Business-Oriented Evaluation of the PaaSage Platform: A Case Study in the Financial Sector

Achilleas P. Achilleos
and Georgia M. Kapitsaki
Department of Computer Science,
University of Cyprus
Nicosia, Cyprus
{achilleas, gkapi}@cs.ucy.ac.cy

Geir Horn
Faculty of Mathematics and Natural Sciences, University of Oslo
Oslo, Norway
geir.horn@mn.uio.no

George A. Papadopoulos
Department of Computer Science,
University of Cyprus
Nicosia, Cyprus
george@cs.ucy.ac.cy

***Abstract*— Cloud computing is an efficient and cost effective realization of the utility function principle. This enables, respectively, providers and end-users to offer and access computing resources on demand, benefiting from a pay-as-you-go business model. Over the last years, marketing and increased diversification of the cloud offerings have created a vast pool of choices for businesses, as well as increased expectations. This diversity of cloud infrastructures, platforms, and tools creates several challenges. In particular, it hinders interoperability, promotes vendor lock-in, but more importantly prevents businesses from making informed and optimal decisions when transitioning to the cloud. This paper presents the results of a business-oriented evaluation for cloud computing in the industry in Cyprus. The evaluation confirms that a set of key features are important for overcoming the above challenges and enable businesses to exploit the full potential of the cloud. A study and analysis of highly-related cloud tools is performed, which further reveals the support for most of the necessary features, and thus their importance. Finally, a platform for autonomic multi-cloud deployment is presented, showcasing its support for the required features through a case study in the financial services domain.**

Keywords—cloud computing; model driven; cloud deployment; component-based development; cloud resource scaling

I. INTRODUCTION

Cloud computing has changed the business landscape since it offers a new model that promises to deliver cost-effective utility computing based on a pay-on-demand business model. More to the point, businesses have high expectations for Cloud computing. In many cases deciding, selecting and transitioning to the cloud is performed without knowledge of the features, capabilities, and limitations that are offered by the technology. Moreover, the market uptake of cloud computing has created a diversity of cloud technologies and a landscape of an ever-growing number of providers offering a multitude of infrastructure-as-a-service (IaaS) and platform-as-a-service (PaaS) solutions [1], which strive to meet customers' high expectations. On the one side, this has the advantage that businesses have a variety of technologies and cloud offerings to choose from. On the other hand, the end-result is that all these cloud offerings create interoperability challenges, promote vendor lock-in and impede businesses by increasing the complexity in making informed decisions in terms of transitioning to the cloud.

There are different aspects to take into account when considering moving to the cloud. Specifically, the decision of a business to move to the Cloud is based on an assortment of requirements, conditions, constraints and preferences. For instance, financial businesses such as banks, audit and law firms have a strong requirement for data confidentiality, and thus the decision is expected to be a choice of a private cloud. If we take also into consideration the necessity for firms to reduce costs, a hybrid cloud deployment is also a desirable choice, where for example the applications servers can be deployed in a public cloud, while the database servers are deployed in the private cloud of the firms. In exceptional cases, a financial firm may be even satisfied with a selection of only a public cloud, but even in this case various cloud providers provide different offerings in terms of hardware specifications, costs, etc. Hence, even if a public cloud is chosen, the business needs to decide and make a selection among many different cloud providers.

This work focuses on identifying, defining and validating the key features supported by a diversity of Cloud tools, including the PaaSage[1] platform, a Model-based Cloud Platform Upperware, which can contribute to exploiting the benefits of the Cloud. In particular, these tools offer agility for the business in terms of deployment, monitoring, adaptation, and deployment of Cloud applications. Foremost, a business-oriented evaluation is performed during two industrial events in Cyprus, which aims to prove the importance of these features for exploiting the potential of the Cloud. Moreover, relevant tools are examined to showcase that these tools base their efforts on providing the best possible support for these key features. Finally, the support of the PaaSage platform for these features is confirmed via the demonstration of the capabilities of PaaSage through a business case in the financial domain that utilises PaaSage to address diverse needs of businesses.

The financial use case scenario is defined in the form of a Cloud Application Modelling and Execution Language (CAMEL) model [2], which is the key input of the PaaSage platform. This CAMEL model allows business stakeholders and developer operators (DevOps) to define the business preferences and technical requirements, which provide the capability to select and execute the optimal Cloud deployment. In essence, the model includes the necessary abstract

[1] **PaaSage: Model-based Cloud Platform Upperware** – http://www.paasage.eu/

information that drives and enacts the initial Cloud provisioning and deployment, while the PaaSage platform continues with monitoring and adaptation of the deployment solution when needed, e.g., when performance metrics are violated. In overall, this paper provides a business-oriented analysis and identification of key features for exploiting the full potential of Cloud computing, while introduces also their support by an open source research platform; i.e., PaaSage.

The rest of the paper is structured as follows. Section 2 presents the results of the business evaluation and the definition of the key features. Section 3 defines the study and examination of relevant Cloud tools and their support for the features. The next section serves as an introduction to the PaaSage platform and presents the capabilities of the platform through the definition of the CAMEL model for a financial business use case scenario. The language is at the core of PaaSage and reveals how the key features presented in this work are considered and covered by the platform, and can facilitate businesses in terms of making an informed and effective transition to the cloud. Finally, the conclusions of this work are provided in the last section.

II. A BUSINESS-ORIENTED EVALUATION

The evaluation presented in this section was performed in two cloud computing events contacted in Cyprus: The 5th ICT Summit and Exhibition and TechConnect, which took place earlier this year in Cyprus. The key objective of the evaluation was to identify and validate the significance of key features for exploiting the full potential of the Cloud, as viewed by professionals involved in different ICT but also business activities that consider the Cloud as an enabler.

Fig. 2 presents the results obtained from the completion of Part A of the defined business questionnaire[2], in relation to the experience of participants and their organisations with Cloud computing. In specific, Question A refers to the participant's personal experience with Cloud computing, as well as Cloud deployment. Moreover, Questions B and C refer respectively to the organisation's experience with Cloud and Multi-Cloud deployment. The answers to the questions show, respectively, that 76% of the participants and their organisations have medium-to-very-high knowledge in terms of the concepts of Cloud computing and Cloud deployment, while for Multi-Cloud deployment the percentage drops to 68%. This confirms that businesses in some cases are not knowledgeable and well informed on the capabilities, and thus do not fully exploit the potential of the Cloud. The very-low-to-low results refer to businesses outside the technology domain (e.g., auditing, lawyer firms), which completed the questionnaire.

The second part of the questionnaire was defined so as to validate the main features that a platform should provide in order to allow exploiting the full potential of the Cloud. Prior to the presentation of the results, the following list enumerates and defines a set of cloud key features, which are offered also in PaaSage, which are evaluated using the prepared business-oriented questionnaire.

- ***Multi-Cloud Provisioning***: Provision your Cloud infrastructure easily and across multiple providers [3]. Manage your Cloud servers throughout their full life-cycle, including your physical servers.
- ***Automated Deployment***: Deploy applications flexibly, automated and independently from the Cloud providers according to the applications' technical requirements.
- ***Monitoring and Adaptation***: The consumer can dynamically provision and release computing resources without requiring human interaction with the provider.
- ***Scalability***: Scalability is a planned level of capacity, with appropriate overhead, that you anticipate your company's systems to require over time, in addition to the ability to scale in a quick and easy manner when (and if) you need more (or less) resources.
- ***Elasticity:*** Ability to handle sudden, unanticipated, and extraordinary loads in an automated manner. This results in a massive, but brief, influx of users and load on the system.
- ***Deployment Optimisation***: The consumer can unilaterally be supported with continuous monitoring and optimisation of the deployment solution.
- ***Cloud Bursting***: Allows an application that runs in a private cloud to burst into a public Cloud or hybrid deployment when the demand for computing capacity spikes.

Appropriate questions were defined for each of the aforementioned features in order to assess their importance. For instance, in terms of Multi-Cloud provisioning the participants were asked to state their answer to the following: *"I think that my organization would benefit from exploiting multiple Cloud infrastructures"*. The results showcased that businesses do value as important the ability to exploit multiple cloud infrastructures. In fact, as shown in Fig. 2, twenty out of the twenty-five businesses (80%) agree that it is vital to exploit multiple cloud providers.

In respect to having the capability to perform automated deployment independent of the cloud provider, more than 80% of the participants declared that they somewhat agree or strongly agree with this point. Specifically, for Question E: *"I think that simplifying the management of Cloud services is useful"*, essentially all participants stated that they agree with this statement and also believe that automated deployment provides a vital feature for businesses (Question D). The optimization feature, i.e., Question F: *"Taking into account the characteristics of available cloud platforms (e.g. costs), the data to be used and the end-user preferences or restrictions (e.g. price, location, availability, privacy, etc.) for producing an optimal deployment solution is important for my business"*, is highly important for all businesses as shown in Fig. 2.

Finally, the evaluation results reveal that the ability to monitor and adapt applications is vital for a business. Going a step further, up to 24 out of 25 businesses acknowledged that monitoring, scalability, elasticity and Cloud bursting are critical for ensuring operational integrity of the business. Overall, the results of the survey conducted demonstrated the need for more flexibility in the management of cloud environments and applications deployed on the cloud.

[2] **PaaSage Business Questionnaire** – https://docs.google.com/forms/d/17ceAxV4O3b-Ez06WbsmN3LcXlOmUX5m-ZoqLuFK54As/viewform

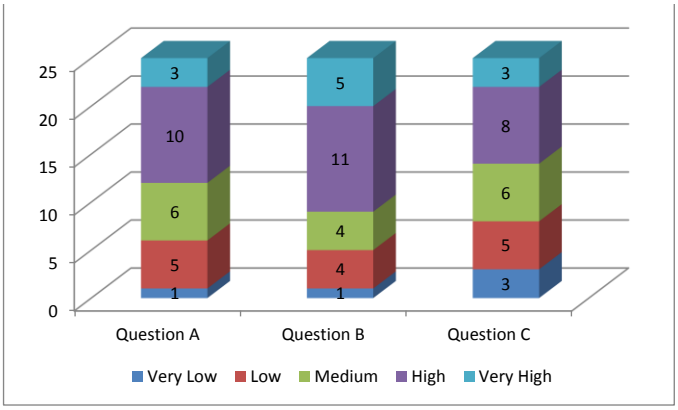

Fig. 1. Results of the Business Questionaire – Part A.

III. Open Source and Commercial Cloud Tools

In order to demonstrate further the necessity for introducing PaaSage, PaaSage platform and related Cloud tools are validated against the set of features that are defined as essential in this work for exploiting the benefits of the Cloud. In fact, the study is not performed as a means of comparison, but simply to showcase the added value that these shared features bring for many businesses and their importance in making informed and optimal decisions when transitioning to the cloud. Fig. 2 showcases the list of products that can be positioned within the SaaS and PaaS sphere of the cloud. In this analysis, Azure, Amazon Web Services (AWS), VMWare and OpenStack are considered as IaaS offerings used by these tools, and thus excluded from the comparison. Note that the study and analysis on the support of these features is based on documentation, white papers and tutorials available for the tools. The level of support of a feature by the tool is defined as: YES (Y), NO (N) or Limited (L).

The *multi-cloud provisioning* feature is basically satisfied by the tools along two directions. The first direction defines the support for applications deployment on mainstream Cloud providers (i.e., AWS, Azure, OpenStack and VMWare), even on the basis of hybrid configurations. Such systems refer to the PaaSage platform, RightScale and Apache Brooklyn that support major providers, and Cloudify that has an open plug-in architecture to support other Clouds such as Amazon AWS, GCE, CloudStack as well as Linux containers such as Docker. Cloudify currently has built-in plug-ins available for SoftLayer, Apache CloudStack and VMware. On the other hand, tools like OpenShift, IBM Bluemix and CloudFoundry offer multi-cloud provisioning capabilities respectively to Cloud Foundry Core compatible instances (AppFog, Tier 3, Uhuru Software, Micro Cloud Foundry and CloudFoundry.com), Bluemix-enabled Clouds (mainly OpenStack) and clouds running OpenShift Runtime Environments. These tools support public, private and hybrid configurations on their own proprietary clouds.

The *automated deployment* feature is heavily supported by all tools, while the differences in the scores have to do mainly with the support or not of mainstream Cloud infrastructures. In specific, the PaaSage platform offers a model-driven approach, which incorporates workflow-driven, script-based deployment, of applications. Cloudify offers a comparable approach that is based on a standard (i.e., TOSCA) [4]; although not model-driven. It provides a TOSCA-based Cloud orchestration platform that allows graphical-based design of blueprints for portability and management of Cloud applications. A blueprint defines the application topology and deployment plans. In overall, both PaaSage and Cloudify support workflow-driven, script-based deployment, but at a higher level of abstraction and with aim to increase automation. Apache Brooklyn uses a similar approach to Cloudify, since it employs the blueprint concept that defines an application, using a declarative YAML syntax supporting JVM plugins. Brooklyn can utilise Chef, Salt and similar scripts, to dynamically deploy application components. The rest of the tools employ mainly web dashboards that offer application development and workflow-driven, script-based automation capabilities in most cases.

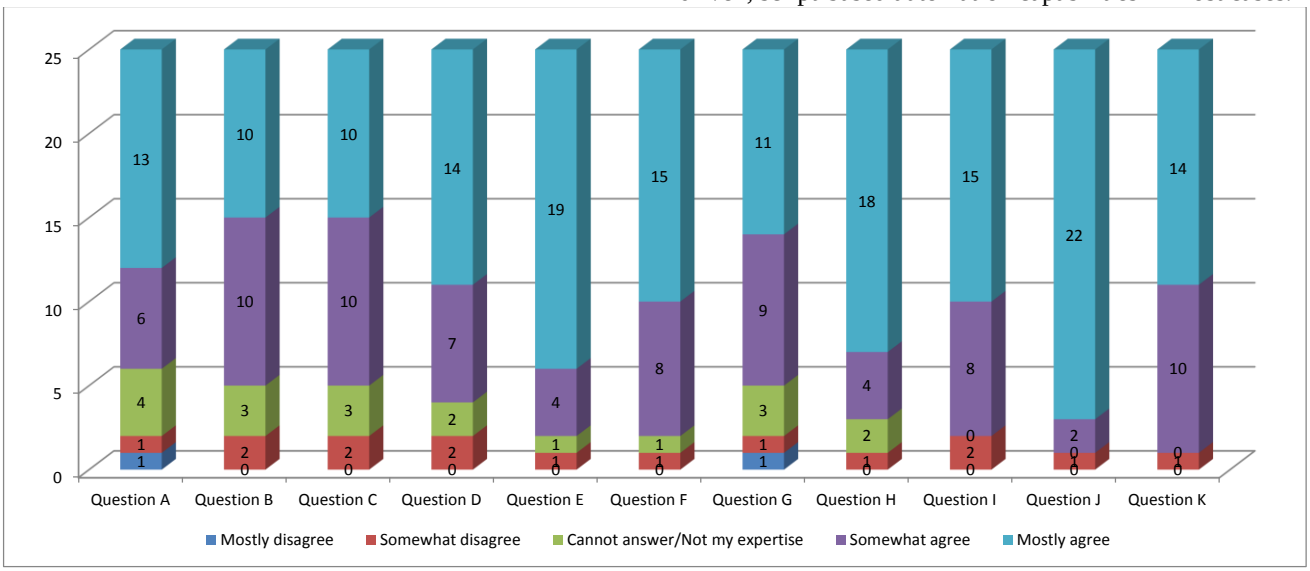

Fig. 2. Result of the Business Questionaire – Part B.

Monitoring and adaptation is highly related and an enabler for scalability, elasticity, optimisation and cloud bursting. The PaaSage platform supports monitoring and adaptation through the developed probes for CPU and memory, but also provides the capability for the application owner to develop additional monitoring probes. These probes can be used irrespective of the Cloud provider. For CloudFoundry, monitoring is supported on the five Cloud Foundry Core compatible instances. Moreover, for Cloudify, there is the need to develop monitoring tools and interface them with Cloudify to get events and metrics into Cloudify's Policy Engine. In terms of OpenShift the capabilities are limited to basic web traffic monitoring, Heroku core product does not support this feature, while monitoring and adaptation can be supported by paid add-ons such as HireFire and Scale Adept. RightScale supports server and application monitoring, and the triggering of adaptation and notifications with alerts and escalations. This is performed with the use of Tags that utilise CPU and memory metrics. Monitoring and adaptation in Bluemix is provided through the autoscaling add-on and its triggers (CPU, memory, Java heap memory). Finally, Apache Brooklyn supports collection of metrics that can be fed into policies, which automatically take actions.

Both *scalability and elasticity* are features provided by the PaaSage platform: The CAMEL language allows defining scalability rules acting on simple and user defined composite metrics. Based on the developed probes these metrics are continuously monitored and the rules are evaluated, which triggers the needed scalability and elasticity actions within a single Cloud provider when the conditions are violated. Using comparable approaches, RightScale and IBM Bluemix provide strong support for scalability and elasticity. In the case of RightScale, horizontal auto-scaling is configured with Voting Tags. These server tags are used for the purpose of setting up a scalable, alert-based server arrays. In specific, a server array is defined so that additional server instances are launched into the array (grow) or removed (shrink). Correspondingly, Bluemix provides an auto-scaling add-on and using the currently implemented auto-scale triggers (CPU, memory, Java heap memory) by runtime. In overall, these three cloud tools support CPU and memory metrics, while the capabilities are provided to develop extra probes for supporting other metrics.

The rest of the products have various levels of support for scalability and elasticity. More to the point, CloudFoundry has support for horizontal scalability on the CloudFoundry Core compatible instances, while Cloudify is at a similar level since it offers support for Softlayer, Apache CloudStack, and VMWare by writing scalability rules in text-based recipes. In terms of OpenShift, by default, the developed applications are not scalable. The possibility is offered to enable in the web dashboard the auto-scale functionality basically for monitoring web traffic. Furthermore, manual scaling is also possible by issuing console commands. The Heroku Cloud offering does not provide support by default, but third-party add-ons such as HireFire and Scale Adept can be purchased and used for load-based scaling and schedule-based scaling. Finally, Brooklyn includes the available policies of Auto-scaler and Load Balancer, and provides the possibility to develop new policies (e.g., CPU, memory) to perform custom runtime management.

A key characteristic refers to the capability to optimise the deployment of an application. PaaSage offers the capability to define in the CAMEL model optimisation requirements that are types of soft-requirements that indicate optimisation objectives on metrics or properties (at least one of these two should be referenced). In this sense, optimisation requirements indicate to the PaaSage platform that the respective values of these metrics or properties should be optimised in the context of a particular application or the component of an application. For instance, an optimisation requirement could indicate that the cost for the deployment of a particular application must be minimised.

We should highlight here that optimisation requirements can be characterised by non-negative priorities in the model, which indicate their relative importance to the user. Such priorities are indications to the PaaSage system for precedence to specific soft-requirements with respect to others that have also been specified by the end-user when e.g., deriving the respective deployment model of the application at hand. They can also guide the generation of the corresponding utility function of the constraint problem mapping to the end-user deployment (as well as other kind of) requirements. Priorities could be specified in the scale from 0.0 to 1.0 and the corresponding optimization requirements could map to metrics that are generated based on normalization functions which take values again in the set [0.0, 1.0]. In this way, the respective utility function would simply be the weighted sum of the functions of the metrics involved in the end-user's optimisation requirements.

IBM Bluemix provides an analogous approach and strong support for dynamic optimisation. In specific, it provides the Decision Optimization on Cloud (DOcloud) tool that is part of the BlueMix (PaaS) system. DOcloud utilises the Optimization Programming Language (OPL) models [5], which are defined using the CPLEX Optimization Studio. The OPL language provides a natural mathematical description of optimization models. Its powerful syntax supports all expressions needed to model and solve problems using both mathematical programming and constraint programming. Explicitly, OPL offers mathematical and constraint programming models that allow defining decision variables and decision expressions over index sets to represent choices and key performance indicators affected by them. Hence, it enables definition of an objective function of the decision expressions to maximise or minimise.

Finally, there is the concept of *Cloud bursting* that allows provisioning an application that runs in a private cloud to burst into a public (or hybrid) cloud when the demand for computing capacity spikes. In many cases this is critically important for a business since it enables to meet unpredicted demands, e.g., by transitioning to the public cloud that offers more resources, and then reverting back to the private cloud. Many of the tools provide support for the Cloud bursting feature. In specific, the PaaSage platform provides support for Cloud bursting based on the scalability rules and metrics defined in the model. CloudFoundry supports Cloud bursting, as with scalability and elasticity, on the Cloud Foundry Core compatible instances. It is performed with the autoscaler-process, which monitors the queues, and based on configurable thresholds, will increase or decrease the number of worker- process instances via the Cloud Foundry API.

TABLE I. OPEN SOURCE AND COMMERCIAL CLOUD TOOLS

Feature / Product	Model Driven	Multi-Cloud Provisioning	Automated Deployment	Monitoring	Scalability	Elasticity	Deployment Optimisation	Cloud Bursting
PaaSage	Y	Y	Y	Y	Y	Y	Y	Y
CloudFoundry	N	L	Y	Y	Y	Y	N	Y
Cloudify	N	Y	Y	Y	Y	Y	N	Y
OpenShift	N	L	Y	L	L	L	N	L
Heroku	N	L	Y	N	N	N	N	N
RightScale	N	Y	Y	Y	Y	Y	L	N
IBM Bluemix	N	Y	Y	Y	Y	Y	Y	Y
Apache Brooklyn	N	Y	Y	L	L	L	L	N

Next, Cloudify supports Cloud bursting on the supported clouds by writing recipes, while OpenShift allows deploying across private and public (OpenShift) Clouds with the intelligent capacity on demand capability of the tool. In terms, of Heroku and RightScale the study of relevant documentation did not reveal any support for Cloud bursting. The IBM Bluemix system does not offer built-in support for Cloud bursting, but it can be used together with the IBM UrbanCode tool to support Cloud bursting and hybrid cloud deployment. Finally, the study of relevant documentation did not reveal any details for the support of Cloud bursting by Apache Brooklyn.

Overall, PaaSage proves to comply with all key features defined for Cloud environments in comparison to the other tools. Note that, however, emphasis is given in the facilitation of the development process and in the automation in cloud-relevant decisions (i.e., hybrid deployments). A brief description of the use of PaaSage is given in the next section.

IV. A BUSINESS CASE FROM THE FINANCIAL SECTOR

This section of the paper defines a financial business use case and provides a step-by-step walkthrough on how the key features presented in this work are delivered by the PaaSage platform. This allows demonstrating, in practice, the importance of these features for providing the capability to exploit the benefits of the cloud.

A. Infoscreen Quorum

Infoscreen Quorum[3] is a software solution, ideal for organisations wishing to upscale and streamline their entity management and company secretarial operations, as well as improve their corporate compliance. The solution is used by many of the major auditing, legal, trust and specialist providers offering corporate secretarial, trust administration, financial and additional professional services, mainly in the island of Cyprus. This Windows-based application is based on a client/server architecture and it is currently used by businesses on their own infrastructure. The Quorum application consists of three components: the Quorum Client, Application Server and the Database Server (Firebird RDBMS). Fig. 3 defines the client/server architecture of the application. The application and database servers are the ones considered for cloud deployment.

B. Infoscreen Quorum: A Sample Use Case Scenario

In this subsection one of the scenarios of the financial use case is described as follows, although more cases are considered in the framework of PaaSage. The scenario is defined around the notion of the Central processing unit (CPU) as the metric that enables monitoring and adaptation of the deployment of the Quorum application.

Use Case Scenario: *"The Quorum application is heavily used during the corporate levy period (i.e., in April) by employees of financial, auditing and lawyer firms. The customers require that the Quorum application is deployed at the private cloud (i.e., Windows Server 2012 R2 - Cyprus), with one VM for the Application Server and one VM for the Firebird DB Server. In the case of increased load (i.e., CPU load to high, CPU > 70%), mainly occurring during the levy period, the customers require to dynamically scale out to the public cloud (i.e., Windows Azure - Ireland) and provision VMs for both the application and database servers to meet the increasing demands and facilitate efficient execution of employees tasks."*

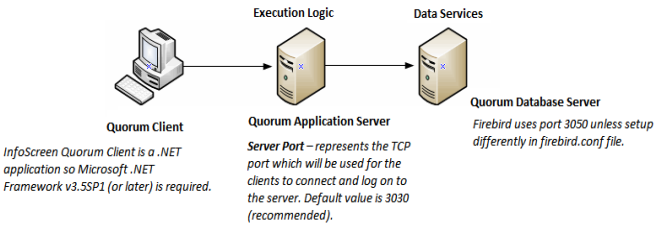

Fig. 3. The architecture of the Quorum Financial Application.

C. The PaaSage Platform

Prior to the definition of the CAMEL model for the above use case scenario the PaaSage platform is introduced in this section. Model Driven Engineering (MDE) [6], [7], refers to a software engineering approach that brings models as the prime artefacts of the software development process. In MDE, the abstract syntax of a Domain Specific Language (DSL) is typically defined in the form of a metamodel. This metamodel describes the set of modelling elements, their attributes, and

[3] **Quorum** – http://www.quorumcentral.com/index.php

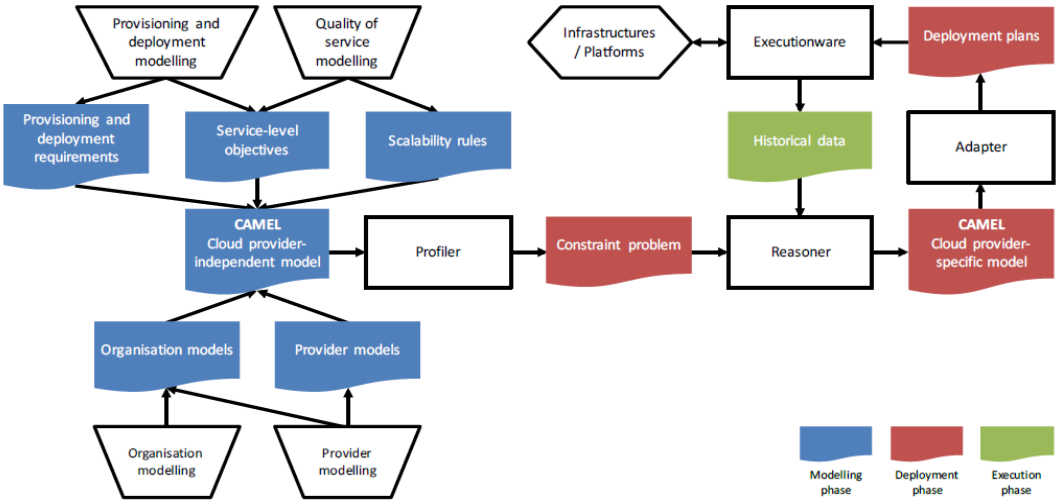

Fig. 4. The PaaSage workflow [2].

their relations, as well as the rules for combining these concepts to specify valid models that conform to the metamodel [2]. On the other hand, the concrete syntax may vary depending on the domain, e.g., a DSL could provide a textual notation as well as a graphical notation along with the matching serialisation in XML Metadata Interchange (XMI).

CAMEL is a domain specific modelling language, which is found at the core of the PaaSage platform. In fact, CAMEL integrates and extends existing domain-specific languages (DSLs), namely the Cloud Modelling Language (CloudML) [1], Saloon, and the Organisation part of CERIF [2]. In addition, CAMEL integrates new DSLs developed within the project, such as the Scalability Rule Language (SRL) [2]. The abstract syntax of the CAMEL language describes in fact the set of cloud concepts, their attributes, and their relations, as well as the rules for combining these concepts to specify valid statements that conform to this abstract syntax. On the other side, the concrete syntax of CAMEL, in line with the MDE principles, describes the textual or graphical notation that renders these concepts, their attributes, and their relations [2].

More to the point, the CAMEL model includes several sub-models (e.g., deployment, provider, scalability, metric, security, organisation) that define the concepts. It also provides the editors that enable a developer-operator not only to model the application to be deployed, but to model the features of the available Cloud providers, goals and preferences to be satisfied by the deployment, such as response times or deployment cost, etc. CAMEL integrates the DSLs using the Eclipse Modelling Framework (EMF) on top of the Connected Data Objects (CDO) persistence solution, to maintain model information between different application deployments and parts of the PaaSage autonomic deployment platform. The CDO repository is referred to as the metadata database. The intention is that different users will be able to form a "social network" that will enable them to exchange and/or trade models and statistical information from the deployments. For instance, one user has developed a parametrised model for a particular Cloud provider, which can be shared or traded with other users of the PaaSage and immediately allows all PaaSage users to deploy to this Cloud provider.

In overall, the key objective of PaaSage, served at the core by the CAMEL modelling language (see Fig. 5), is to define a CAMEL model, which is then transformed by the rest of the components of PaaSage to a deployed application in one or more Clouds (i.e., private, public, or hybrid). More to the point, PaaSage adopts CAMEL models that are progressively refined throughout the PaaSage workflow (Fig. 4), and which drive the integration with and across the components responsible for the *three life-cycle phases [2] of (i) modelling, (ii) deployment and (iii) execution of multi-cloud applications*.

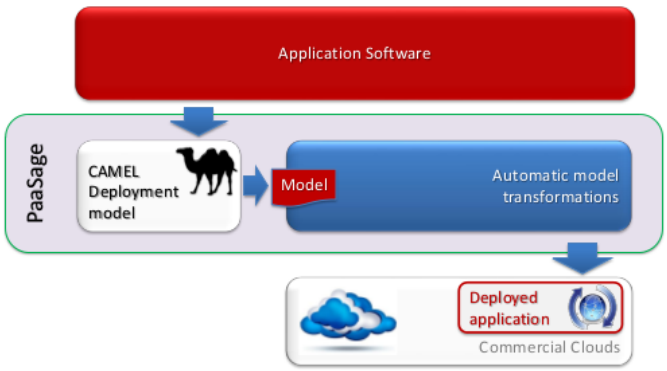

Fig. 5. The objective of the PaaSage platform.

The application's CAMEL model is first defined using the developed Eclipse-based modelling editors, which is then transformed by the *Upper Ware*. The *Upper Ware* consists of a set of software components (i.e.,a profiler, a reasoning engine, and an adapter) that transform the model, to derive application deployment plan, which satisfies all the constraints and goals for the deployment set defined by the user in the CAMEL model. The generated deployment plan enables selecting one or more Cloud providers and platforms (i.e., private, public, or hybrid). The result of the Upper Ware's model transformations is the generated deployment plan, i.e. the application's installation scripts passed to the *Execution Ware*, which is responsible to instantiate the various application components on the selected Cloud providers and platforms.

Furthermore, the *Execution Ware* is responsible to monitor the set of defined metrics, so as to make autonomous scalability, elasticity and Cloud bursting decisions within the boundaries of the deployment plan. In the case the application execution triggers conditions that cannot be satisfied by the current deployment plan; the *Execution Ware* will pass control back to the *Upper Ware* to find a more suitable plan for the current application context if the scaling limits of the current deployment plan are reached. The monitored metrics will subsequently be consolidated as statistical knowledge in the metadata database to guide decisions on future deployment plans. This feedback loop controlling the application deployment is depicted in Fig. 4, which showcases in overall the PaaSage workflow and the different logical parts of the *Upper Ware* and the *Execution Ware*.

D. Definition of the Quorum CAMEL Model

The definition of the Quorum CAMEL model for the sample scenario starts with the definition of the software application architecture, in a subpart of the model, namely the *deployment model*. In this sub-model, the main components of the application are defined, which will be used for the deployment of the application. The definition includes aspects such as the communication channels (e.g., ports) between the components. In the case of Quorum the two components considered and defined in the model are: (i) the Quorum application server and (ii) the Firebird database server.

Fig. 6 demonstrates the definition of the deployment model, which presents the definition of Firebird as the database server. The communication channel provided by the Quorum Firebird component is defined on port 3050, which is respectively the required communication channel defined in the Quorum application component. This means that the application server communicates with the Firebird database server on port 3050. Another key part of the definition of the sub-model refers to the deployment information. The deployment information points to the source from which both the binaries of each component and the deployment scripts can be downloaded for installing each application. In this case, two scripts have been developed using PowerShell, which will be used to perform the workflow-driven, script-based deployment of components to the Cloud.

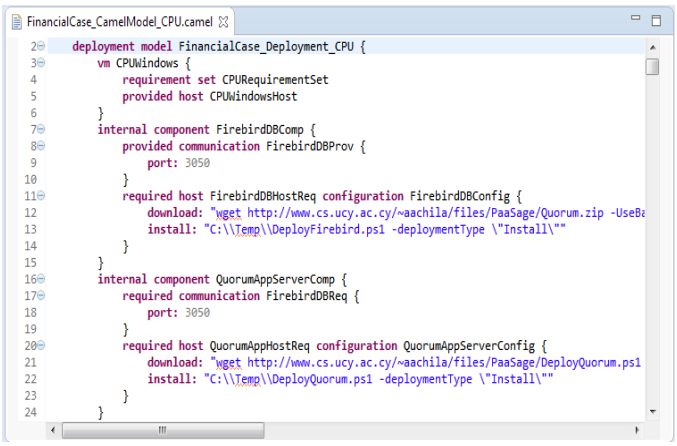

Fig. 6. The deployment specification of the CAMEL model.

Next, the organisation sub-model is defined as showcased partly in Fig. 7. This part of the model defines various aspects of an organisation that is provided access, and thus allowed to exploit the PaaSage platform, in order to deploy applications and adapt them at runtime. It also defines details on a Cloud provider, since it specifically provides additional information such as whether the Cloud offered is public (not defined in this case since the Cloud considered is private), and the types of Cloud services offered (see PaaS, IaaS attributes defined in this case). Also, the security capabilities offered and the description of the provider's service offerings are defined explicitly through the reference to this specific cloud provider model (i.e., *UCY_PrivateCloud*).

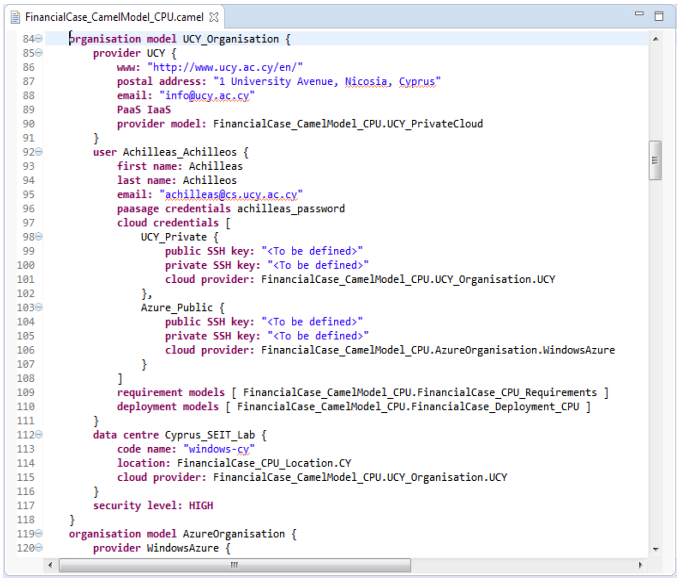

Fig. 7. The organisation specification of the CAMEL model.

In addition, an organisation user can be defined, which is another entity type that is identified by the first and last name, an email, the credentials, and the URL of a personal web site. It is also associated to a set of requirements that are specified in the requirements sub-model(s) and to a deployment sub-model. Both sub-models will be described later. Moreover, specific credentials are defined and used to authenticate the user and grant access to the PaaSage platform and its capabilities. An additional set of credentials are defined and will be used for authenticating the user against a Cloud provider during deployment, thus enabling the PaaSage platform to perform Cloud-specific tasks on behalf of the user and in accordance to the defined CAMEL model. The definition of Cloud credentials includes: (a) the reference to the respective cloud provider on which these credentials should be used, (b) the public SSH key and (c) the private SSH key, which are needed to enable the deployment of the application on this specific cloud provider.

Correspondingly, each provider is defined in the relevant sub-model presented in Fig. 8, which shows two extracted parts of the provider models; i.e., the UCY private cloud and the Azure public cloud. PaaSage uses provider models in the life cycle phases of modelling and deployment, in order to match deployment models with the compatible cloud providers. In

specific, and as shown in the following figure, the various VM offerings provided by each cloud provider and their characteristics are captured in each VM definition. For instance, the *TINY_VM_Constraint_Mapping* refers to a VM at the University of Cyprus (UCY) private cloud that offers the specified RAM, Cores and Storage characteristics. Respectively, the Azure public cloud provider offers VMs with more resources, like the *A3_Large_VM_Constraint_Mapping*, which is one of the VMs defined in this provider sub-model (A4 Large is also defined), and refers to a specific VM (named A3 Large) offered by Azure[4].

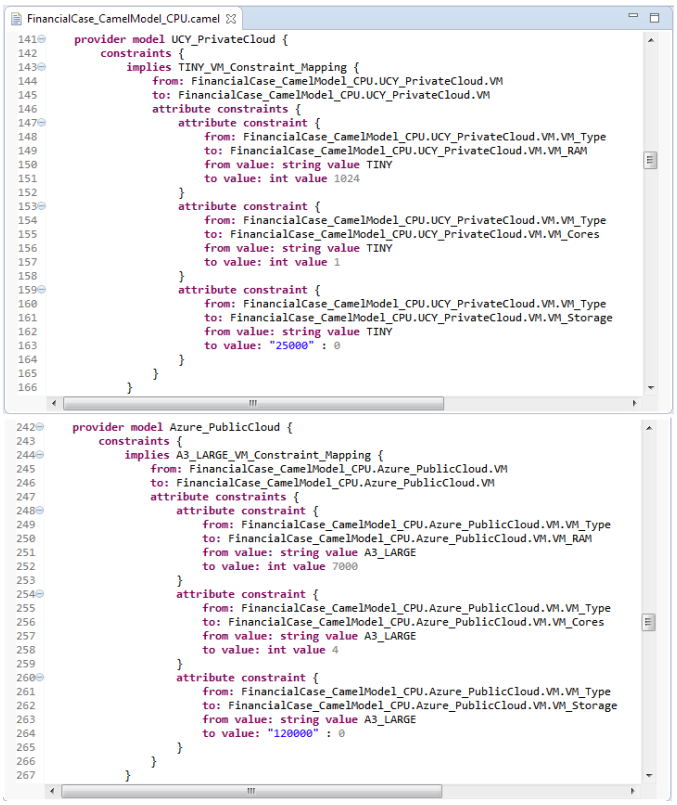

Fig. 8. The provider specification of the CAMEL model.

The following figure refers to the location specification of the CAMEL model that defines the name, the geographical regions and the countries. Countries and geographical regions are denoted by a region keyword, where in both cases the ISO2 code of the respective region is used. In the region body the full name of the geographical region is defined. Respectively, the body of the country definition includes its name and reference to the parent region. For instance, in the sample scenario the region is defined as Europe (i.e., EU), while Cyprus (i.e., CY) for the private cloud and Ireland (i.e., IE) for the Azure public cloud are defined as the countries. Both have EU defined as the parent region. Note that a parent region is referenced starting with the name (i.e., *FinancialCase_CPU_Location*) of the location model, followed by *'.'* and then the ISO 3166-1 alpha-2 code of this parent region (i.e., *EU*). In general, this is the way that referencing across different sub-models is supported in terms of the concrete syntax provided by the xText based editor of the CAMEL language.

[4] **Azure General purpose compute: Basic tier VMs** – http://azure.microsoft.com/en-us/pricing/details/virtual-machines/

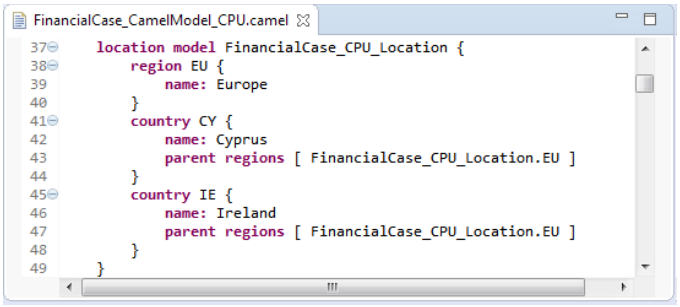

Fig. 9. The location specification of the CAMEL model.

PaaSage uses metrics in the lifecycle phase of execution (see Fig. 4). For this purpose, the metric sub-model is based on the Scalability Rules Language (SRL) [8], [9], which allows specifying rules that support complex adaptation scenarios within each Cloud provider use for the application execution. In this case, a CPU-based metric is defined, as shown in Fig. 10, which refers to the property to be monitored (defined as CPU), and a CPU raw metric is defined with a range from 0 to 100. A sensor for monitoring the CPU load and a metric condition are also defined for checking that the CPU does not exceed the threshold of 70% utilisation. This check applies for the value calculated by the mean value composite metric. In this way the metric package of the CAMEL language allows defining metrics that are associated to sensors and enable monitoring and adaptation actions.

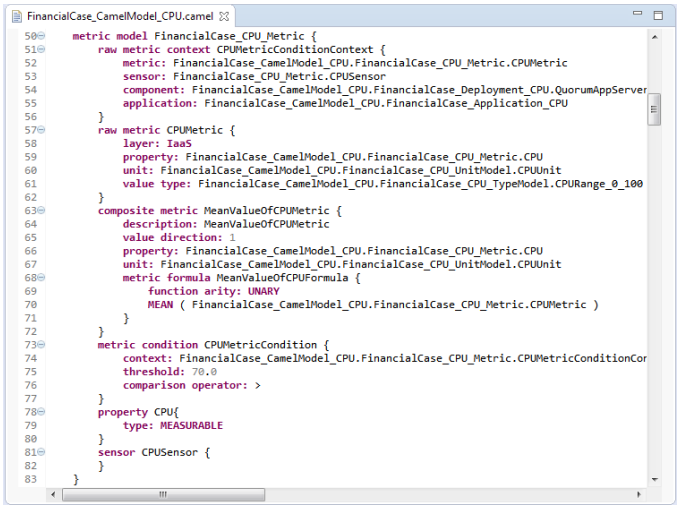

Fig. 10. The metric specification of the CAMEL model.

Adaptation is covered mainly by the scalability sub-model of CAMEL, which defines basically scalability rules, event patterns and actions. The scalability sub-model is also based on the SRL language. Therefore, SRL provides mechanisms for (a) specifying event patterns, (b) specifying scaling actions, and (c) associating these scaling actions with the corresponding event patterns [2]. In order to identify event patterns, the components of the application must be monitored. Therefore, SRL provides mechanisms for (d) expressing which

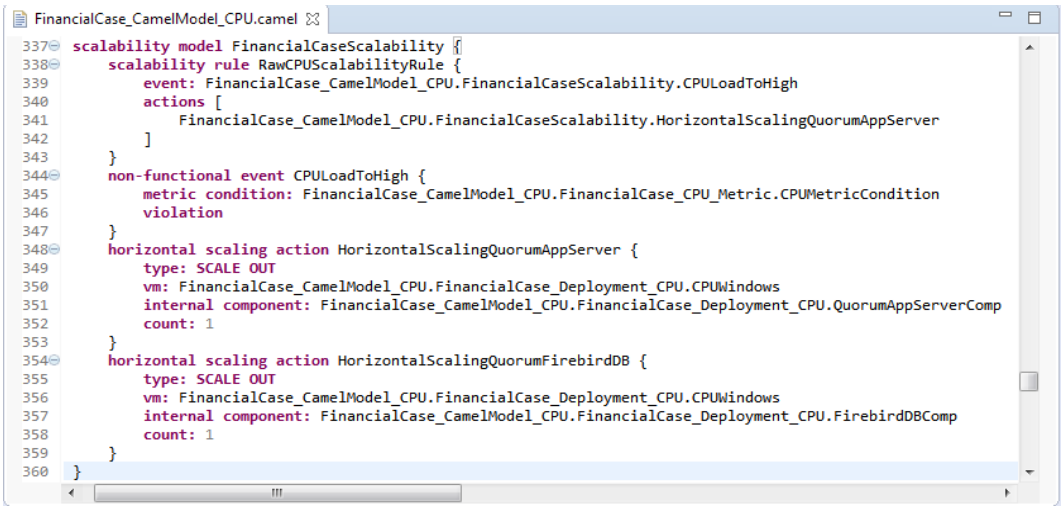

Fig. 11. The scalability specification of the CAMEL model.

components must be monitored by which metrics and for (e) associating event patterns with monitoring data [2].

Fig. 11 shows the scalability sub-model for the financial case scenario. A scalability rule (i.e., *RawCPUScalabilityRule*) is defined and associated with the *CPULoadToHigh* non-functional event and the appropriate actions are also defined in the scalability sub-model. As can also be seen in Fig. 11, the non-functional event is associated to the metric condition. The event thus fires when the condition is violated, and triggers accordingly the associated scaling actions.

Continuing the example of the CAMEL model for the financial use case, the specification of the requirement sub-model is performed; see Fig. 12. The requirement part of the model defines two location and two scale requirements, one Operating System (OS) requirement, one quantitative hardware requirement and a Service Level Objective (SLO). Foremost, the location requirements are identified by a specific name and in the body specification a set of locations must be provided separated by comma. In this case the location requirement *CyprusReq*, indicates that a single location (i.e., Cyprus) is defined for the private cloud, with reference to the location model (defined also in the model). The other location defined in the requirement model refers to the Azure public cloud and indicates that it should be deployed at Ireland's datacenter.

Two horizontal scale requirements are also defined in the requirement part of the model, which first indicate the internal component associated with the scaling action. This is made via a reference to the respective deployment model and then the minimum and maximum number of instances for this component. In particular, the *HorizontalScaleRequirement* is defined for the *QuorumAppServerComp* component and another one is defined for the *FirebirdDBComp* component. Both indicate that at least one instance of each component must be generated. In the body of the OS requirement, the *String* specification of the desired OS must be specified followed by an optional indication of whether the OS is 32-bit or 64-bit. In this specific case, the OS is defined as *"Windows"* and it's also indicated that it must be *64-bit*.

Quantitative hardware requirements are used to define the characteristics of one or more VMs, specifying explicitly the minimum and maximum values for each VM property. In overall, each hardware requirement defines the minimum and maximum values needed for supporting the execution of an application after deployment. In this example scenario the values are defined for the VM cores, RAM and storage as depicted in Fig. 12. The use of the *".."* token indicates that the values provided are the minimum and maximum values for each defined VM characteristic. In the case there is an absence of a value, for instance for the upper bound of a characteristic, this indicates that it is not a requirement imposed by the user. Thus, it's does not need to be taken into consideration by the *Reasoning engine* component of the PaaSage platform, when selecting the proper VMs offered by cloud providers.

Following, a Service Level Objective (SLO) is defined and supported via a reference to a metric condition already specified in the metric sub-model (e.g., see Fig. 10). In the financial use case, one SLO named *CPUMetricSLO* has been defined, which is associated to the *CPUMetricCondition*. The condition refers to the raw CPU utilization of any instance of the *QuorumAppServerComp* VM, which in the context of the Quorum application must be less than 70%, in order to trigger the corresponding scaling action.

Although not specified in this sample scenario, optimisation requirements can be defined in the requirement specification of the CAMEL model. These "soft" requirements define one or more optimisation objectives (e.g., maximise performance), the optimisation function to use, the metric to be considered, the Component on which the optimisation should be applied and the priority of each optimization requirement. The priority is needed, since these are "soft" requirements and as such they are prioritised, in order for the *Reasoning engine* component to treat these requirements as the most important ones. For instance, if two optimisation requirements were defined in the model: (i) minimise the response time for the application server and (ii) maximise storage availability, then the priority property for each one will indicate the importance level. A lower value will designate higher importance for that requirement and this should be considered by the *Reasoning engine* component in

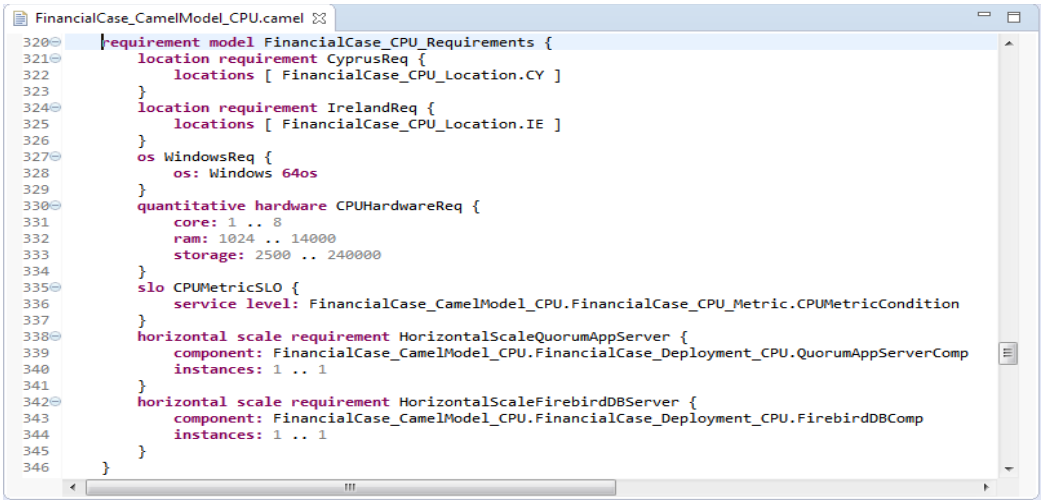

Fig. 12. The requirement specification of the CAMEL model.

the case of an adaptation. Hence, the utility function could be generated as the weighted sum of the functions of the metrics involved in the end-user's "soft" requirements. Therefore, these priorities allow the PaaSage platform to rank these requirements when reasoning on the application and generating a new cloud-provider specific model for an optimised deployment.

V. Conclusions

The market uptake of Cloud computing technologies created a vast set of Cloud tools and Cloud providers that strive to meet customers' high expectations. The advantage is that businesses have a variety of technologies and Cloud offerings to choose from, but at the end several challenges are faced by companies. Foremost, it creates interoperability challenges and promotes vendor lock-in, which is not desirable since businesses in many cases have a diversity of applications and systems. Therefore, businesses, as the business-oriented evaluation performed in this work confirmed, require flexibility and the possibility to select different providers for deploying different applications. In many cases Multi-Cloud deployment between private and public Clouds is also desirable. The evaluation also validated that another fundamental requirement from businesses is to be able to take informed and optimal decisions, as well as be able to efficiently monitor, adapt and optimise deployments.

This work proceeded with a study and analysis of existing cloud tools, including the PaaSage platform, in regards to the level of support they can offer for the key cloud features confirmed via the evaluation as important for business. These features can in fact enable businesses to overcome the aforesaid challenges and thus contribute to exploiting the benefits of the Cloud. Hence, the level of support of these features by Cloud tools contributes to the agility of the business, in terms of deployment, monitoring, adaptation and deployment of Cloud applications. Finally, the PaaSage platform is demonstrated, exemplifying the complete support for these features through a business case in the financial domain.

Acknowledgment

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number IP 317715: PaaSage: Model-based Cloud Platform Upperware (http://www.paasage.eu –) project.

References

[1] Nicolas Ferry, Hui Song, Alessandro Rossini, Franck Chauvel, Arnor Solberg, "CloudMF: Applying MDE to Tame the Complexity of Managing Multi-Cloud Applications", IEEE/ACM 7th International Conference on Utility and Cloud Computing (UCC 2014).

[2] Alessandro Rossini, Kiriakos Kritikos, Nikolay Nikolov, Frank Griesinger, Jörg Domaschka and Daniel Romero, "D2.1.3 – CAMEL Documentation (Final version)", PaaSage project deliverable (October 2015).

[3] Beniamino Di Martino, Giuseppina Cretella, and Antonio Esposito, "Advances in Applications Portability and Services Interoperability among Multiple Clouds", Second University of Naples, Italy, IEEE Cloud Computing, IEEE Computer Society, 2325-6095/15, 2015.

[4] Rawaa Qasha, Jacek Cała, and Paul Watson, "Towards Automated Workflow Deployment in the Cloud using TOSCA" , 8th IEEE International Conference on Cloud Computing, New York, USA (IEEE CLOUD), 2015.

[5] IBM, "Modeling with OPL: Represent business problems mathematically in order to create effective analytical decision support applications", Available Online (Last Accessed: 17 July 2015): http://www-01.ibm.com/software/commerce/optimization/modeling/.

[6] Object Management Group, "MDA – The Architecture Of Choice For A Changing World", Available Online: (Last Accessed: 21 July 2015): http://www.omg.org/mda/.

[7] Schmidt, D. C., Guest editor's introduction: Model-driven engineering.*Computer*, *39*(2), 0025-31, 2006.

[8] Jörg Domaschka, Kyriakos Kritikos and Alessandro Rossini. "Towards a Generic Language for Scalability Rules". In: Advances in Service-Oriented and Cloud Computing - Workshops of ESOCC 2014. Ed. by Guadalupe Ortiz and Cuong Tran. Vol. 508. Communications in Computer and Information Science. Springer, 2015, pp. 206–220. isbn: 978-3-319-14885-4. doi: 10.1007/978-3-319-14886-1_19.

[9] Kyriakos Kritikos, Jörg Domaschka and Alessandro Rossini. "SRL: A Scalability Rule Language for Multi-Cloud Environments". In: Cloud-Com 2014: 6th IEEE International Conference on Cloud Computing Technology and Science. Ed. by Juan E. Guerrero. IEEE Computer Society, 2014, pp. 1–9. isbn: 978-1-4799-4093-6. doi: 10.1109/ CloudCom.2014.170.