MODACLOUDS: A Model-Driven Approach for the Design and Execution of Applications on Multiple Clouds

Danilo Ardagna, Elisabetta Di Nitto
Politecnico di Milano, Italy
{*ardagna,dinitto*}*@elet.polimi.it*

Giuliano Casale
Imperial College London, UK
g.casale@imperial.ac.uk

Dana Petcu
Institute e-Austria Timisoara, Romania
petcu@info.uvt.ro

Parastoo Mohagheghi, Sébastien Mosser
SINTEF, Norway
{*Parastoo.Mohagheghi,Sebastien.Mosser*}*@sintef.no*

Peter Matthews
CA, Spain
Peter.Matthews@ca.com

Anke Gericke
BOC, Austria
Anke.Gericke@boc-group.com

Cyril Ballagny
SOFTEAM, France
cyril.ballagny@softeam.fr

Francesco D'Andria
ATOS, Spain
francesco.dandria@atos.net

Cosmin-Septimiu Nechifor
Siemens, Romania
septimiu.nechifor@siemens.com

Craig Sheridan
Flexiant, UK
csheridan@flexiant.com

***Abstract*—Cloud computing is emerging as a major trend in the ICT industry. While most of the attention of the research community is focused on considering the perspective of the Cloud providers, offering mechanisms to support scaling of resources and interoperability and federation between Clouds, the perspective of developers and operators willing to choose the Cloud without being strictly bound to a specific solution is mostly neglected.**

We argue that Model-Driven Development can be helpful in this context as it would allow developers to design software systems in a cloud-agnostic way and to be supported by model transformation techniques into the process of instantiating the system into specific, possibly, multiple Clouds. The MODACLOUDS (*MOdel-Driven Approach for the design and execution of applications on multiple Clouds*) approach we present here is based on these principles and aims at supporting system developers and operators in exploiting multiple Clouds for the same system and in migrating (part of) their systems from Cloud to Cloud as needed. MODACLOUDS offers a quality-driven design, development and operation method and features a Decision Support System to enable risk analysis for the selection of Cloud providers and for the evaluation of the Cloud adoption impact on internal business processes. Furthermore, MODACLOUDS offers a run-time environment for observing the system under execution and for enabling a feedback loop with the design environment. This allows system developers to react to performance fluctuations and to re-deploy applications on different Clouds on the long term.

***Keywords*-Cloud computing, model-driven development, performance, portability.**

I. INTRODUCTION

Cloud computing is emerging as a major trend in the ICT industry. The wide spectrum of available Clouds, such as those offered by Microsoft, Google, Amazon, HP, AT&T, and IBM, is contributing to a predicted compound annual growth rate of 19.5% [1] and provides a vibrant technical environment, where *small and medium enterprises* (SMEs) can create innovative solutions and evolve their existing service offer. Despite the flexibility of this environment, Cloud business models and technologies are in their initial hype stage and are characterised by many critical early stage issues which pose specific challenges from a software engineering perspective. Specifically, *"one of the most pressing issues with respect to Cloud computing is the current difference between the individual vendor approaches, and the implicit lack of interoperability [...]. Whilst a distributed data environment (*Infrastructure-as-a-Service *or IaaS) cannot be easily moved to any platform provider (*Platform-as-a-Service *or PaaS) and may even cause problems to be used by a specific service (*Software-as-a-Service *or SaaS), it is also almost impossible to move a service / image / environment between providers on the same level."* [2]. In this setting we can identify a number of challenges for systems developers and operators, especially for SMEs that have limited resources and do not have the strenght to influence the market. In particular:

1) *Vendor Lock-in*. The risk of technological lock-in is a major concern for Cloud customers [3], [4]. Cloud providers, in fact, offer proprietary solutions that force Cloud customers to decide, at the early stages of software development the design and deployment models to adopt (*e.g.,* public vs. hybrid Clouds) as well as the technology stack (*e.g.,* Amazon Simple DB vs. Google Bigtable). Lack of past expertise in Cloud computing makes this a high-risk choice, especially for SMEs. This, if the target platform does not fulfil the original expectations, has potentially catastrophic business consequences. Thus, portability of applications and data between Clouds, as well as reversibility (moving applications and data from Cloud to non-Cloud environments) should be addressed.

978-1-4673-1757-3/12/$31.00 © 2012 IEEE

2) *Risk Management*. There are several concerns when selecting a Cloud technology such as payment models, security, legal and contractual, quality and integration with the enterprises architecture and culture. Thus, proper tools to support such choice could be beneficial and limit serious financial consequences for a SME. However, while risk management has been well coded for traditional IT systems [5], at present only embryonic tools and decision support methods exists to support selecting and binding to a specific target Cloud, or taking a decision to move from Cloud to Cloud in case requirements or services change. Late binding [6] to a specific target Cloud can decrease project failure risks, but at the moment it is not supported neither at the design nor at the run-time level.
3) *Quality Assurance*. Cloud performance can vary at any point in time. Elasticity may not ramp at desired speeds. Unavailability problems exist even when 99.9% up-time is advertised (*e.g.,* Amazon EC2 and Microsoft Office 365 outages in 2011). Given the criticality of many business applications, analytical techniques are needed to predict QoS and to reason on software systems properties at design-time, but also run-time mechanisms and policies able to provide end-to-end quality.

We argue that all above mentioned challenges lead to the need for developers to be able to design their software systems for multiple Clouds and for operators to be able to deploy and re-deploy these systems on various Clouds depending on the convenience. The current Cloud literature, however, does not seem to pose attention to this issue as it is focused on considering the perspective of the Cloud providers, by offering mechanisms for auto-scaling of Clouds and for interoperability and federation between Clouds.

In this paper we present our research agenda and ongoing work for the development of MODACLOUDS, a *MOdel-Driven Approach for the design and execution of applications on multiple Clouds* that aims at supporting system developers and operators in exploiting multiple Clouds and in migrating their applications from Cloud to Cloud as needed. To do so, MODACLOUDS proposes an advanced quality-driven design, development and operation method based on the *Model-Driven Development* (MDD) paradigm. The model-driven approach allows one to *"model once and run everywhere."* In addition, early analysis of models regarding performance characteristics and user constraints is supported and drives the ability to migrate a software solution to multi-Clouds. Applications developed using the MODACLOUDS framework will be designed at a high-level to abstract from the targeted Cloud, and then semi-automatically translated into code able to run on multi-Cloud platforms. This frees developers from the need to commit to a fixed Cloud technology stack during software design and increases available options for reaching business targets in terms of cost savings, risk management, *Quality-of-Service* (QoS), and flexibility in the development process.

The MODACLOUDS framework is currently under development and this paper aims at highlighting the main ideas and design principles behind it.

II. MOTIVATING EXAMPLE

In order to provide the intuition of the benefits that a MDD approach can deliver to Cloud development, we consider a running example. In our example, MODAFIN is a European software development company specialised in IT applications for financial services. Its main product line is a proprietary solution for stock market operations, cash administration, and lending management. Related to this product, the most profitable activities are software customisation and life-cycle management.

The former involves development of custom modules to accommodate new functional requirements. Customisation also involves application integration with existing databases and legacy business systems at the customer's site. In addition, customers in the trading sector require assured quality, such as high-availability for real-time calculations during market hours, and scalability and low operational costs for batch analytic workloads running after-hours. MODAFIN currently fulfils these quality requirements with a capacity management consultancy team following the application life-cycle at the customer's site.

With the adoption of Cloud, MODAFIN's customer requirements evolve rapidly. At night, customers want to run their batch analytic workloads at the cheapest operational costs of Amazon on-spot instances. During the day, calculation engines are expected to ramp-up computing power at an unprecedented pace when the stock market gets volatile. Moreover, some customer applications are collecting and processing stock market data directly on the Cloud using *PaaS* datastore services such as Google Bigtable and Amazon SimpleDB. At the same time, customers are cutting costs spent in consultancy services for life-cycle management as they are relying more and more on SaaS services.

To remain competitive, MODAFIN's solution must evolve to address all above requirements. To do so, the company needs to revise both the software development process and its life-cycle management services:

1) The solution must support integration of the existing legacy with a broad spectrum of customer PaaS/IaaS, and possibly, SaaS solutions in order to exploit customer data locality.
2) It needs to be replicated on several Clouds to provide quality assurance to avoid that availability or performance outages of a single Cloud provider would turn into a disaster for MODAFIN's own business.
3) It must also have a flexible architecture that could be adapted to new Cloud offers emerging in the next 5-10 years to adapt to changes of context and requirements.

4) The life-cycle management team, rather than providing support at the customer's site, must support a system deployed on multi-Clouds.

The MDD approach allows one to design, develop, and re-engineer existing components into software modules that operates directly on multiple Clouds. It is of paramount importance for MODAFIN. Workloads can be run on different platforms depending on the customer preferred PaaS datastore and hence enjoy the performance benefits of customer data locality. The MDD approach allows one to include in the design model a description of MODAFIN's legacy software and generates automatically interfaces and connectors for application integration with customer systems for all of the supported target platforms. A *Decision Support System* (DSS) can be used to determine which Cloud to adopt for hosting the different components of new solutions, comparing costs, risks, and analysing non-functional characteristics for each alternative provider, improving also the trust in Cloud solutions. Furthermore, a run-time management API (independent of Cloud vendors) could be beneficial to natively implement Cloud-to-Cloud migration and multi-Cloud load-balancing.

Finally, run-time operation can be easily integrated inside MODAFIN's software development process through a feedback system that automatically provides recommendation on the best design and deployment patterns for a new custom module, thus making it able to adapt to context and requirements changes. MODAFIN can then remain a leader in its sector and neutralise competitive advantages of emerging start-ups that have developed their solutions directly on the Cloud.

III. MODACLOUDS APPROACH

MODACLOUDS solution targets *system developers* and *operators* by providing them with tools that support the whole life-cycle phases of a Cloud-based software system.

A. General Overview

First, during the feasibility study, when various Cloud alternatives are analysed, a DSS enables developers to analyse and compare various Cloud solutions. In fact, developing applications for multi-Clouds may impact established enterprise procedures and business models. Metrics (e.g., business stability, on-going cost, or availability) are needed to quantify the notion of risk for a particular choice relatively to the ecosystem in which it will evolve. The decision models included in the DSS comprise risk management and take into account variability in Cloud resource prices across time (*e.g.,* Amazon spot instances), geographic location, performance, legal aspects, etc. It also includes guidelines for their extension. In developing the DSS, we take advantage of established model-driven risk and quality analysis methods such as CORAS [7] and PREDIQT [8], as well as Cloud business models [9].

During design, implementation and deployment, the MODACLOUDS *Integrated Development Environment* (*IDE*) supports a Cloud-agnostic design of software systems, the semi-automatic translation of design artifacts into code, and the automatic deployment on the targeted Clouds. The *run-time layer* offered by MODACLOUDS: *(i)* enables system operators to oversee the execution of the system on multiple Clouds; *(ii)* automatically triggers some adaptation actions (*e.g.,* migrate some system components from a IaaS to another offering better performance at that time); and *(iii)* provides run-time information to the design-time environment (the IDE) that can inform the software system evolution process. Figure 1(a) represents an overview of the MODACLOUDS solution. It shows the DSS at the top and the run-time layer at the bottom, and details the internal of the IDE that is at the core of the MODACLOUDS approach. As already mentioned, it follows a MDD approach and, thus, it allows system developers to construct/elaborate system design models at different levels of abstraction. At all levels, the IDE offers also mechanisms for estimating QoS and costs of the system under development in order to verify that the implementation matches the requirements. Moreover, it supports designers in understanding the implications of deploying, replicating and migrating data at one site and in making appropriate decisions on this issue. At the *Cloud-enabled Computation Independent Model* (CIM) level applications and data are designed by exploiting the main components typical for service-oriented applications [10] and by including the design artifacts information about non-functional requirements and constraints (see the top level part of Figure 1(b)). This last information is used by a reasoning plugin of the IDE that helps in optimising the matching between the target Cloud environments and application characteristics.

CIM artifacts are semi-automatically translated and further refined at the *Cloud-Provider Independent Model* (CPIM) level, where Cloud concepts (including IaaS, PaaS, and SaaS level elements and considering the specificities of public, private and hybrid deployments as well) are incorporated and kept abstract from any Cloud specific installation (see the middle level part of Figure 1(b)). This level incorporates and extends the REMICS approach for modelling Clouds [11] and includes some Cloud design patterns that will be identified from a study of the way applications exploit Cloud capabilities today.

Finally, CPIM artifacts are semi-automatically translated into a *Cloud-Provider Specific Model* (CPSM) that enables the creation of those deployment artifacts that are needed to install and operate the application on the selected Clouds (see the bottom level part of Figure 1(b)).

The framework will allow to define models through domain specific languages. Vertical transformation will be implemented with state-of-the-art approaches to support

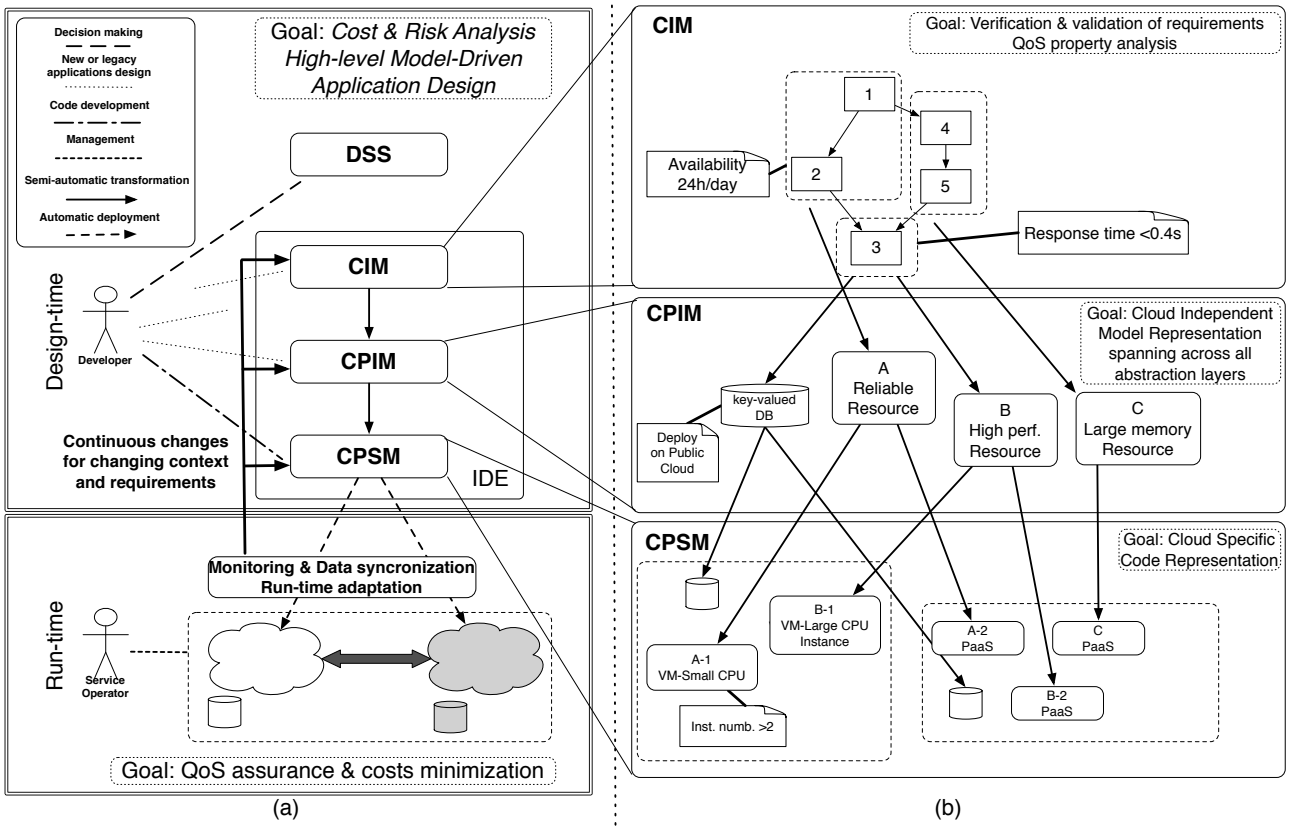

Figure 1. MODACLOUDS approach.

model transformation. Specialized approaches (e.g., ATL, Kermeta) as well as generic approaches (e.g., Scala) will be considered, targeting the most complete integration in the complete tool chain (i.e., the IDE). Some transformation can be fully automated, e.g., the generation of a deployment script based on a deployment descriptor artifact. But some transformation can only be semi-automated, i.e., the generation of skeleton instead of complete code. They will require an involvement of the user to provide additional information. The only way to avoid this is to capture more technical details in the abstract models, which is not an option w.r.t. our objective of abstraction at the Cloud model level.

MODACLOUDS provides also support for the migration of existing software into Cloud-based applications, extending the REMICS approach. In particular, it enables the exploitation of pre-existing application workload profiles for supporting the optimal partitioning of software components and data on multiple platforms.

A closed-loop between the run-time and design-time environments triggers the dynamic re-deployment of the final application or of its components. This way, system developers and operators are able to adapt their system to changing contexts and requirements both reacting to long-term failures of the Cloud providers and exploiting Cloud additional services or improved performance (*e.g.,* new virtual machine instances or reduced prices).

B. Applying the MODACLOUDS *approach*

We take our example again and consider now a simple application implementing a stock market transaction provided by MODAFIN to its customers. The CIM is reported in Figure 2. When an investment request is received, the application obtains information about several stocks from rating agencies, which are accessed as SaaS. Subsequently, the application determines the best stock market where to place the order (*Order analysis* operation) and the current stock prices are retrieved. In the following, the application places an order on behalf of the customer at the selected stock exchange. When the stock exchange acknowledge message is received (indicating the successful proceeding of the order transaction) the application performs a transaction on the customer trading account. At the CIM layer the application is annotated with its expected workload profile. Furthermore, performance and availability constraints are introduced. Availability requirements are of paramount importance for the application, which must be deployed on multi-Clouds. The annotations can be part of any other model at the business level such as business process models

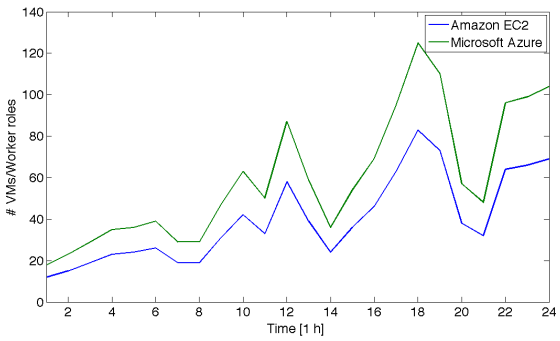

Figure 3. Computing Resources for the Trading Application.

or even use cases. Moreover, the analysis operation has to be performed extremely quickly (within 0.5s). Figure 2 shows how the CIM is mapped to the CPIM. Assuming that Amazon EC2 and Microsoft Azure are selected, the CPIM is translated into two CPSMs.

Figure 3 shows the planned number of EC2 instances and Azure Worker roles over a 24 hours period, assuming that the incoming workload is evenly shared among the two Cloud providers. This planning is determined as in [12], assuming a 20-30% degradation of the Cloud providers performance during the peak hours of the day [13]. The sizing has been obtained by modelling the VMs as M/G/1 processor sharing queues characterized by time varying parameters [14] and by observing that Azure Workers are 30% slower than the selected EC2 instances. This planning, determined at design time, is used and continuously updated at run-time to properly manage the application.

IV. Related Work

Modelling is a central part of all the activities that lead up to the deployment of high-quality software. Models are built for several reasons, *e.g.,* to communicate the desired structure and behaviour of a system or to visualise and control the system's architecture [15]. Popular formalisms exist to represent models and meta-models, for example the *Meta–Object Facility* (MOF) [16] and the *Eclipse Modelling Framework* (EMF) [17]. Modelling for Cloud-based applications is a new research topic. Preliminary ideas are proposed in the REMICS FP7 project [11], [18], however the REMICS approach only covers IaaS and defining Cloud abstractions (even at this layer) is far from being mature. OMG has proposed the *Knowledge Discovery Metamodel* (KDM) as part of the *Architecture Driven Modernization* (ADM) [19] initiative as a language for reverse engineering of applications. Software engineering challenges for moving to the Cloud have been the subject of recent research, for example in the FoSEC workshop [20] and are investigated in our research as well.

In particular we argue that architectural abstractions of Cloud artefacts are necessary to improve the understandability of the designed systems, promote better run-time control by including the quality measures, and for deploying applications on multi-Clouds.

From the side of quality evaluation at design time, modern *Model Driven Quality Prediction* (MDQP) techniques [21], [22] propose the following approach: *(i)* Designers work only at the abstraction levels which they are used to, possibly augmented with concepts and information about non-functional properties, *(ii)* Quality models are then automatically derived from high-level abstractions and solved by tools to predict the non-functional properties of interest. Such target quality models include *Queuing Network*s (QNs) [14], *Petri Net*s (PNs) [23], *Discrete Time Markov Chain*s (DTMCs) [24], and *Continuous Time Markov Chain*s (CTMCs) [25]. Notable examples of MDQP approaches are the *Performance by Unified Modeling* (PUMA) methodology [26], the *Palladio Component Model* (PCM) [27], and KlaperSuite [22]. Methodologies for MDQP automation include rule-based approaches, meta-heuristics, and approaches based on logic [28], [29], [30].

However, none of the above mentioned works consider multiple Clouds as the target for the deployment. In our work we address explicitly the peculiarities of Clouds, *e.g.,* the fluctuations in the customer workloads, burstiness and flash crowds phenomena, and the variability of Cloud performance. Furthermore, we also provide a framework for specifying QoS constraints at the CIM level and for estimating the QoS characteristics of applications deployed on multiple Clouds at the CPIM and CPSM levels. Proper feedback mechanisms and guidelines for the selection of Cloud solution that better suit the non-functional requirements and constraints defined for the application under development are also proposed.

If we consider run-time aspects, the increasing diffusion of the Cloud computing paradigm is fostering the creation of a new ecosystem of Cloud providers characterised by an extremely variegated technological offer. Multi-Cloud refers to the utilisation of more than one Cloud provider. According to [31] multiple Clouds can be organised in four flavours: *(i) Horizontal federation* where two or more providers join together to create a federated Cloud; in a horizontal federation participants who have exceeding resources or need additional resources can cooperate on a agree-upon price scheme, *(ii) InterCloud*s federation where Clouds with common addressing, naming, identity, trust, presence, messaging, multi-cast, time domain and application messaging are put together [32], [33], *(iii) Cross-Cloud*s where the federation establishment between a Cloud needing external resources and a Cloud offering resources (not necessarily in agreement), passes through three main phases [34]: discovery, looking for available Clouds; match-making, selecting the ones fitting the requirements; authentication, establishing a trust context with the selected Clouds, lastly, *(iv) Sky computing* where a broader usage of different Cloud

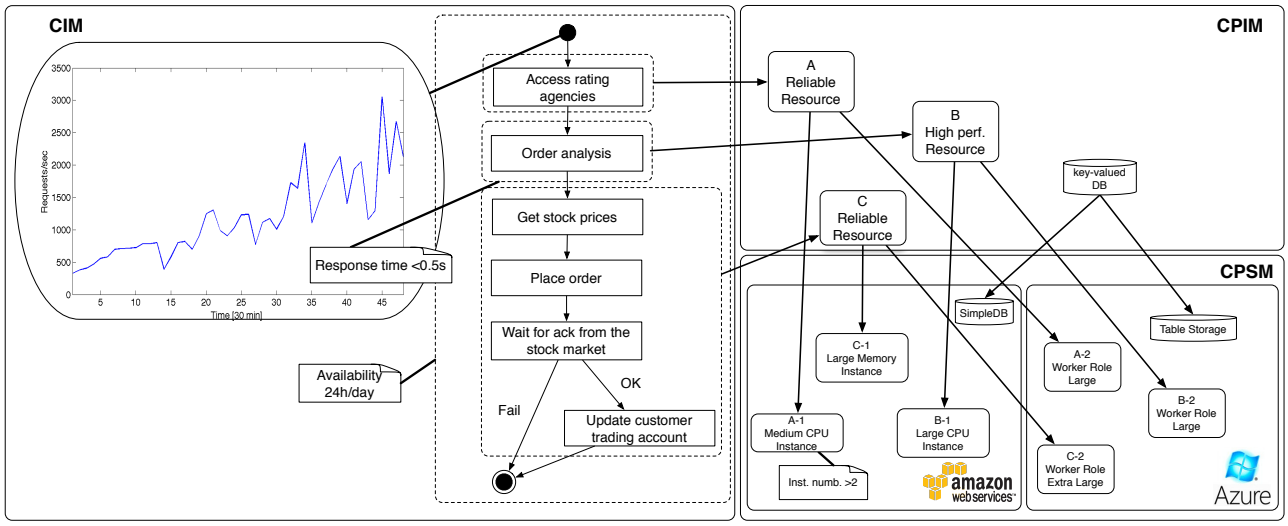

Figure 2. A Trading Application Design.

providers is envisaged by integrating resources, applications and platforms under a unique architecture. In this case the conjunction of multiple Cloud providers offers additional services obtained by the composition of existing services and thus the creation of "virtual data-centres" [35]. The mOSAIC [36] FP7 project is providing some solutions to achieve portability in Cloud federations providing APIs that abstract the IaaS services to a level that ensure an easy application migration between different Clouds.

Some commercial products have been proposed to simplify the use of multiple Clouds at run-time. For example, MultiCloud by RightScale [37] support the management of computing, storage, and networking resources of different Clouds (*e.g.,* AWS, CloudStack, Eucalyptus and Openstack).

As a closing remark we can state that MODACLOUDS aims at offering a comprehensive approach to the problem of designing software systems for multi-Cloud environments thus incorporating and harmonising many specific contributions that so far have been developed in separate and distant research communities.

V. Conclusions & Perspectives

This paper presents on-going research on implementing a framework and IDE for developing and deploying applications in multi-Clouds. The MDD approach abstracts the complexity of Cloud platforms and allows early definition and assessment of quality at design time. A decision support system enables risk analysis for the Cloud provider selection and for the evaluation of the impact of the Cloud adoption on internal business processes. The run-time environment monitors the applications in the Clouds and provides feedback for adaptation and optimization. Our ambition is to provide a platform for development, deployment, monitoring, and adaptation (both at run-time and in the long term) of future Internet applications in Clouds, with support for development from scratch or from legacy systems.

In our research, we consider also the more complex case of distributed deployment, when a certain application may be distributed across two or more Cloud providers adopting different technologies. In order to ensure the independence from the provider offer we are developing an abstract API for the deployment and management of multiple Clouds.

Acknowledgment

The authors would like to thank Eser Ay, Harald Kühn, Andrey Sadovykh, and Tabassum Sharif for many fruitful discussions. This work is partially funded by the EU Commission in the FP7 programme through the REMICS project (http://remics.eu), contract number 257793, the MOSAIC project (http://www.mosaic-cloud.eu/), contract number 256910, and the S-CUBE NoE (http://www.s-cube-network.eu/), contract number 215483.

References

[1] Pierre Audoin Consultants SAS, "Economic and Social Impact of Software & Software-Based Services http://cordis.europa.eu/fp7/ict/ssai/docs/20090730-d2-eu-ssbs-industry_en.pdf," 2011.

[2] SSAI Expert Group Report, "The Future of Cloud Computing http://cordis.europa.eu/fp7/ict/ssai/docs/cloud-report-final.pdf," 2010.

[3] Gartner, "2012 Cloud Computing Planning Guide http://my.gartner.com/portal/server.pt?open=512&objID=249&mode=2&PageID=864059&resId=1837017&ref=Browse," 2012.

[4] Forbes, "Cloud Computing's Vendor Lock-In Problem: Why the Industry is Taking a Step Backward http://www.forbes.com/sites/joemckendrick/2011/11/20/cloud-computings-vendor-lock-in-problem-why-the-industry-is-taking-a-step-backwards/," 2011.

[5] G. Stoneburner, A. Goguen, , and A. Feringa, "Recommendations of the National Institute of Standards and Technology http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf," NIST, Tech. Rep., 2002.

[6] S. Padhi, H. Pi, I. Sfiligoi, and F. Wuerthwein, "Use of late-binding technology for workload management system in CMS," in *Nuclear Science Symposium Conference Record (NSS/MIC), 2009 IEEE*, 2009.

[7] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis: The CORAS Approach*, 1st ed. Springer Publishing Company, Incorporated, 2010.

[8] A. Omerovic and K. Stølen, "Interval-Based Uncertainty Handling in Model-Based Prediction of System Quality," in *Second International Conference on Advances in System Simulation (SIMUL)*, August 2010, pp. 99 –108.

[9] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb 2009, http://www.eecs.berkeley.edu/Pubs/TechRpts/2009/EECS-2009-28.html.

[10] M. Colombo, E. Di Nitto, M. Di Penta, D. Distante, and M. Zuccalá, "Speaking a Common Language: A Conceptual Model for Describing Service-Oriented Systems," in *ICSOC 2005*, 2005.

[11] P. Mohagheghi, A.-J. Berre, A. Henry, F. Barbier, and A. Sadovykh, "REMICS- REuse and Migration of Legacy Applications to Interoperable Cloud Services - REMICS Consortium," in *ServiceWave*, 2010, pp. 195–196.

[12] D. Ardagna, B. Panicucci, and M. Passacantando, "A Game Theoretic Formulation of the Service Provisioning Problem in Cloud Systems," in *WWW*, S. Srinivasan, K. Ramamritham, A. Kumar, M. P. Ravindra, E. Bertino, and R. Kumar, Eds. ACM, 2011, pp. 177–186.

[13] Bitpipe, "Cloud performance from the end user perspective http://www.bitcurrent.com/download/cloud-performance-from-the-end-user-perspective/," 2011.

[14] E. D. Lazowska, J. Zahorjan, G. S. Graham, and K. C. Sevcik, *Quantitative System Performance: Computer System Analysis Using Queueing Network Models*. Prentice Hall, 1984.

[15] G. Booch, J. Rumbaugh, and I. Jacobson, *Unified Modeling Language User Guide, The (2nd Edition) (Addison-Wesley Object Technology Series)*. Addison-Wesley Professional, 2005.

[16] OMG, *Meta Object Facility (MOF) Core Specification Version 2.0*, 2006. [Online]. Available: http://www.omg.org/cgi-bin/doc?formal/2006-01-01

[17] The Eclipse Foundation, "Eclipse Modeling Framework (EMF)," http://www.eclipse.org/modeling/emf/, 2011.

[18] A. Sadovykh, C. Hein, B. Morin, P. Mohagheghi, and A. J. Berre, "REMICS- REuse and Migration of Legacy Applications to Interoperable Cloud Services," in *ServiceWave*, 2011, pp. 315–316.

[19] OMG, *Architecture Driven Modernization*, http://adm.omg.org/.

[20] IEEE, "The IEEE International Workshop on the Future of Software Engineering FOR and IN the Cloud http://www.cs.bham.ac.uk/~rzb/CloudWorkshop.htm," 2011.

[21] H. Koziolek, "Performance Evaluation of Component-based Software Systems: A Survey," *Perform. Eval.*, vol. 67, no. 8, pp. 634–658, 2010.

[22] A. Ciancone, A. Filieri, M. L. Drago, R. Mirandola, and V. Grassi, "KlaperSuite: An Integrated Model-Driven Environment for Reliability and Performance Analysis of Component-Based Systems," in *TOOLS*, ser. LNCS, vol. 6705, 2011, pp. 99–114.

[23] C. A. Petri, "Kommunikation mit Automaten." Ph.D. dissertation, University of Bonn, 1962.

[24] R. C. Cheung, "A User-oriented Software Reliability Model," in *Computer Software and Applications Conference, 1978. COMPSAC '78. The IEEE Computer Society's Second International*, 1978, pp. 565 – 570.

[25] A. Immonen and E. Niemela, "Survey of Reliability and Availability Prediction Methods from the Viewpoint of Software Architecture," *Software and Systems Modeling*, vol. 7, pp. 49–65, 2008, 10.1007/s10270-006-0040-x. [Online]. Available: http://dx.doi.org/10.1007/s10270-006-0040-x

[26] M. Woodside, D. C. Petriu, D. B. Petriu, H. Shen, T. Israr, and J. Merseguer, "Performance by Unified Model Analysis (PUMA)," in *WOSP*. ACM, 2005, pp. 1–12.

[27] S. Becker, H. Koziolek, and R. Reussner, "Model-Based Performance Prediction with the Palladio Component Model," in *WOSP*. ACM, 2007, pp. 54–65.

[28] V. Cortellessa, A. Martens, R. Reussner, and C. Trubiani, "A Process to Effectively Identify "Guilty" Performance Antipatterns," in *FASE*, 2010.

[29] D. E. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley, 1989.

[30] B. Eames, S. Neema, and R. Saraswat, "DesertFD: a Finite-domain Constraint Based Tool for Design Space Exploration," *Design Automation for Embedded Systems*, vol. 14, pp. 43–74, 2010, 10.1007/s10617-009-9049-z. [Online]. Available: http://dx.doi.org/10.1007/s10617-009-9049-z

[31] D. Petcu, "Portability and Interoperability between Clouds: Challenges and Case Study," in *Towards a Service-Based Internet*, ser. Lecture Notes in Computer Science, W. Abramowicz, I. Llorente, M. Surridge, A. Zisman, and J. Vayssire, Eds. Springer, 2011, vol. 6994, pp. 62–74.

[32] D. Bernstein, E. Ludvigson, K. Sankar, S. Diamond, and M. Morrow, "Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability," in *Internet and Web Applications and Services, 2009. ICIW '09. Fourth International Conference on*, may 2009, pp. 328 –336.

[33] R. Buyya, R. Ranjan, and R. N. Calheiros, "InterCloud: Utility-Oriented Federation of Cloud Computing Environments for Scaling of Application Services," in *ICA3PP (1)*, ser. Lecture Notes in Computer Science, C.-H. Hsu, L. T. Yang, J. H. Park, and S.-S. Yeo, Eds., vol. 6081. Springer, 2010, pp. 13–31.

[34] A. Celesti, F. Tusa, M. Villari, and A. Puliafito, "Three-Phase Cross-Cloud Federation Model: The Cloud SSO Authentication," in *Advances in Future Internet (AFIN), 2010 Second International Conference on*, july 2010, pp. 94 –101.

[35] K. Keahey, M. Tsugawa, A. Matsunaga, and J. A. Fortes, "Sky computing," *Internet Computing, IEEE*, vol. 13, no. 5, pp. 43 –51, sept.-oct. 2009.

[36] D. Petcu, G. Macariu, S. Panica, and C. Craciun, "Portable cloud applicationsFrom theory to practice," *Future Generation Computer Systems. In press*, 2012.

[37] Rightscale, "Multicloud platform," http://www.rightscale.com/products/multicloud-platform.php.