PaaSage

Model Based Cloud Platform Upperware

Deliverable D6.1.3

Initial Requirements

Version: 1.1.1

PROJECT DELIVERABLE

Name, title and organization of the scientific representative of the project's coordinator:
Mr. Tom Williamson Tel: +33 4 9238 5072 Fax: +33 4 92385011
E-mail: **tom.williamson@ercim.eu**
Project website address: http://www.paasage.eu

Project	
Grant Agreement number	317715
Project acronym:	PaaSage
Project title:	Model Based Cloud Platform Upperware
Funding Scheme:	Integrated Project
Date of latest version of Annex I against which the assessment will be made:	30th September 2013
Document	
Period covered:	M16
Deliverable number:	D6.1.3
Deliverable title	Initial Requirements (Extended)
Contractual Date of Delivery:	31/1/ 2014
Actual Date of Delivery:	25/11/2013
Editor (s):	Stefan Spahr (LSY)
Author (s):	Maciej Malawski (AGH), Bartosz Baliś (AGH), Dariusz Król (AGH), Achilleas Achilleos (UCY), Christos Mettouris (UCY), Avgoustinos Costantinides (IBSAC)
Reviewer (s):	Tom Williamson (ERCIM), Pierre Guisset (ERCIM), Brian Matthews (STFC)
Participant(s):	AGH, UCY, IBSAC
Work package no.:	WP6
Work package title:	Requirements, Integration and Testing
Work package leader:	Stephane Mouton (CETIC)
Distribution:	PU
Version/Revision:	1.1.1
Draft/Final:	Final
Total number of pages (including cover):	32

DISCLAIMER

This document contains description of the PaaSage project work and findings.
The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any responsibility for actions that might occur as a result of using its content.
This publication has been produced with the assistance of the European Union. The content of this publication is the sole responsibility of the PaaSage consortium and can in no way be taken to reflect the views of the European Union.

The European Union is established in accordance with the Treaty on European Union (Maastricht). There are currently 28 Member States of the Union. It is based on the European Communities and the member states cooperation in the fields of Common Foreign and Security Policy and Justice and Home Affairs. The five main institutions of the European Union are the European Parliament, the Council of Ministers, the European Commission, the Court of Justice and the Court of Auditors. (http://europa.eu)

PaaSage is a project funded in part by the European Union.

Executive summary

This document extends D6.1.1 of the original PaaSage project by describing objectives, requirements and scenarios for the extended eScience case study by AGH and new financial sector by IBSAC, as introduced in the enlarged PaaSage project.

The purpose of the document is to describe how the large-size scientific workflows, data farming experiments and financial sector applications will fit into and benefit from the PaaSage framework. The description assumes the current PaaSage architecture based on the deliverables of month 6 and 12 of the project, and presents the case studies in this context.

The large-scale scientific workflows will be based on the HyperFlow workflow execution engine developed by AGH. The planner module of the workflow system will enable to plan workflow execution and prepare an application description in CAMEL together with elasticity rules that will be used by exiting PaaSage components. The application-specific events generated by the running workflow will trigger these autoscaling rules, thus enabling enforcement of the provisioning plan by using the PaaSage platform.

The data farming experiments will use the Scalarm, a massively self-scalable platform, which supports all phases of these experiments, using the master-worker design pattern. Scalarm will benefit from the PaaSage platform by the possibility to automatically generate the deployment plan and scaling rules based on the application requirements, as well as by using the multi-cloud infrastructure.

The financial sector application will benefit from the multi-cloud methodology and model-driven tools offered by PaaSage, by porting the Windows application to the cloud while preserving key aspects that are crucial for auditing firms, their staff and their clients. These aspects are mainly data security and data confidentiality, rapid elasticity and avoiding cloud vendor lock-in and thus data lock-in. This could be facilitated as follows: the PaaSage Upperware should allow porting the legacy application by defining and designing CAMEL models that capture the requirements and allow configuring the application to be ported to the full-spectrum of the Clouds, while the ExecutionWare should provide platform-specific mapping and technical integration to the APIs of the execution infrastructure of the Cloud providers; e.g. perhaps porting application modules to a public cloud, while porting different copies of the data (e.g. database) on private clouds for added security.

In the document, we describe the current status of the use case, and then we describe how the new applications and tools will fit into the PaaSage architecture. Finally, we provide a step-by-step walkthrough PaaSage platform using example deployments and scaling rules.

Contents

List of Figures

1 Introduction

The main goal of this document is to describe the requirements coming from the extended PaaSage project, namely the extended eScience case study. It follows the structure of the D6.1.1 Deliverable that described the requirements of the original PaaSage applications. Since this document is prepared after the Month 12 of the project, it reflects also the information from the Deliverables D1.6.1 (Initial Architecture Design) and D2.1.1 (CloudML Guide and Assessment Report) where the main components and languages used by PaaSage platform are defined. Therefore, the description of the use cases is more technical where possible, so that it allows showing how the extended use cases fit into the current PaaSage architecture.

The extended eScience case study provided by AGH includes large-scale scientific workflows and massively-scalable data farming experiments. We describe the organizational background of the university and its involvement in the eScience related projects, and then give overview of the solutions that are used to support these applications. The tools are HyperFlow workflow execution engine and Scalarm platform for data framing experiments. We describe how both of them fit into the PaaSage architecture and how they will benefit from the PaaSage platform.

The financial auditing use case captured in this document involves the study and initial requirements analysis performed by UCY and IBSAC. Initially, the motivation behind the definition and realization of this case study are described, followed by an overview of the objectives of the financial use case and the current status of the desktop-based application that is platform-dependent (i.e. Windows) and thus this poses limitations in terms of the functionality but also due to the fact that it can only be used within the context of the auditing company. Finally, we describe the envisioned use of PaaSage technologies to port this application to the full spectrum of the Clouds thus providing added functionality and avoiding in-house, platform specific deployment limitations.

2 Use Case Structure

In this document we follow the same structure as use case descriptions in D6.1.1 deliverable.

2.1 Organisation behind the Case.

This section describes the organization of the company presenting the case. The description relates to the general organization described in the overview report. How are the roles realized in your organization? Where are organizational boundaries? What is the competence or responsibility of the actors in real life? Are there other processes in your organization that overlap or interact with the PaaSage workflow etc.

2.2 Objectives

What is your company doing in general; describe the classes of products or processes which can be improved by using cloud computing in general and especially by using the PaaSage method.

2.3 Current Status (as-is)

Give a description of the current status of the selected case.

2.4 Target Picture (to-be)

Describe the improvement which should be reached by using cloud computing together with the PaaSage method.

2.5 Walkthrough PaaSage Workflow.

In this section a case-specific walk through the general PaaSage workflow is described. Are there any specific requirements and constraints in the context of individual steps? Do you use specific tools or technologies? Which steps are the most important or critical ones? How could the platform make a significant difference compared to today's practices?

3 Extended eScience case study

3.1 Organisation behind the Case.

AGH University of Science and Technology is one of the best Polish technical universities. It educates students in 54 branches of studies, including over 200 specializations run at 16 faculties and employing teaching and research staff of 1 887 persons (including 181 full professors). The scientific activity of AGH employees is reflected by the number of over 1600 yearly publications in Polish and international scientific magazines and about 2000 papers delivered at conferences, out of which about 600 are published in international journals. Within the AGH structure there is the Academic Computer Centre CYFRONET with its powerful supercomputers (classified in first 100 on the list of a top 500 fastest computers in the world), AGH operates also one of the largest and most important nodes of the Polish part of the Internet.

The **Department of Computer Science**, which will host the project, employs teaching and research staff of over 80 people, devoting their research efforts to various IT directions, including scalable distributed systems, cross-domain computations in loosely coupled environments, knowledge management and support for life sciences.

The department has successfully took part/led numerous scientific national and international projects, the most important in the area of distributed computing are: CrossGrid (interactive middleware for scientific computations on Grid), K-WfGrid (ontological modeling of scientific or crisis team workflows, semantic composition, monitoring and execution of workflows) and Int.eu.grid (adaptation of infrastructure to e-Science applications), EU IST projects ViroLab (providing a modern virtual laboratory for HIV-related research and treatment in Europe) and Gredia (secure while easy to adopt collaborative scenario enactment environment for business: media and banking). Department staff has been also participating in the CoreGRID Network of Excellence project in the work package devoted to tools and environments and takes part in the CoreGRID follow-up working groups. Recent projects include development of Common Information Space for EU ICT UrbanFlood, multiscale modeling tools for EU ICT MAPPER project and Atmosphere cloud management platform for EU ICT VPH-Share project. Department staff also actively participates in research tasks in PL-

Grid and PL-Grid Plus projects that supports and develops computing infrastructure for Polish research community.

Research work that is relevant to PaaSage includes solutions for virtualization and management of computing and storage resources, high-level programming tools and environments for e-science and scalable systems in service-oriented architecture.

The Department of Computer Science has a strong tradition of international collaboration. This includes organizing conferences, such as yearly Cracow Grid Workshop series since 2001 and International Conference on Computational Science in 2004 and 2008.

The new building of Department of Computer Science provides not only the excellent working environment for research and teaching, but also operates the latest networking and computing infrastructure. This infrastructure provides excellent environment for doing computer science research, with emphasis on distributed systems and cloud computing. Moreover, close collaboration with ACC Cyfronet AGH gives access to the largest computing resources in Poland, including Zeus cluster and PL-Grid project infrastructure, which will be available for large-scale experiments planned in the project.

3.2 Objectives

Within the scope of research projects, AGH collaborates closely with researchers and application users from the eScience domain, both local and international. The interesting use cases for PaaSage are those that require either large-scale workflow or data farming processing. AGH is either involved directly in supporting these applications on grids and clouds or develops tools that enable and facilitate execution of them on these infrastructures.

Local eScience applications and tools are related mostly to the PL-Grid project users and include:

- Bioinformatics applications, in collaboration with the Jagiellonian University Medical College. They include genetic data analysis (sequence alignment, similarity search) as well as proteomic experiments: protein folding and structural comparison. The infrastructures used for these experiments are clusters, grids and clouds [1][2].
- Investigating potential benefits of data farming application to study complex metallurgical processes including generation of Statistically Similar Representative Volume Element and Digital Material Representation. This research is conducted Faculty of Metals Engineering and Industrial Computer Science AGH [7][8].

International collaborations in eScience domain include:

- Virtual Physiological Human initiative, where the scientific workflows are deployed on the cloud in the scope of VPH-Share project [3]. The workflows mainly use Taverna [16] engine for orchestrating the Atomic Services and a specific plugin for Taverna is developed to dynamically create service instances on the cloud using the Atmosphere [14] cloud platform developed by AGH. Other large-scale workflows that are under development use DataFluo workflow engine [15] developed by University of Amsterdam.
- Multiscale applications from fusion domain developed using workflow tools

and MAPPER framework [4]. The MAPPER project provides tools for running multiscale applications on distributed computing infrastructures. The application from the fusion domain used Kepler [17] workflow system to orchestrate its tasks.

- Collaboration with Pegasus team from University of Southern California for support of scientific workflows on cloud infrastructures [5][6]. This collaboration resulted in algorithms for scheduling and provisioning for workflow ensembles on clouds and cost optimization of applications on cloud infrastructures [18]. One of the important benefits of this collaboration is the experience with scientific workflows that use Pegasus [19] workflow management system and the workflow gallery that contains real and synthetic workflows [20].
- Mission planning support in military applications with data farming within the EDA EUSAS project. In the scope of the project, a novel approach to military training was developed, based on behaviour modelling and multi-agent simulations. At first, soldiers' behaviour was captured during a series of training sessions and tranformed into a set of rules, which was then used during highly realistic agent-based simulations of military missions [9]. The aim of the data farming in the process was to develop a better understanding of soldiers' behaviour and identify potential vulnerabilities. During data farming experiments, numerous agent-based simulations were executed, each with different environmental conditions, e.g. emotional state of civilians involved in a mission. Data generated during the simulations was collected and analyzed to find cases when the selected strategy was wrong, e.g. there were too many casualties. The underlying infrastructure for executing the simulations included private clusters, Grids and Clouds [10][11].

The two main tools that are developed by AGH to support these applications are:

- HyperFlow workflow execution engine that is based on hypermedia paradigm and supports flexible processing models such as data flow, control flow, and includes the support for large-scale scientific workflows which can be described as directed acyclic graphs of tasks [13].
- Scalarm is a massively self-scalable platform for data farming, which supports phases of data farming experiments, starting from parameter space generation, through simulation execution on heterogeneous computational infrastructure, to data collection and exploration [12].

While these eScience applications and supporting tools are in various stages of development and maturity, none of them uses the model based approach for development and deployment on clouds that is proposed within PaaSage. Therefore all of them can benefit from the PaaSage platform.

In the following sections we describe the current status and target picture, showing the improvements that can be reached by using cloud computing together with PaaSage platform. We first describe scientific workflows, and subsequently the data farming experiments.

3.3 Large-scale scientific workflows

3.3.1 Current Status (as-is)

In HyperFlow, a workflow is simply a set of **processes** connected through **ports** and exchanging **signals**. The basic abstraction for workflows, a **process,** is defined by:

- **Input ports** and associated **signals** which arrive at the process.
- **Output ports** and associated **signals** which are emitted by the process.
- **Function** invoked from the process which transforms input signals to output signals.
- **Type** of the process which determines its general behavior. For example, a *dataflow* process waits for *all* data inputs, invokes the function, and emits all data outputs. A *parallel-foreach* process, in turn, waits for *any* data input, invokes the function, and emits the respective data output.

A **workflow** is simply a set of processes connected through ports. As a simple example, consider a workflow which computes a sum of squares of several numbers, depicted in Figure 1.

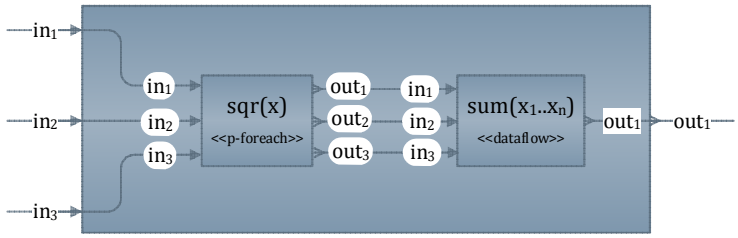

Figure 1: A simple HyperFlow workflow computing a sum of squares of three numbers.

Note that selected input and output signals of some processes are mapped as inputs and outputs of the whole workflow. Consequently, a workflow may act as a process in another workflow. The specification of this workflow in the format of HyperFlow is presented in Figure 2. This JSON specification along with the implementation (in JavaScript) of two functions used by the workflow processes is sufficient for the execution of the workflow by the HyperFlow engine.

Figure 2: Specification of the *sum of squares* workflow in the HyperFlow format.

The workflow execution in this model consists of the following steps:

1. Await input signals.
2. Forward arrived signals to the input queues of their sink processes.
3. Each process:
    a) when all required signals have arrived, invoke the function passing the signals,
    b) wait for callback,
    c) emit output signals.

In this simple model it is very easy to transform a workflow into a distributed system. The key is step 3a) in which a process passes the execution to its function. The function could perform the calculations in-process, or it could just as well construct a job specification and pass it to a remote executor.

A cloud-based workflow execution could be implemented as depicted in Figure 3. When invoked, the function of a process sends a job specification to a remote message queue. It immediately subscribes to the queue in order to wait for job results. In parallel, local Execution Engines residing on Virtual Machines deployed in a cloud fetch the jobs from the queue, invoke the appropriate application components, and send the results back to the queue. When the results are received in the function of the Process, the callback is invoked. Note that multiple VMs and execution engines could be deployed and connected to the same queue which would act not only as a communication medium, but also as a load balancing mechanism.

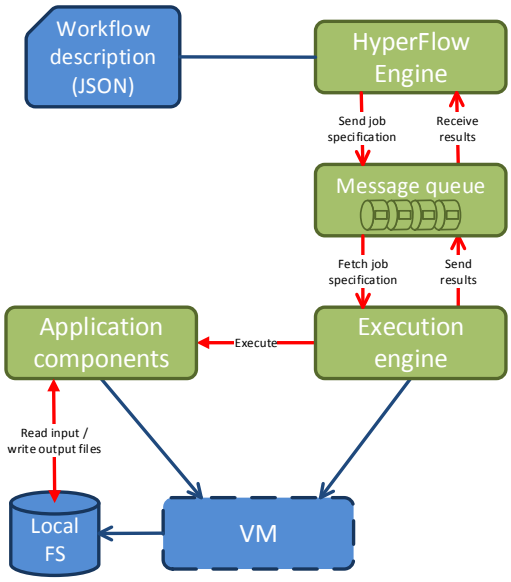

Figure 3: Workflow execution in a cloud enacted by the HyperFlow engine.

Currently used mechanisms to deploy scientific workflows on clouds require deployment of a cluster of virtual machines for running the worker nodes that execute workflow tasks. This step can be done manually, or use some automated tools for infrastructure setup and application deployment. These tools such as e.g. Chef [21] allow scripting the infrastructure configuration and automate the process of deployment and set up of application modules. Other tools, such as CycleCloud [22] are more specialized towards deployment of HPC clusters on demand. There is also theoretical and experimental work on scheduling and provisioning of cloud resources for scientific workflows exits (e.g. [6][24]), as well as on autoscaling of virtual clusters on the cloud [23]. These approaches are either based on advance planning of the full workflow schedule, which is not always feasible due to uncertainties in runtime estimation and the infrastructure behavior; or on dynamic autoscaling rules that do not take into account the application structure. Therefore, there is still need for systematic development on the new approaches and tools, and the model-based approach in PaaSage will provide a step towards creating a generic and extensible framework for supporting the large scale scientific workflows on clouds.

3.3.2 Target Picture (to-be)

General picture of the planned support of workflows in PaaSage is given in Figure 4. The workflow execution scenario fits well into the existing PaaSage architecture and is designed in such a way that the generic components of the PaaSage architecture are application agnostic. In the Upperware level the Workflow Planner will be responsible for preparing a scheduling and provisioning plan of a workflow and generate a CAMEL description that will be processed by the Reasoner and Adapter. In a similar way the Workflow Engine within the Executionware will be responsible for execution of the workflow tasks, while the deployment and applying of elasticity rules will be performed by the Deployer and Enforcement Engine.

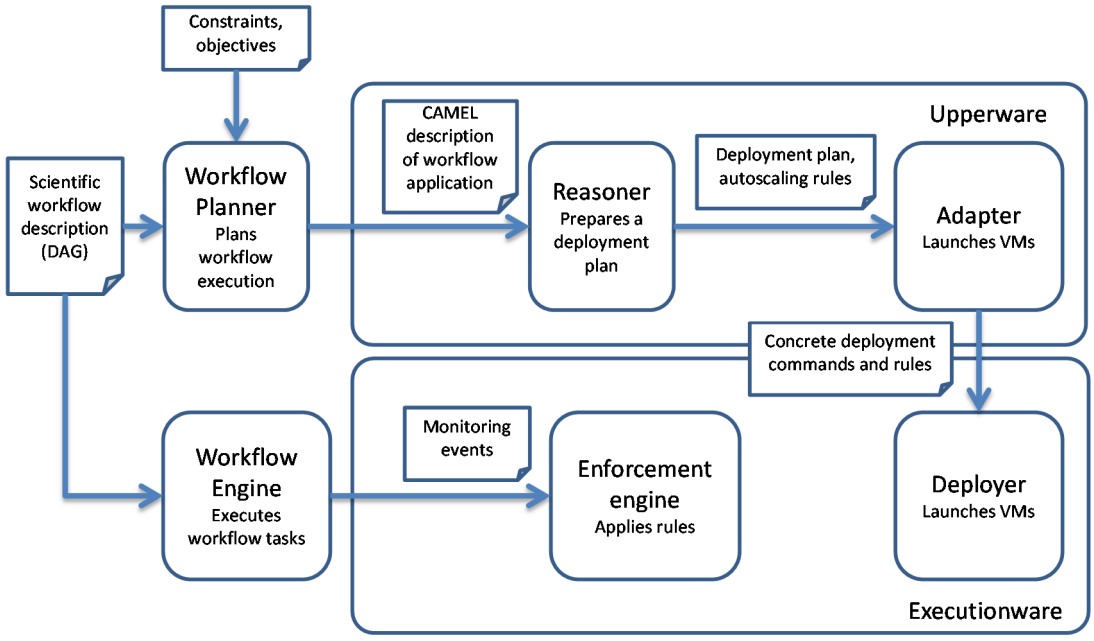

Figure 4 Target picture of scientific workflow planning and execution within PaaSage

The roles of the modules are explained as follows:

- The Workflow Planner is an application specific component and will use algorithms for planning the execution of large-scale scientific workflows. It will require the scientific workflow description in the form of a Directed Acyclic Graph (DAG) that will be a parameter of (or referenced by) the application model. The output of the planner will be the provisioning plan in the form of a CAMEL description of the workflow application architecture together with the elasticity rules to adjust the number of VM instances during the workflow execution.
- The roles of Reasoner and Adapter will be not changed, i.e. they will generate a concrete initial deployment plan based on the CAMEL description and it will issue proper deployment commands to the Deployer. The Adapter will be also able to trigger the Adaptation or request a new deployment plan from the Reasoner if such condition arises.
- The Deployer will deploy the workflow application and launch the required VMs on the clouds according to the plan. The application will then start.
- The Workflow Execution Engine (Hyperflow) is an application specific component from the perspective of the PaaSage architecture, but it is generic in the sense that it can execute different scientific workflows based on their description in DAG format. Therefore we distinguish as a separate module of the Executionware. During the execution, the workflow engine will spawn the processes of the workflow based on their dependencies. It will also trigger the monitoring events to the Enforcement Engine of PaaSage via the monitoring system.
- The Enforcement Engine of the Executionware will be provided with the elasticity rules for the specific scientific workflow application. When the application specific events are received via the monitoring system, the relevant autoscaling rule is triggered, and the proper scaling action is executed.

We believe that integrating the HyperFlow engine with the PaaSage platform will have mutual benefits. HyperFlow will benefit from PaaSage cloud deployment, execution, and autoscaling capabilities. The PaaSage platform, in turn, will gain the capability to support a new class of applications: large-scale resource-intensive scientific workflows. This support will be in terms of composition of existing application components into workflows, and – in conjunction with the workflow scheduler – their effective autoscaling.

3.3.3 Walkthrough PaaSage Workflow.

Step 1: Providing application workflow description and requirements
The user needs to provide the workflow description using the DSL, which is a DAG in JSON format. This will include all tasks with the dependencies, as well as input and output files in the form of URLs. Moreover, all non-functional requirements and constraints have to be provided. The examples of these constraints can include:

- the maximum cost (budget),
- a deadline,
- constraints on the cloud infrastructures (e.g. only private or public cloud can be used)

The possible objectives include:

- minimize time,
- minimize cost,
- maximize the number of workflows completed.

The user will also provide the application model in CloudML. The example of the application model for workflow engine is shown in Figure 5. The application consists of:

- Master which includes a Workflow Engine (Hyperflow) together with Redis Database and RabbitMQ server for communication,
- Shared storage (e.g. NFS server) that requires a separate VM for data exchange between workers,
- Worker that includes a part of executor (a generic component managing task execution) and application-specific binaries (e.g. for Montage application). Worker may be executed on multiple VMs, i.e. scaled out (horizontally) for parallel execution.

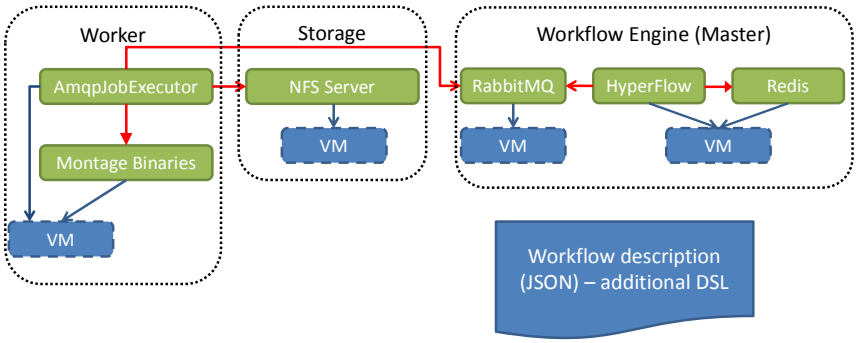

Figure 5 Workflow application described in CloudML in cloud provider independent model

Step 2: Workflow Planner prepares a deployment and provisioning plan in the form of autoscaling rules together with CAMEL description of the application.
The planner reads the workflow description together with the requirements and uses the planning algorithms to create the scheduling and resource provisioning plan.

- The scheduling plan will be then used by the workflow execution engine (Hyperflow) to schedule individual workflow tasks. The scheduling plan is opaque to the PaaSage Executionware, i.e. it is used as application-specific input data to the workflow execution engine.
- The provisioning plan describes when the VMs should be started and terminated, as required by the demands of the workflow. To interact with the PaaSage Executionware, the plan has the form of autoscaling rules.

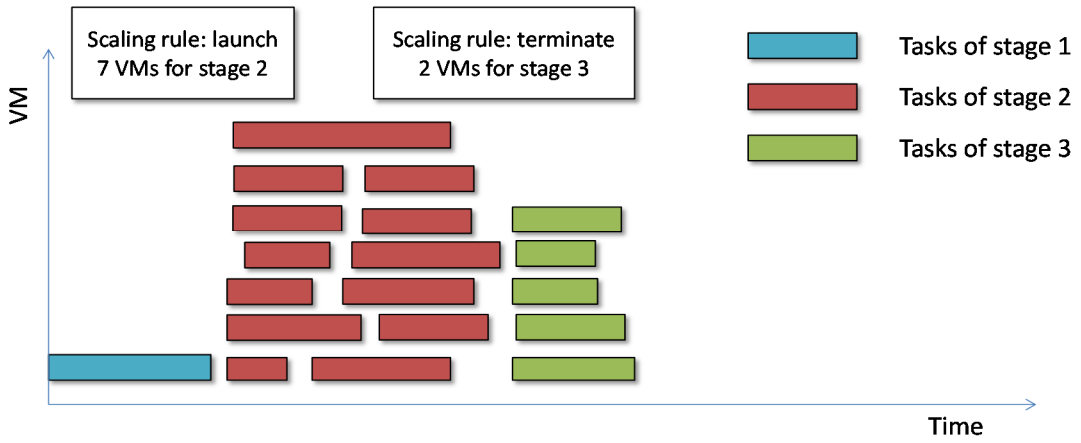

Figure 6 Example provisioning plan for workflow with 3 stages

The example of a scheduling and provisioning plan for large-scale workflow with 3 stages is shown in Figure 6. The initial deployment plan will require starting 1 worker VM instance for the stage 1 of the workflow and the range for autoscaling is set to [1..8]. In order to scale out the number of VMs for the stage 2 of the workflow, the following alternative autoscaling rules can be generated:

1. **if** workflow execution reaches stage 2 **then** launch 7 more worker VMs
2. **if** workflow execution time reaches 2 hours **then** launch 7 more worker VMs

These autoscaling rules do not require changing autoscaling ranges, or changing VM types. However, the rule 1 is application specific, i.e. it requires the application-specific event to be generated by the application to the monitoring system.

Step 3: Upperware generates the deployment plan
Reasoner and Adapter prepare a concrete deployment plan that fulfills the constraints of the workflow application provided in the CAMEL description. The output of the Upperware is a set of concrete deployment actions and elasticity rules that can be processed by the Executionware. The application specific scheduling plan will be passed from the workflow planner to the workflow execution engine after the deployment is ready.

Step 4: Executionware deploys and runs the application
In this step the deployer and the execution engine of the Executionware deploy the application using the deployment plan. The example deployment plan described in CloudML (Cloud Provider Specific Model) may look like the one shown in Figure 7.

In this example the initial deployment plan uses a private clod (OpenStack) for running the master part of the application, and the storage together with the worker are deployed on Amazon EC2. In this example the initial deployment contains a single instance of a worker.

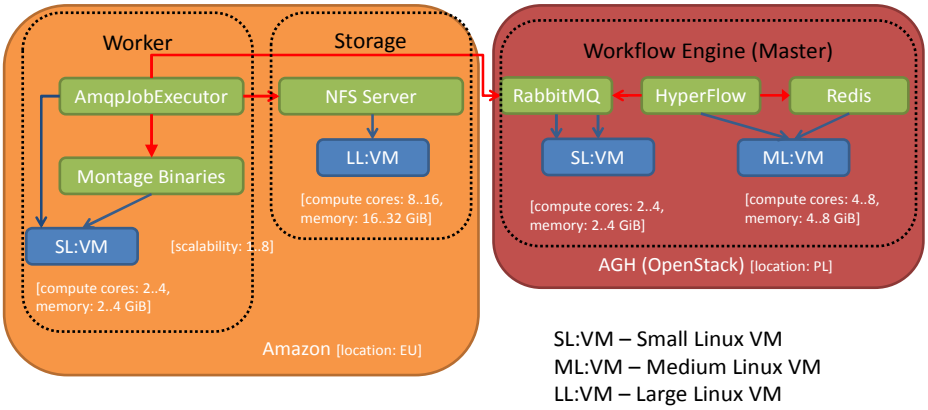

Figure 7 Workflow application deployment description in CloudML

Step 5: Application starts: Workflow engine manages task execution based on the DAG

The application starts its execution: the workflow engine reads the workflow description (DAG) and submits tasks to execution. In this scenario Hyperflow engine sends the ready tasks to the RabbitMQ server that manages the queue of tasks. The worker also connects to the RabbitMQ server and reads the tasks, processes them, and sends the results (confirmation) back to the RabbitMQ server. The input and output files are accessed using the shared storage (NFS).

Step 6: Executionware monitors the application and applies elasticity rules with execution engine

The execution of application can generate events to the monitoring system. There can be several possible types of events:

- Generic monitoring events from VMs, such as the update of the current CPU load or I/O load,
- Application specific events from the RabbitMQ server, e.g. update of the queue length
- Application specific events from the worker, e.g. task ID_42 has completed,
- Application specific events from the workflow engine, e.g. task ID_42 has been released to the queue, or stage 1 of the workflow completed.

The application specific events from the monitoring can trigger autoscaling rules in the Executionware. For example, if the workflow completes stage 1, new instances of the worker VM should be launched. Such scenario is illustrated in Figure 8, where an additional instance of the worker VM was launched.

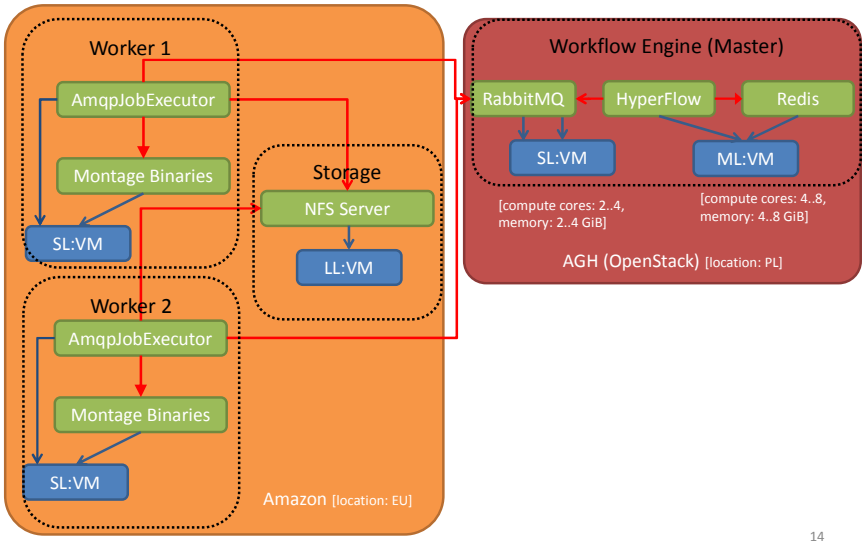

Figure 8 Workflow application after autoscaling rule launched additional worker instance

It should be noted that for the large-scale workflow applications the most relevant is horizontal scalability (scale-out), since the workflow application can benefit from parallel processing and thus launching new instances of VM is the best strategy, as opposed to vertical scaling.

3.4 Data farming

3.4.1 Current Status (as-is)

The data farming methodology defines a process of a virtual experiment, depicted in Figure 9, which organizes scientific research in a systematic manner. Each data farming experiment consists of the following steps:

1. *Experiment objective definition* formulates objectives which should be achieved by the experiment, along with a stop condition as the data farming process can be iterated many times before stopping.
2. *Simulation scenario building* concerns providing a simulation capable of generating the necessary data to answer the questions stated at the beginning of the experiment.
3. *Input space specification* results in a set of input vectors, each of which represents a single simulation case. As the input space can be extremely large, Design of Experiment (DoE) methods are often employed to reduce the number of input vectors.
4. *Simulation execution* involves execution of simulations with input vectors generated in the previous step, often run in parallel using High Throughput Computing (HTC). Depending on the input space this step can require a large amount of resources working together to provide the necessary computing power. Results from all simulations are aggregated for further analysis.
5. *Output data exploration* is where knowledge is extracted and new insights obtained. Should a simulation scenario require adjustments, step 2 may be repeated as needed. If additional areas of the parameter space need to be explored, step 3 is repeated. Otherwise the stop condition is considered fulfilled and the experiment concludes.

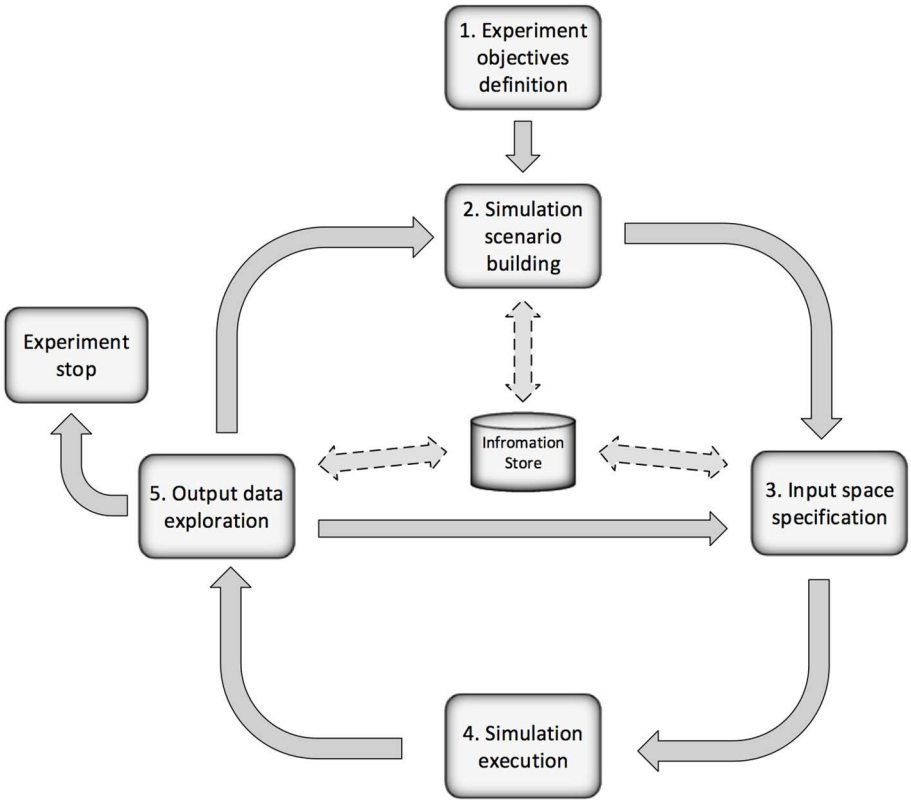

Figure 9: The process of a data farming experiment.

The Scalarm platform is our software of choice to conduct data farming experiments within PaaSage, due to its versatility and provided support for different computational infrastructures. Scalarm intends to fulfill the following requirements:

- support phases of a data farming experiment, namely: "*Input space specification*", "*Simulation execution*", and "*Output data exploration*",
- support different sizes of experiments from dozens to millions of simulations through massive self-scalability,
- support for heterogenous computational infrastructure including private clusters, Grids and Clouds.

Massively self-scalability is the main non-functional requirement which has to be supported by Scalarm in order to conduct data farming experiments at a large-scale efficiently. Activities performed in different phases of a data farming experiment impose that the used software and infrastructure is elastic and can be scaled automatically on demand, e.g.:

- during "*Input space specification*" multiple time consuming DoE methods can be executed to explore possibilities of input space size reduction,
- "*Simulation execution*" usually requires numerous simulation to be executed in parallel in a HTC manner,
- "*Output data exploration*" often involves executing computationally intensive data mining methods on large data sets to extract knowledge from simulations output.

A diagram of current Scalarm management process is depicted in Figure 10. Scalarm consists of loosely coupled services responsible for managing experiments, storage, and simulations.

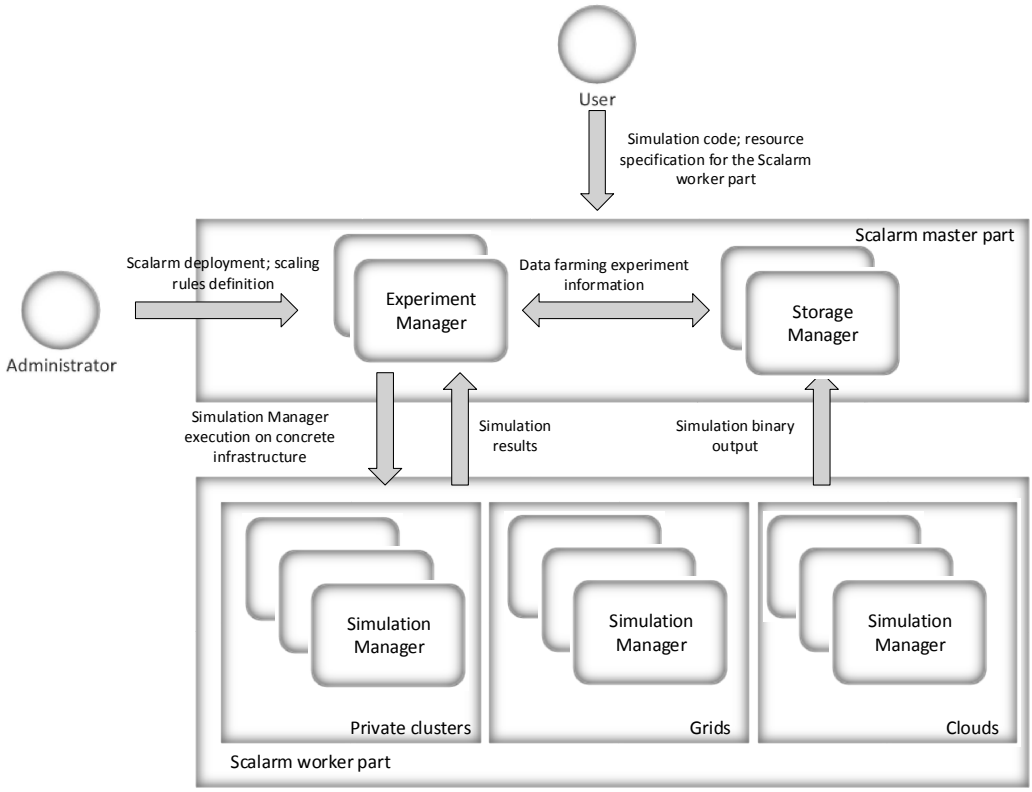

Figure 10: Scalarm management – current status.

Experiment and Storage Managers intend to be self-scalable, i.e. scalability of the services can be expressed in form of rules defined by an administrator, and enforced by the service itself.

The Scalarm architecture follows the master-worker design pattern, where the worker part is responsible for executing simulations efficiently, while the master part manages data farming experiments in general. In the current version, the user manually manages resources of the worker part (dedicate to run instances of the Simulation Manager service), i.e. additional workers can be scheduled to different infrastructures manually. On the other hand, an administrator can define so called *scaling rules* for services constituting the master part, namely Experiment and Storage Managers, to enforce scaling operations when necessary.

However, to attain a fully autonomous platform in regard to scalability, Scalarm needs to be extended in several areas. The first necessary extension is related to requirement definition for data farming experiments. Based on the provided requirements, a deployment plan along with scaling rules for Scalarm will be created. Hence the user or administrator will not have to define any additional rules. The second extension concerns elasticity of underlying, multi-cloud based infrastructure. Although Scalarm supports various computational infrastructures, using cloud environments only will enable better cost management.

3.4.2 Target Picture (to-be)

Incorporating PaaSage infrastructure into Scalarm will provide several improvements required to build a fully autonomous platform for data farming. The main difference between current status and target picture is the way in which Scalarm scales itself.

Currently the administrator and user are responsible for preparing scaling rules and managing resources in the worker part of Scalarm. In the target picture we intend to transform a data farming experiment into a model-based application with specified non-functional requirements, e.g. regarding cost and time constraints. A PaaSage-based data farming experiment is depicted in Figure 11.

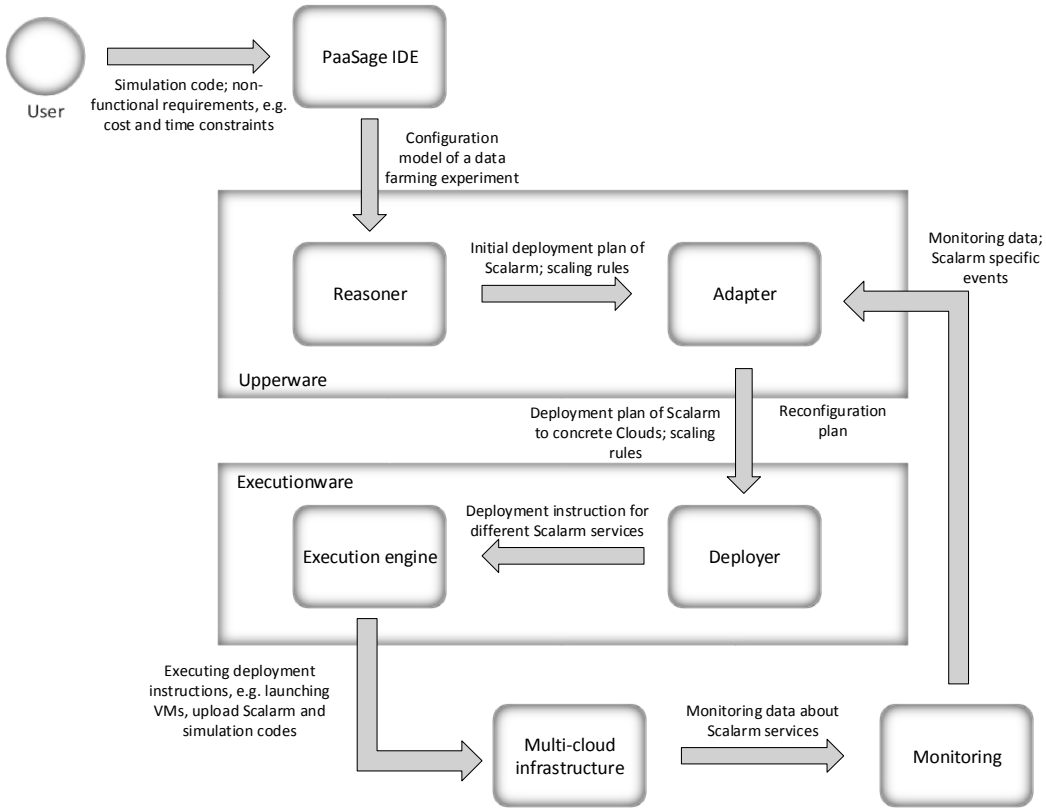

Figure 11: Data farming experiment in PaaSage with Scalarm.

Comparing to the current version, the target picture assumes there is no need for and administrator to be involved anymore. Scaling rules for Scalarm services are now generated by the Reasoner module, based on user-provided non-functional requirements. Moreover, the user doesn't need to manage Simulation Manager manually, provided requirements also include information related to Simulation Manager. As a result, the PaaSage Upperware is responsible for enforcing scaling actions for all Scalarm services.

Another improvement concerns Scalarm deployment. In the target picture, the PaaSage Executionware handles all actions related to deployment of different Scalarm services in a multi-cloud infrastructure. It will be supported by model-based descriptions of each Scalarm service in the CAMEL language. In addition, Scalarm will be able to generate specific events to PaaSage Monitoring, based on which scaling actions will be triggered. However, the user still will be able to use Scalarm to manage resources dedicated to the Scalarm worker part in a more manual fashion, e.g. to boost the data farming experiment by adding more resources. In such a case, Scalarm will generate events to inform PaaSage via the Adapter module, that a new deployment plan is required.

3.4.3 Walkthrough PaaSage Workflow.

Workflow of data farming is related to the data farming process in general (depicted in Figure 9) and intended deployment of Scalarm in PaaSage (depicted in Figure 11). More specifically it involves the following steps:

Step 1: Data farming experiment setup
An initial step of the process is to prepare simulation code and parameter space for a data farming experiment. In the PaaSage IDE, the user will provide all information necessary to start a new data farming experiment in Scalarm, e.g. parameter space to explore or Design of Experiment methods to apply. In addition, the user will be able to specify non-functional requirements, e.g. the stopping condition, maximum cost, or amount of resources to be used. Based on the provided information, the PaaSage IDE will prepare a configuration model of a data farming experiment to be conducted with the Scalarm platform and the provided simulation code.

Step 2: Deployment preparation and scaling rules generation
The configuration model will be passed to Reasoner to prepare initial deployment plan. The plan will involve deploying both the Scalarm master part, i.e. Experiment and Storage Managers, and the Scalarm worker part, i.e. Simulation Manager, which will execute the provided simulation code. The amount of necessary resources will be derived from the parameter space size and provided non-functional requirements. In addition, a set of scaling rules for all Scalarm services will be generated to attain user's requirements.

Step 3: Deployment on multi-cloud infrastructure
Then, the created deployment model will be passed from Upperware to Executionware to be executed. Deployer and Execution engine will call appropriate operations on selected Clouds, and the data farming experiment will be started. Scalarm will provide the user with a Web-based Graphical User Interface to enable progress monitoring and analysis of output data.

Step 4: Adjustements at runtime
Scalarm services will be monitored during runtime to attain requirements defined by the user at the beginning of the experiment. If any condition of generated scaling rules is met, appropriate scaling actions will be executed. Moreover, the user can decide to change previously created deployment plan, e.g. by adding additional resources to boost the experiment or deciding to execute time-consuming data exploration. In any case, reconfiguration plans will be prepared and and executed by Adaptor.

Step 5: Execution completion
When the stopping condition of the experiment is met, the user will be able to download any necessary output data and results. Then, all deployed Scalarm services will be stopped and Cloud resources will be released.

3.5 Summary

In this section we described the extended eScience case study by AGH that includes large-scale scientific workflows and data farming experiments. The analysis of how these applications fit into the PaaSage architecture reveals that the model based

approach proposed by PaaSage, together with the architecture of its Upperware and Executionware provide a suitable framework for supporting these applications.

The proposed planner module for planning workflow execution will generate a CAMEL description of the application and a set of autoscaling rules that will be subsequently used by the components of the current PaaSage architecture. The rules for scaling of workflow and data farming experiments will use application-specific events that will be passed via the monitoring system of the Executionware. Therefore the support for workflow execution and data farming experiments will fit well into the existing architecture of the PaaSage platform.

4 Financial Sector Case – Accounts Audit software

This case is supported by IBSAC Intelligent Business Solutions Ltd (IBS).

4.1 Organisation behind Case

IBSAC Intelligent Business Solutions Ltd provides consulting, IT and cloud services for several industries and has a leading position as a cloud services provider throughout Cyprus. We are offering to our customers the entire range of IT-services, including IT consulting, cloud services, systems installation and setup.

Our scope is to provide services and products that will enable our clients to use technology for the benefit of their organization, hence increasing their business value and profitability while at the same time organizing information and streamlining operations. Also, using our expertise, clients are implementing their businesses and having fast and actual ROI (Return On Investment).

All of the above can be fulfilled with the proven expertise of our engineers and sale consultants which have a vast amount of certifications from leading partners such as Microsoft, HP, Fujitsu and many others. All the certifications and memberships can be viewed on our website.

4.2 Objectives

Cyprus is one of the largest financial centres in Europe and Middle East. In specific, Cyprus has a large financial and auditing services sector where firms include the big four – Deloitte, PWC, Ester Young, KPMG – and several local town firms. Figure 12 shows that the financial and insurance sector had the biggest percentage in terms of economic activities in 2010. Thus, today's financial firms and internal accounting departments need Accounts Audit software in order to work faster and effectively, while constantly maintaining operational integrity and full compliance with international financial standards. These challenges call for a state-of-the-art Accounts Audit solution which optimally supports all the necessary functionality.

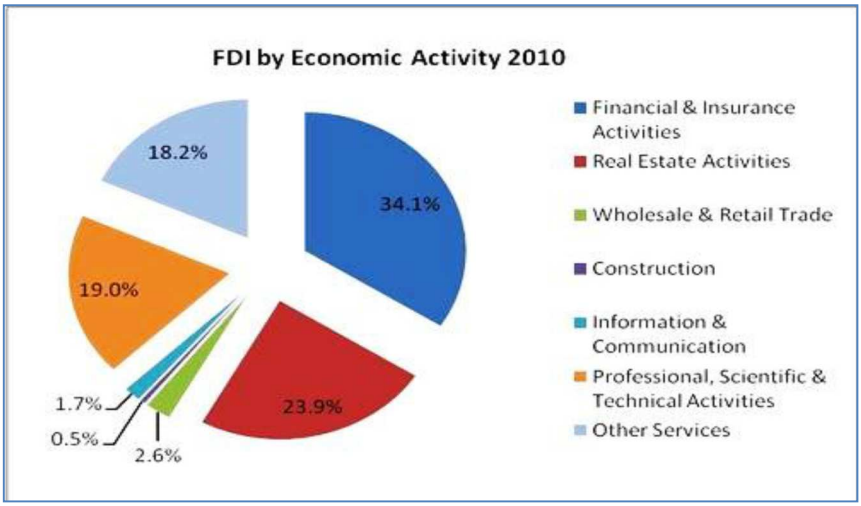

Figure 12 Foreign Direct Investment (FDI) By Economic Activity 2010

[Source: Cyprus Promotion Investment Agency -http://www.cipa.org.cy/easyconsole.cfm/id/131]

Yet no dedicated cloud-based auditing solutions exist, but rather proprietary solutions for each type of financial business. This provides a large and diverse market based on the diverse Cloud platforms currently employed by these firms for other company technological activities. From a wide variety of business applications and services that IBSAC Intelligent Business Solutions Ltd offers and supports, an application from the financial sector called Accounts Audit was selected. The application is used for the preparation of financial statements in full compliance with the International Financial Reporting Standards.

On one dimension the financial use case aims to address and fulfil the rapid elasticity requirement, which is mutual and critical for all industrial domains, by scaling resources on demand and employing a pay per use business model. On a complementary dimension, auditing firms have diverse requirements in terms of application setup, functionality, data security and maintenance. For instance, some auditing firms could well need to expose restricted functionality (i.e. read-only view of financial data) to their clients, while at the same time provide secured access to its employees while working remotely, e.g. from a client's network, while at home or while travelling. In overall, data confidentiality is a critical and sensitive issue for auditing firms due to the above point but also because in several cases the need may arise to provide data access to authorised parties for a limited period.

Moreover, the objectives of the Cloud-based Accounts Audit software include offering to auditing firms the opportunity to purchase SaaS auditing services deployed on their desired platform through a web based interactive tool, providing a SaaS auditing solution to consumers without the requirement of an initial investment, providing expandability for end-users (accounting firms), minimizing maintenance and support costs, minimizing operational costs through a pay-per-use model, maximizing the profitability of cloud provider and its clients, ensuring 99.99% uptime with an SLA (Service Level Agreement), providing interoperability with other operating systems and offering a faster, more efficient and more effective work while constantly maintaining operational integrity and full compliance with international financial standards.

4.3 Current Status (as-is)

Accounts Audit software is a specialised program which provides assistance to internal accounts departments of a company and accounting/auditing firms for the preparation of financial statements in full compliance with the International Financial Reporting Standards. The specific application has the following functionality:

- Provides the functionality to import the trial balance of a company in order to generate a full set of financial statements.
- Based on the parameters that are imported in the system (i.e. subsidiaries), the appropriate notes to the financial statements are generated.
- The system automatically generates the tax computations of the company where adjustments can be made to comply with the various tax laws and circulars issued to arrive at the taxable income of a company and its tax liability.
- It is generating automatically the income tax return of a company.
- Provides functionality of Lead schedule. Lead schedule is a report giving the makeup of each line included in the balance sheet and the profit and loss account.
- Includes a mechanism of generating documents based on specimens for income tax forms for the company, etc. In general, all the documents related with income tax forms are generated automatically from the system, or are available for manual input.

Financial Statements are prepared in accordance with IFRS and the Cyprus Companies Law both in Greek and in English. Options are available for companies, partnerships and sole traders. The Cash Flow statement and the Company Income Declaration (IR4 2004) are prepared effortlessly within the Template. Standardized Minutes of the Shareholders Annual General Meeting are also available for small companies. Financial Statements can be exported to various formats for use by other software, including Microsoft Word and PDF.

Some of the key features of the current software are the following:

- In accordance with IFRS and the Cyprus Companies Law: Financial statements are prepared in accordance with International Financial Reporting Standards and the requirements of the Cyprus Companies Law, Cap. 113.
- Bilingual - Greek and English: Either of the two languages can be selected for the preparation of the Financial Statements. Translating from one to the other language is achieved within minutes.
- Company Income Declaration: The Company Income Declaration (IR4) can be produced quickly and easily together with the Financial Statements and has been officially approved by the Ministry of Finance.
- Choice of Entity: Option to prepare Financial Statements for companies, partnerships or sole traders
- Cash Flow statement: The Cash Flow Statement is automatically produced with information from the Financial Statements and the accompanying notes.
- Printing choices accountants understand: Option to print a full set of Financial Statements, or various cut down versions, as required in practice.
- Avoiding duplication: The entry table is used for the entry of parameters and other information that are repeated throughout the Financial Statements, eliminating duplication.

- Dates as text: Dates are displayed in long format to avoid conflict due to computer regional settings
- No restrictions for users to amend the document: No restrictions are imposed in amending the financial statements to suit the particular disclosure requirements of each client.
- No entries outside the template are necessary: The intuitive interface enables entry on the face of the Financial Statements, just like any word processing software.
- Additional statements available for Tax purposes: Various additional statements can be produced if required for tax purposes.
- Standardized Minutes of the Shareholders AGM: Standardised Minutes of the Shareholders Annual General Meeting are available and can be particularly useful for small companies.

Currently, the Accounts Audit software is a Windows application, which operates on a client-server architecture. Client software is locally installed on the PC of each user. Users are strictly restricted to the auditing firm staff since the software can be accessed only while at the office. The server installation consists of two parts: Database and application server. The recommended scenario is to have two servers which one of them will host the database and the other the application. Only Windows Server can be used for database and application. Maintenance and support is carried out from an internal IT person or is subcontracted to an IT company.

However, the limitations of existing setup are the following:

- Cannot be used by employees and auditing firm clients on other operating systems such as Mac OS, Android and iOS.
- Users are strictly restricted to the auditing firm staff.
- Auditing firm clients are not able to interact at all with the software in order to view or update their financial data.
- Allows employees to work only from the office and not from a remote location.
- Mobile users (e.g. when an employee is travelling or is at a meeting with a client) cannot work or view data even with limited functionality.
- Maintenance and support are too expensive since the company needs to maintain a server, backups and daily check-up of the data integrity.
- Since the recommended scenario is with two servers (database and application) capital investment on hardware is high.
- Due to the fact that both application and database server are compatible only with Windows, the initial investment on licensing is quite high.

4.4 Target Picture (to-be)

The aim of the financial use case is to port the Audit application to the cloud utilizing the MDE methodology and environment defined and developed in PaaSage project. This allows providing a SaaS auditing solution without the requirement of an initial investment and on a pay-per-use model. Furthermore, this will enable support for any cloud platform and OS as required by different financial (auditing) firms that will suffice the following requirements:

1. Users will be able to work from anywhere without any additional costs - the only requirement is Internet connection.
2. Full desktop functionality or restricted data-view on mobile Smartphone or tablet devices will be enabled. Hence, auditing staff will be able to work:
    a. While at their home office using their PC or tablet.
    b. While at a meeting with an auditing firm client.
    c. While travelling since the bandwidth requirements will be minimal.
3. The Audit application being ported to a multiple cloud environment managed by the PaaSage platform will offer:
    a. Reusability of clouds without compatibility issues in case different auditing firms already use diverse clouds to deploy other applications. E.g. Auditing Firm A uses Windows Azure Cloud, while Auditing Firm B uses OpenStack Cloud.
    b. Transfer of the application or its modules (e.g. database) among multiple clouds without problems. E.g. Auditing Firm A deploys the application on OpenStack cloud to reduce costs, while deploys its database for added data security at an extra cost to Windows Azure Cloud.
    c. An accounting firm will be able to change cloud provider at any given time. E.g. an accounting firm uses Amazon cloud and they need to change to Windows Azure for company reasons.
    d. The software will offer load balancing using the PaaSage platform since the application can use Cloud provider A, database 1 can use cloud provider B and database 2 (replicated with database 1) can use cloud provider C. With this scenario, high availability and load balancing is provided since the load of the database will be managed from the application and will be shared based on resources.

Moreover, added benefits of porting the Audit application to the cloud are the following:

- Provide a SaaS auditing solution without the requirement of an initial investment and on a pay-per-use model.
- Maintenance and support will be carried out on a central location which corresponds to minimum overheads.
- Provide expandability, scalability and rapid elasticity.
- Customers as well will be able to interact with the software to view/update their financial data
- 99.99% uptime with an SLA
- Provide disaster recovery

4.5 Walkthrough PaaSage Workflow

This section was defined mostly based on the business requirements of IBSAC and in accordance also to the business needs of their clients, as well as on the initial PaaSage Workflow (D6.1.1, Figure 2-1), rather than the actual development workflow to be executed for the financial application use case.
Moreover, based on the Model Driven Development tools to be developed within the PaaSage project, the actual development workflow of the financial application use case will be further refined and customized to satisfy the use case requirements.
Are there any specific requirements and constraints in the context of individual steps?
The key technical requirements and constraints in terms of the individual steps to be performed in accordance to the PaaSage workflow are:

- Important data security concerns must be considered since a form of restricted functionality needs to be provided to financial firms' personnel and their clients.
    - On a complementary dimension data confidentiality is a critical and sensitive issue for auditing firms and must be achieved 100% . Specific issues include:
        - Secure VPN connection for remote data access.
        - Data encryption to facilitate information security and confidentiality.
- The PaaSage Upperware needs to support porting of the financial legacy application and its modules (e.g. database) among multiple clouds, via the IDE collection of tools and components to capture the needs when developing and porting models at design-time.
- The PaaSage Executionware needs to provide platform-specific mapping and technical integration to the Application Programming Interfaces (APIs) of the execution infrastructure of the Cloud providers including possibly:
    - Private clouds – database porting for information added security and confidentiality.
    - Public clouds – porting application modules.

Additional issues to be addressed are the following:

- Platform independence should be maintained in the case the cloud infrastructure changes.
    - Portability without vendor lock-in, including development, testing and elastic deployment during runtime.
- Services availability must be ensured based on the application of the pay-per-use model for financial services.

On the business side there are the following needs and issues to be addressed:

- The rapid elasticity requirement is critical for the financial use case and will be satisfied by scaling resources on demand and employing a pay per use business model.
- Maintenance and support will be carried out on a central location which corresponds to minimum overheads.
- Offer 99.99% uptime with an SLA.

- Providing disaster recovery.
- Offer SaaS auditing services deployed on customer's desired platform through a web based interactive tool, which also ensures interoperability with other operating systems:
    - The application will be used by many financial audit companies and clients, therefore users will be using various types of devices and operating systems on the client side such as laptops on newer as well as older versions of Windows, tablets on iOS, Android, Windows, etc.
- Offer a faster, more efficient, flexible and effective way to work while constantly maintaining operational integrity and full compliance with international financial standards.

Do you use specific tools or technologies?

The financial application is currently Windows-based and operates on a client-server architecture. The client application is installed locally on the PC of each user. Users are strictly restricted to the auditing firm staff since the software can be accessed from the office. The server installation consists from two parts: the database and application server.

Which steps are the most important or critical ones?

The key step is the porting of the legacy application, during which it is envisioned to be developed as a web-based application accessible via a web browser and using web services running on the server. Hence, the user will constantly need an active network connection to access the application, which may not be always available. Also, another important point is that since there is no access to the application's source code three mechanisms could be exploited based on the PaaSage initial architecture as follows:

1. The profile (characterisation) of the application will be performed by the application developers on the basis of what they know about the application;
2. Data collected in the metadata database from monitoring indicates the characteristics of the application;
3. External experts have knowledge of the characteristics of the application and will contribute by feeding these characteristics into the metadata database;

Moreover, remote access and handling of sensitive data is a critical issue to be ensured in the above step so as to ensure data confidentiality and security in overall by maintaining data encryption and a secure network access to those web services through possibly a setup of VPN network.

Finally, a business requirement that relates to the technical aspects of porting the application is rapid elasticity based on a pay-per-use model that should ensure proper scaling of resources to meet the variable customers' load.

How could the platform make a significant difference compared to today's practices?

The benefits of the platform in regards to today's practices are captured in Section: Target Picture (to-be).

5 Summary

This document is a first step towards designing the support of extended eScience use cases in the PaaSage project. The next steps will be focused on the concretization of the design and preparation of the first prototype, in the close collaboration of the efforts in WP3 and WP5 of the project. The experience with the prototype will lead to further refinements of the architecture, mainly towards the possible adaptation of the application at runtime and optimization of multi-cloud deployments.

Moreover, this document captures the initial study and requirements of the financial use case and its applicability to the PaaSage project. The next steps will be focused on further study and analysis of the languages, architecture, tools and methodology of PaaSage to be developed as part of WP1-5, to refine and capture the final requirements of the financial case study and use the PaaSage platform for the realization of the multi-cloud auditing application prototype.

6 References

[1] M. Malawski, J. Meizner, M. Bubak, and P. Gepner. Component approach to computational applications on clouds. *Procedia Computer Science*, 4:432-441, May 2011.

[2] T. Jadczyk, M. Malawski, M. Bubak, and I. Roterman, "Examining protein folding process simulation and searching for common structure motifs in a protein family as experiments in the Gridspace2 virtual laboratory," in *PL-Grid*, Springer-Verlag, 2012, pp. 252–264.

[3] M. Bubak, M. Kasztelnik, M. Malawski, J. Meizner, P. Nowakowski, and S. Varma, "Evaluation of Cloud Providers for VPH Applications," in *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, 2013, pp. 200–201.

[4] M. Ben Belgacem, B. Chopard, J. Borgdorff, M. Mamoński, K. Rycerz, and D. Harezlak, "Distributed Multiscale Computations Using the MAPPER Framework," *Procedia Comput. Sci.*, vol. 18, pp. 1106–1115, 2013.

[5] E. Deelman, G. Juve, M. Malawski, and J. Nabrzyski, "Hosted Science: Managing Computational Workflows in the Cloud," *Parallel Process. Lett.*, (accepted), 2013.

[6] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Cost- and Deadline-Constrained Provisioning for Scientific Workflow Ensembles in IaaS Clouds," in *SC '12 Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, 2012.

[7] D. Król, L. Dutka, J. Kitowski, "Towards efficient virtual experiments in metallurgy with data farming on heterogeneous computational infrastructure", in Cracow'13 Grid Workshop: November 2013, Krakow, Poland, pp. XX–XX, 2013.

[8] M. Ambrozinski, K. Bzowski, L. Rauch, M. Pietrzyk, Application of statistically similar representative volume element in numerical simulations of crash box stamping, Archives of Civil and Mechanical Engineering / Polish Academy of Sciences. Wrocław Branch, Wrocław University of Technology, 2012, vol. 12(2), pp. 126–132.

[9] Hluchy L., Kvassay M., Dlugolinsky S., Schneider B., Bracker H., Kryza B., Kitowski J.: Handling internal complexity in highly realistic agent-based models of human behaviour. In Proceedings of 6th IEEE International Symposium on Ap- plied Computational Intelligence and Informatics (SACI 2011), pp. 11–16, 2011.

[10] B. Kryza, D. Krol, M. Wrzeszcz, L. Dutka, and J. Kitowski, "Interactive cloud data farming environment for military mission planning support," Computer Science Journal AGH, vol. 13, no. 3, pp. 89–100, 2012.

[11] D. Krol, B.Kryza, M. Wrzeszcz, L. Dutka, and J.Kitowski, "Elastic Infrastructure for Interactive Data Farming Experiments," Procedia Computer Science, vol. 9, no. 0, pp. 206 – 215, 2012. Proceedings of the International Conference on Computational Science, ICCS 2012.

[12] D. Krol, M. Wrzeszcz, B. Kryza, L. Dutka, and J. Kitowski, "Massively Scalable Platform for Data Farming Supporting Heterogeneous Infrastructure," in The Fourth International Conference on Cloud Computing,

GRIDs, and Virtualization, IARIA Cloud Computing 2013, Valencia, Spain, pp. 144–149, 2013.
[13] B. Balis, B. Hypermedia workflow: a new approach to data-driven scientific workflows. In High Performance Computing, Networking, Storage and Analysis (SCC), 2012 SC Companion, pp. 100-107, IEEE 2012.
[14] P. Nowakowski, T. Bartynski, T. Gubala, D. Harezlak, M. Kasztelnik, M. Malawski, J. Meizner, and M. Bubak, "Cloud Platform for Medical Applications," in *eScience 2012*, 2012.
[15] R. Cushing, S. Koulouzis, A. Belloum, and M. Bubak, "Dynamic Handling for Cooperating Scientific Web Services," in *2011 IEEE Seventh International Conference on eScience*, 2011, pp. 232–239.
[16] T. Oinn, M. Greenwood, M. Addis, M. N. Alpdemir, J. Ferris, K. Glover, C. Goble, A. Goderis, D. Hull, D. Marvin, P. Li, P. Lord, M. R. Pocock, M. Senger, R. Stevens, A. Wipat, and C. Wroe, "Taverna: lessons in creating a workflow environment for the life sciences," *Concurr. Comput. Pract. Exp.*, vol. 18, no. 10, pp. 1067–1100, 2006.
[17] B. Ludascher, I. Altintas, C. Berkley, D. Higgins, E. Jaeger, M. Jones, E. A. Lee, J. Tao, and Y. Zhao, "Scientific workflow management and the Kepler system," *Concurr. Comput. Pract. Exp.*, vol. 18, no. 10, pp. 1039–1065, 2006.
[18] M. Malawski, K. Figiela, and J. Nabrzyski, "Cost minimization for computational applications on hybrid cloud infrastructures," *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1786–1794, Jan. 2013.
[19] E. Deelman, G. Singh, M.-H. Su, J. Blythe, Y. Gil, C. Kesselman, G. Mehta, K. Vahi, B. Berriman, J. Good, A. Laity, J. Jacob, and D. Katz, "Pegasus: A framework for mapping complex scientific workflows onto distributed systems," *Sci. Program.*, vol. 13, no. 3, pp. 219–237, 2005.
[20] G. Juve, A. Chervenak, E. Deelman, S. Bharathi, G. Mehta, and K. Vahi, "Characterizing and profiling scientific workflows," *Futur. Gener. Comput. Syst.*, vol. 29, no. 3, pp. 682–692, 2013.
[21] "Chef | Opscode." [Online]. Available: http://www.opscode.com/chef/.
[22] "CycleCloud: Overview." [Online]. Available: http://www.cyclecomputing.com/cyclecloud/overview.
[23] M. Mao and M. Humphrey, "Auto-scaling to minimize cost and meet application deadlines in cloud workflows," in *SC '11 Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, 2011.
[24] H. M. Fard, R. Prodan, and T. Fahringer, "A Truthful Dynamic Workflow Scheduling Mechanism for Commercial Multicloud Environments," *IEEE Trans. Parallel Distrib. Syst.*, vol. 24, no. 6, pp. 1203–1212, Jun. 2013.