache Hadoop). Apache Hadoop is an open-source software framework for storage and large-scale processing of data-sets on clusters of commodity hardware. The installation is explained in the step-by-step instructions on the official website (Hadoop MapReduce Next Generation - Cluster Setup). Most of the installation steps involve setting up the configuration files for Hadoop. A sample of the configuration of these files can be found on CACTOS SVN repository. The steps to set up a Hadoop cluster are the following:

1. Prerequisites for Hadoop need to be set up and installed on the cluster nodes. This includes a working Java installation and configuring ssh access to the nodes.
2. The configuration file hadoop-env.sh needs to be configured on each of the nodes by setting the path of the Java installation to use.
3. Set the conf/core-site.xml configuration options. Here the host of the NameNode, which is a master service of Hadoop, is defined.
4. Set the conf/mapred-site.xml configuration options. This file is used to set the host and port that the MapReduce job tracker runs at. Optionally, we can set the number of map and reduce slots assigned to the node.
5. In the file conf/mapred-site.xml, configure the number of data replications that Hadoop creates. This allows for Hadoop to recover data in case one the Hadoop cluster nodes is inaccessible. The number of replication cannot exceed the number of available DataNodes in the system.
6. On the master node (i.e. where the NameNode is going to be running) setup the conf/masters file. This involves adding the nodes in the file where SecondaryNameNodes are running. Further details of the NameNode and secondary NameNodes functionality are given in (D4.1 Data Collection Framework).
7. Set in conf/slaves the nodes that will be running the slave daemons DataNodes and TaskTrackers.
8. Format the HDFS filesystem via the NameNode.
9. Once the above configuration is finished; run the start-all.sh script to start HDFS.

After setting up a Hadoop cluster, the next step is to set up the HBase cluster and the Apache Pig tool. HBase is a NoSQL database modelled after Google’s Bigtable, which provides random, real-time read/write access to Big Data. HBase comprises a set of tables. Pig, which is the last of the aforementioned tools, is a high-level platform for creating MapReduce programs used with Hadoop. Detailed instructions on setting up HBase and Pig are available on the official Hadoop website (Apache Hadoop). Preconfigured files for the CACTOS-specific configuration of HBase are available in the CACTOS SVN repository. The following steps describe the HBase configuration:
1. Configure hbase-site.xml file. The user has to define the location of the node where the Hadoop NameNode service is running. Then, the user has to set the location of the node where the master daemon service for HBase will be running.

2. The nodes where the secondary daemons for HBase run are set in conf/regionervers. The list of nodes set in these files is the same as in conf/slaves of the HDFS configuration.

3. After the configuration has finished, run hbase-start.sh script to start HBase.

The extended CactoScale-version of Chukwa is available in the CACTOS SVN repository. Once the previous steps are completed, the administrator has to configure and set up the agents on the monitored sources and then assign the agents to the available collectors. Custom configuration files are available in CACTOS SVN repository. The most important steps for setting up the tool are listed below:

1. Set the environment variables in conf/chukwa-env.sh.
2. Configure the agents file by adding all nodes that an agent is to be installed.
3. Configure the collectors file by adding the nodes where the collectors are running.
4. Run the start-chukwa script to start the monitoring services.

A collection of all the utilities (Pi scripts, Hadoop and MapReduce programs) geared toward the use in CactoScale is available in the SVN repository.

2. CactoOpt

The CactoOpt optimisation toolkit is developed as a standalone Java application. The latest builds are available from the CI experimental repository, both in automated (daily) builds and manual (scheduled) releases.

3. CactoSim

The CACTOS Prediction Toolkit’s simulation core CactoSim is Eclipse based, meaning it is available as a plugin for the Eclipse ide that is installed like every other plugin. Eclipse supports automatic dependency resolution. This allows the download of missing required software components on the fly, making the whole installation process user friendly.

CactoSim plugins can be installed from two installation sites. The first site is part of the CI experimental repository, which is automatically compiled every day to from the most recent build. This makes the latest advancements in the development available to the user, but may be unstable and contain few bugs. The second site is a more stable repository, which is updated with manual releases based on CactoSim’s feature release schedule. This is the mature and tested build.

In order to install CactoSim, first start an instance of Eclipse. Then select “Install New Software” option via the Help menu and connect to the CactoSim repository location. At this stage Eclipse will connect to the repository and retrieve all the available versions that are available for selection. The installation is completed following simple on-screen instructions.
IV. Example Use Case

This section sketches the feature scope of both the CACTOS Runtime toolkit in its first release, as well as the integration with the Prediction Toolkit using one of the CACTOS testbeds as a running example. The testbed deployed by the University of Ulm is chosen as a running example since it offers a heterogeneous infrastructure at a size where the information flows through the CACTOS Toolkit can still be comprehensively explained in detail.

Figure 2 schematically depicts the infrastructure topology of the UULM cloud testbed. It consists of 16 x86-based compute nodes, a storage node, a cloud controller and a network controller. The compute nodes run scientific computations inside VMs, the cloud controller runs the cloud middleware controller, e.g. to handle VM management operations. Some of the compute nodes do not even have an HDD. Thus, they rely on the dedicated storage nodes to persist data.
A brief overview on the hardware components of the nodes is given in Table 1. For more details please refer to (D7.2.1 Physical Testbed).

The UULM CACTOS cloud testbed is controlled by OpenStack cloud middleware. However, it could also be managed by other cloud middleware solutions such as the Flexiant Cloud Orchestrator. The Cloud Controller node is utilised to manage the 16 compute nodes and perform runtime monitoring. It runs CactoScale for the runtime monitoring of the cluster. The optimisation framework CactoOpt is also executed on the cloud controller node. CactoOpt makes placement decisions for virtual machines by considering the 16 compute nodes as potential hosts. Optimisation operations recommended by CactoOpt are realized through a middleware integration controller. This middleware integration controller triggers deployment and assignment operations on a specific middleware.
CactoOpt, CactoScale and the Virtualisation Middleware Integration controller are built so that they can be hosted in a distributed fashion. However, this isn’t necessary for the UULM testbed due to the limited complexity of the managed infrastructure topology.

Section 1 describes how the automated measurement-optimisation cycle is carried out and coordinated across CactoOpt, CactoScale and the integration tooling. Section 2 outlines the integration of the CACTOS Prediction Toolkit with the CACTOS Runtime Toolkit. The central objective of this integration is facilitating simulations based on measurements on the real infrastructure by the automated collection of simulation models from the runtime tooling.

1. THE CACTOS RUNTIME TOOLKIT

The CACTOS Runtime Toolkit automatically optimises the deployment of virtual machines based on runtime measurements. It consists of CactoScale and CactoOpt. CactoScale monitors the data centre at runtime. It collects infrastructure information and load measurements. CactoOpt is used to optimise the deployment of virtual machines on the data centre.

Section a) outlines how the CACTOS Runtime Toolkit automatically collects information on the physical as well as the virtual infrastructure of the data centre. Subsequently, it is explained how CACTOS Cloud Infrastructure Models are constructed from the collected information. In section b) it is described how optimisations on the infrastructure are triggered. The following section c) sketches how the optimisation operations are carried out in the data centre via middleware actuators.

a) EXTRACTING INFRASTRUCTURE INFORMATION

CactoScale utilises an agent-based monitoring architecture to retrieve load and infrastructure information from the data centre at runtime. Apache Chukwa’s runtime monitoring provides the agents and collectors monitoring mechanisms. Chukwa has been extended in this project to support the monitoring of virtualized environments. A Chukwa agent is running on every node which collects the output of different monitoring utilities. The output is then collected by one of the available collectors. The collector processes the data and stores it in HBase.

CactoScale is also capable of gathering and extracting information from application and error logs. An agent installed in each Virtual Machine is able to collect log files and transmit these data to the collectors for processing. The agents can also be paired with in-situ analytics modules. These in-situ modules perform a pre-analysis and aggregation to enable higher sampling rates for load measurements (e.g. utilisation) or to filter the data that flows to the database for post-processing.
Figure 3 sketches the architecture and deployment of CactoScale. CactoScale provides a scalable framework for collecting and processing data from a cloud data centre. The major components of this tool can be distinguished in two categories:

1. The components placed in the system being monitored
2. Components which belong to the monitoring infrastructure.

The Chukwa agents, which are distributed over the data centre nodes, belong to the first category. A Chukwa agent consists of dynamically loadable modules that run inside the agent process. The modules read out data from monitored data sources. There is generally one module for each data source. These modules are also referred to as adaptors. Possible data sources for the adaptors are file or Unix commands. The behaviour of every agent can be accessed manually and controlled independently by connecting to the specific agent on port 9093 using telnet. There are a number of commands that you can use to inspect and control it. The available commands are shown in Table 2.

<table>
<thead>
<tr>
<th>Command</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>add</td>
<td>Start an adaptor.</td>
</tr>
<tr>
<td>close</td>
<td>Close socket connection to agent.</td>
</tr>
<tr>
<td>help</td>
<td>Display a list of available commands</td>
</tr>
<tr>
<td>list</td>
<td>List currently running adaptors</td>
</tr>
<tr>
<td>reloadcollectors</td>
<td>Re-read list of collectors</td>
</tr>
<tr>
<td>stop</td>
<td>Stop adaptor, abruptly</td>
</tr>
</tbody>
</table>
The agent can utilise a monitoring tool, such as iostat, sar, Virt-Top, cigar, top, ps and Df, by using a module to collect the output. To add a module one has to use the `add` command which has the following form:

```
add [name =] <adaptor_class_name> <datatype> <adaptor specific params> <initial offset>
```

There are four mandatory fields: The `add` command, the class name for the Adaptor, the data type of the Adaptor’s output, and the sequence number for the first byte. There are two optional fields; the adaptor instance name, and the adaptor parameters.

The adaptor name, if specified, should go after the `add` command, and be followed with an equals sign. It should be a string of printable characters, without whitespace or '='. Chukwa Adaptor names all start with ‘adaptor_’. If you specify an adaptor name which does not start with that prefix, it will be added automatically.

The measurements from the agents are sent over HTTP to the Chukwa collectors. The primary role of a collector is to parse the collected data from the agent and store the extracted information to the connected HBase database. The Chukwa Collectors component is deployed in a distributed manner. Thereby it provides a fail-safe solution in case one of the component instances stops responding. Furthermore, adding more Collector Component instances and distributing the aggregation and analysis tasks among them can scale the monitoring.

HBase is an open-source, distributed, versioned NoSQL database, which provides Bigtable-like capabilities on top of Hadoop and HDFS. HBase is a key-value store that works as an ordered map that associates keys to values. Keys are multidimensional and consist of the row-key, column family, column and timestamp. The mapping has the form of:

```
(row-key, column family, column, timestamp) -> value
```

The HBase data model consists of different logical components such as tables, rows, column families, columns, cells and versions. HBase tables consist of rows and columns. Columns are grouped into column families. Rows are identified by a row-key. Row-keys are unique for each table. Column families have one or more columns and each column family is stored in a separate file.

In the context of offline analyses, the information collected in the HBase database is evaluated by the CactoScale Offline Analysis component. As part of the CACTOS Runtime Toolkit, the EMF Instance Creator Component creates and updates a representation of the current infrastructure and load in instances of the Infrastructure Models. The EMF Instance Creator Component stores these models in the connected Runtime Model Storage component. The Runtime Model Storage is responsible for persisting the current system state in CACTOS Cloud Infrastructure Model instances. This includes updating the Logical Data Centre Model with changes in the virtual topology and both Physical and Logical Load Models with recent utilisation metrics. The Physical Infrastructure Model isn’t updated regularly since it doesn’t change frequently. Rather, it is extracted as part of a manually triggered API call.
OPENSTACK INFRASTRUCTURE INFORMATION
The extended Chuckwa agents provide access to the required measurements within the OpenStack infrastructure.

FCO INFRASTRUCTURE INFORMATION
In order to provide load and infrastructure information to CactoScale, Flexiant has developed scripts to extract node-level information. These scripts run from the platform management box, which polls each node as well as querying a database to compile this data. This information is then collected and stored for access by CactoScale.

A variety of information is made available such as node load and node IP for use by CactoScale. CactoScale shall be able to extract this information from the generated log files within FCO environments.

b) Creating Infrastructure Models
CactoScale provides measurements of the system load and status of the infrastructure by creating and sharing instances of the physical and logical data centre and load models. These instances are stored and accessed via the Runtime Model Repository. The EMF Instance Creator component analyses the data collected in the database (c.f. Figure 3 CactoScale Architecture and Deployment), combines it and updates the Infrastructure Models managed by the connected Runtime Model Storage component. The Runtime Model Storage component is realised as an instance of a CDO Model Repository. It persists EMF-based representations of the Infrastructure Models on a connected database and offers transactional operations for updating and accessing the managed Infrastructure Models. The information exchange between CactoScale, CactoOpt and CactoSim components is done using instances of specifically designed meta-models. They aim at achieving better integration and data exchange amongst the different components of the CACTOS Runtime Toolkit and CACTOS Prediction Toolkit.

The EMF Instance Creator Component periodically creates meta-model instances by invoking the methods `generateLogicalDCModel()`, `generateLogicalLoadModel()` and `generatePhysicalLoadModel()`.

These methods create the CACTOS Cloud Infrastructure Models instances which are stored on the runtime model storage within a transaction through a set of update operations. When the EMF Instance Creator of CactoScale has completed its updates to the Runtime Model Storage, all consecutive transactions and views access the updated model instances.

c) Triggering Optimisations
This section describes how the CactoOpt Infrastructure Optimiser triggers optimisations. A more detailed description of how optimisations are triggered can be found in (D5.1 Model Integration Method and Supporting...
CACTOS infrastructure optimisations are actions performed by the CACTOS toolkit to improve the efficiency of the target infrastructure. From a high-level perspective they entail:

- The process of inspecting the current state of the infrastructure
- Building a set of models to represent the state
- Reasoning and building optimisation plans for the infrastructure, and enact individual actions within the optimisation plans to improve the efficiency of the infrastructure.

The interface between the optimisation core of CactoOpt and the rest of the CACTOS toolkit architecture can be described in terms of a sensor-actuator model. The sensors, which encompass what the optimiser sees of the surrounding world, are the instances of the Infrastructure Models that are stored and updated by CactoScale. The models are briefly described in chapter II. A set of infrastructure model instances contains all the information the optimisation algorithms build their prediction and optimisation models on. The actuators describe how the optimiser can affect the surrounding world using adaptation actions. It consists of a set of optimisation recommendations generated by the optimiser, e.g. actions such as “place virtual machine x on physical machine y” or “migrate virtual machine x to physical machine z”. The optimisation actions, including their internal order and relationships between them, are described in optimisation plans using a set of predefined actions defined in CACTOS’ Optimisation Plan model. Optimisation plans are parsed, interpreted and translated to cloud stack operations by CACTOS’ Virtualisation Middleware Integration components. They are developed within the project and specific to the cloud-stack of a cloud middleware implementation. For simplicity, the interface through which optimisations can be triggered on the CactoOpt optimisation core by the rest of the CACTOS toolkit architecture consists only of a single method:

```csharp
OptimisationPlan generateOptimisationPlan(PhysicalDCModel pdcm,
    LogicalDCModel ldcm, PhysicalLoadModel plm, LogicalLoadModel llm,
    Deadline deadline)
```
Figure 5 The Hierarchy of Optimisation Plan Entities

The result of the `generateOptimisationPlan` method is a list of recommended changes to the logical and physical infrastructure of the data centre. It can be suggested to perform the execution of optimisation actions described by `OptimisationActionStep` sequentially or in parallel, as shown in Figure 5. It is possible to compile an arbitrarily nested sequential and parallel execution plan out of SequentialSteps, ParallelSteps and OptimisationActionSteps. Possible changes include: the initial placement (VMPlacementAction), VM migration (VMMigrationAction), vertical scaling (VerticalScalingAction) and stopping VMs (StopVMAction).

Figure 6 The Hierarchy of Vertical Scaling Entities
For vertical scaling we consider changing an assigned amount of storage (LogicalStorageScalingAction) or memory (LogicalMemoryScalingAction) as well as frequency of both physical (PhysicalFrequencyScalingAction) and virtual CPU (LogicalFrequencyScalingAction), as presented in Figure 6.

The optimisation process of CactoOpt can be triggered in two ways: automatically and manually. Automatic invocation can be triggered by the toolkit itself, e.g., periodically on some configured schedule or in response to some internal or external event such as a hardware failure, an utilisation limit being reached or simply the arrival of a new virtual machine. Manual triggering of CactoOpt is of course also possible, and can be performed by system administrators for example in experimentation in simulated environments (in conjunction with CactoSim) or using administration tools in production environments (e.g., following infrastructure maintenance).

**d) Adapting the Infrastructure Configuration of the Data Centre**

**OpenStack**

OpenStack provides out of the box REST services for managing the infrastructure. Those are used internally by the VMI OpenStack Component in order to execute the Optimisation Plans.

**FCO**

In the Flexiant Cloud Orchestrator (FCO), three types of APIs are available. They either directly manage the cloud for administrative purposes or allow customers to control a partial view of the resources assigned to them. The available interfaces are as follows:

- Customer facing API – allowing end users to orchestrate their own virtual resources.
- Administrative API – allowing you and your resellers suitably segregated access to customer and billing information, creation and deletion of customers, etc.
- Cluster Control API – allowing system administrators to manipulate both physical and virtual resources (for instance taking nodes on and off line). The Flexiant Cloud Orchestrator can be integrated via the use of programmable triggers.

Triggers are functions that allow an action in Flexiant Cloud Orchestrator to initiate a second action. This action can be a call to internal or external APIs. Triggers enable the integration of the FCO middleware with external management software or middleware technologies. FCO Triggers can be initiated before an API call, after an API call or at a scheduled time. The event initiating a trigger can be one of the following:

- Creation of a resource
- Modification of a resource
- Deletion of a resource
- An API call (user or admin API)
- An Exception
- Scheduled on elapsed time
- Various billing events
- Server state change
- Job state change
- Authorisation event

Triggers are integrated with external systems by:
• Reading or writing to local files
• Encoding or decoding a JSON object
• Making an HTTP call
• Making an API call
• Writing to the syslog.

The full documentation on triggers can be found online (Flexiant Cloud Orchestrator Documentation - Triggers).

Triggers can be used to deploy a VM by using the LUA-based Flexiant Developer Language (FDL). As part of its System API the FDL offers methods to manage the data centre resources and reconfigure the virtualisation environment. As is further explained on the web page (Flexiant, 2014), it offers methods to

• Add, remove and modify clusters
• Add, remove and modify storage units
• Add, remove and modify nodes
• Add, remove and modify internetworking VLANs
• Control which virtual machines are running on which nodes.

External calls of the VMI FCO can be mapped to calls on the Flexiant Developer Language called by to add additional VMs when required. VMs can be created from a pre-set image or from an image uploaded by the user.

To configure an existing VM the FDL Server class can be used. Through it the server resources available to the VM can be modified, e.g. the number of virtual CPU cores and available memory. A full list of the available functionality regarding this class can be found on the Flexiant documentation website (Flexiant Cloud Orchestrator Documentation - FDL Server).

2. THE CACTOS PREDICTION TOOLKIT

The CACTOS Prediction Toolkit allows the user to perform experiments for a model abstraction of a data centre infrastructure and its usage profile. The user is able to control every aspect of the experiment by manipulating infrastructure models and their attributes. For example, an experimenter can execute different VM redeployment strategies by using model transformation rules and quickly compare the predicted resource utilisation results. Thereby, the best strategy suited for a desired usage scenario can be identified. Similarly, the simulated system models can be parameterised with different user load patterns to identify possible future resource conflicts.

In the first version release of the CACTOS Prediction Toolkit the focus is on data centre QoS prediction for non-dynamic scenarios. CactoSim can utilise the Physical and Logical Data Centre Models provided by CactoScale together with optimisation algorithms obtained from CactoOpt. This enables the data centre operators to evaluate alternative infrastructure topologies against each other. Ultimately, this eliminates the manual modelling effort necessary to map information on the data centre topology and its usage to a simulative model.

Once CactoScale has populated the CACTOS Cloud Infrastructure Models in the Runtime Model Storage they can be transferred to the Prediction Model Storage, where CactoSim is able to access them. CactoSim allows data centre operators to manipulate models and predict the overall impact of changes to the data centre. All of the models within CACTOS conform to the carefully designed meta-models allowing data exchange between components and deeper integration further into the project.
REFERENCES


CACTOS Consortium. (2014). *D5.1 Model Integration Method and Supporting Tooling*.

CACTOS Consortium. (2014). *D5.3 Operational Small Scale Cloud Testbed Managed by the CACTOS Toolkit*.

CACTOS Consortium. (2014). *D5.3 Operational Small Scale Cloud Testbed Managed by the CACTOS Toolkit*.


