Enough About Standardization, Let's Build Cloud Applications

Javier Miranda, Joaquín Guillén,
Juan Manuel Murillo
Gloin
Calle de las Ocas 2, Cáceres, Spain
{jmiranda, jguillen, jmurillo}@gloin.es

Carlos Canal
Department of Computer Science
University of Málaga, Spain
canal@lcc.uma.es

ABSTRACT

The heterogeneous solutions provided by different cloud vendors have led to application interoperability and migratability issues. Companies that consume cloud services are tied to a single cloud provider due to the high costs of migrating software in the cloud. However, depending on the nature, size and interests of each company, different motivations can lead them to change their cloud provider or to have multiple providers. Several approaches have been proposed to deal with this problem, mainly based on the adoption of standards or the use of middlewares for creating an abstraction between the software and the clouds. However neither of these approaches have yet been consolidated. In this paper an alternate approach is presented for modeling and developing cloud applications, based on MDE and software adaptation techniques. The proposed solution is both cloud vendor and user friendly as it allows the former to freely define their own cloud policies, whilst users continue to be free to choose a cloud provider, even after the application has been developed.

Categories and Subject Descriptors

D.2.2 [**Software Engineering**]: Design Tools and Techniques; D.2.12 [**Software Engineering**]: Interoperability; I.6 [**Simulation and Modeling**]: Model Development

Keywords

service, component, cloud, adaptation, MDE, interoperability vendor lock-in

1. INTRODUCTION

Cloud computing has experienced a tremendous hype and it is now consolidating [11] itself as a solid and reliable means of providing computing resources in an easy and accessible manner. This consolidation will highly depend on how robust the technology is, and a critical factor in this scope will be to determine whether the technology satisfies the requirements of companies interested in it.

Companies interested in cloud will range from those that deploy complex services and architectures which they want to keep under a strict control, to those that deploy smaller public services that provide basic functionalities, such as storage, security, social services, etc. The former will commonly have a limited amount of users for their applications: their interest in cloud computing technologies will mainly be oriented towards lowering their costs and delimiting the investments made on IT. The latter will try to attract a great number of users for their applications: their interest in cloud computing will be focused to accessing a high performance platform that can guarantee a high scalability and availability.

In order to achieve their goals, companies in the first scenario will migrate parts of their technological infrastructure to cloud platforms and will seek to maintain interoperability between the new cloud applications and those kept in their in-house datacentres. For the second scenario, a critical factor will be to be free to duplicate or even migrate services to new clouds where their performance, availability and/or productivity can be rewarded.

However, companies may find that achieving these goals is not as easy as it seems: cloud application migratability and interoperability are currently hot issues that are being strongly questioned [1, 6]. The great potential of this technology and its growing acceptance have resulted in the appearance of multiple cloud vendors which provide similar services. The variability found at different levels of these services has resulted in the vendor lock-in effect [7, 13], which has been aggravated by the little effort that has been made on behalf of cloud vendors to favour interoperability between services and applications hosted by different clouds.

Cloud standardization has been presented as the solution that will mitigate these issues. Different organizations and enterprise forums have taken a first step towards defining standards for homogenizing and integrating the existent cloud platforms [14, 4]. However, a generalized consensus has not been reached in these proposals and neither of them has been widely adopted. Other alternatives, primarily based on the use of middlewares, for providing an abstraction between the applications and the cloud platforms, have also become very popular. These approaches constitute a valid solution for cloud application migratability and interoperability, however applications continue to be coupled to an underlying entity which in this case is the middleware itself.

As a continuation of the work carried out in [9], this paper reviews the main standardization initiatives and the most relevant alternatives based on middlewares that have been proposed. It also presents an alternative approach for developing applications that are decoupled from specific cloud platforms by integrating MDE and Software Adapta-

WICSA/ECSA '12, Aug 20-24 2012 Helsinki, Finland
ACM 978-1-4503-1568-5/12/08.

tion (SA) techniques. This provides developers with a means of creating software that can be deployed in any cloud as if it were developed for an in-house environment. Furthermore, considering that the software is cloud-agnostic, legacy software can easily be migrated to the cloud.

This paper is structured as follows. Section 2 presents the motivation of our work, describing the most relevant standardization and middleware-based solutions. Section 3 presents our approach for developing and deploying cloud agnostic software. Section 4 contains a description of works that are related to ours; finally, Section 5 presents our conclusions and future lines of work.

2. MOTIVATIONS

Cloud vendors tend to have their own implementation and specification of the services that they provide, locking the users into their solutions and preventing the portability of the applications to other providers. This situation threatens the success of cloud computing as a universal service where users can switch between providers as they please [12].

To achieve an interoperable cloud scenario, we differentiate two kinds of initiatives that have been carried out: standardization and alternative interoperability frameworks.

Standardization is being promoted by standardization bodies, not-for-profit organizations and enterprise forums seeking to homogenize cloud solutions by proposing standard APIs[1], cloud model definitions[2], and broker-based marketplaces[3]. All of these attempts are focused on establishing common foundations and definitions for cloud vendors to provide their services uniformly.

The lack of support for these standardization attempts on behalf of cloud vendors suggests other alternatives that provide immediate results are more likely to be adopted in the short term. Furthermore, consolidating more than one of these proposals would provoke interoperability problems between vendors applying different standards.

In the latter case, other approaches have risen as an alternative to standardization. They concur in the need of a common API that could unify the heterogeneous and specific core functionalities provided by each cloud vendor. An alternative technique is the use of service brokering and the introduction of abstraction layers or middlewares which would isolate the specific vendor implementations from the common definition of similar services throughout cloud providers.

These solutions cover some of the main shortcomings of cloud interoperability, however they imply the use of heavyweight middleware systems through which all the traffic of component communications must pass through. Furthermore, they shift the lock-in effect from the vendor to the middleware platform.

In this paper, an alternative approach is presented introducing two main benefits: firstly, it allows cloud providers to leave standards unattended, favouring their freedom of choice regarding the strategies and solutions adopted; secondly, this proposal frees customers from having to choose a unique provider and the technical dependencies with middleware systems that could introduce performance penalizations and crop the user business future in case of a platform discontinuation.

Using MDE and Software Adaptation (SA) techniques we present a cloud provider and customer friendly approach that allows an unprecedented flexibility of choice.

[1]CDMI (http://www.snia.org/cdmi) or OCCI (http://occi-wg.org)

[2]DMTF Cloud Standards (http://www.dmtf.org/standards/cloud)

[3]CIF (http://www.cloudindustryforum.org)

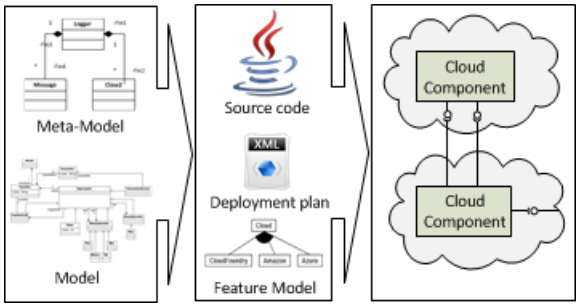

Figure 1: Cloud application development overview

3. APPROACH

In Figure 1 we can appreciate the development phases covered by our proposal and the different software engineering paradigms that have been considered. Each phase is detailed in the following subsections.

3.1 Application modeling

During this phase a cloud application will be modeled for which the source code and a cloud deployment plan will be generated through model-to-text transformations. A very simple extension of the UML base meta-model has been defined in order to include a construct that allows software components to be grouped with one another. Based on this meta-model cloud applications will be modeled as groups of components, further referenced as cloud artefacts, taking into account the following considerations:

- Each cloud artefact must be considered as an atomic software entity deployed in a single cloud platform.
- Components that belong to the same cloud artefact will communicate with one another locally; i.e. no mediation will be required.
- Components that belong to different cloud artefacts will communicate with one another remotely; i.e. mediation will be required.
- Components may consume external services provided by other applications

A model-to-text transformation will be applied upon the models created during this stage in order to generate class skeletons that will establish the basic structure of the application. These class skeletons will be cloud agnostic; i.e. all cloud related information will be generated separately in a XML formatted cloud deployment plan.

3.2 Coding and deployment configuration

During this phase developers will code the application's functionality starting from the skeleton classes generated in the previous phase. Additionally, tools will be integrated in the development environment in order to allow them to configure the cloud deployment plan, where each artefact will

be assigned to a specific cloud platform. Cloud specific information regarding the following points will automatically be included in the deployment plan per cloud artefact.

- Services provided and consumed by the cloud artefact for interoperability with components deployed in different clouds
- Vendor specific core services consumed by the cloud artefact

The technological restrictions of each cloud platform regarding these points will also be automatically generated and included in the deployment plan in order to be able to generate cloud compliant software components during the next phase. This is possible because the approach encloses a feature model (an engineering paradigm frequently used in the scope of Software Product Lines) that documents the variability of each cloud platform, thereby providing a knowledge base that contains the specific features of each cloud platform (supported service protocols, configuration parameters, cloud specific services, etc).

We have chosen to include this intermediate step instead of directly generating the cloud compliant source code in order to make it easier for developers to code the application's functionality. Directly generating the source code used for component-to-component or component-to-cloud interoperability would hinder the development process by forcing developers to integrate their source code into complex generated code and it would also be detrimental for the application's maintainability. Furthermore, encapsulating cloud relative information in a cloud deployment plan will also favor migrating legacy software to the cloud as no changes will be required upon the source code.

3.3 Cloud artefact and adapter generation

The source code and the deployment plan generated in the previous phase will be processed during this stage to generate cloud compliant artefacts. Considering that each cloud platform can impose a different structure for its software projects, the cloud artefacts generated in this phase will be enclosed in predefined source code projects that also contain a series of software adapters.

As part of this phase, we propose to use Software Adaptation techniques as a flexible solution for overcoming variability and interoperability issues in clouds. SA techniques are aimed at developing mediator elements, called adapters, which are automatically built from their correspondent specifications and granting interoperability between mismatching software elements [3].

Adaptation will be performed at any of its four different levels of interoperability [2], depending on the needs of each cloud artefact. At the **signature** level, mismatch is related to the location, the names and/or the parameters of cloud services or component operations. This adaptation could be required between artefacts deployed in different clouds, and between cloud artefacts and their underlying vendor specific core services. At the **behavioural** level, adapters deal with mismatch in the granularity between service requests and responses. For instance, a single service operation defined in a particular cloud environment could imply multiple service operations in other clouds. It also addresses issues related to protocol mismatch (i.e. different ordering in operations) assumed by the services involved. At the **service** level, non-functional issues, such as aspects related with security needs, cost and performance requirements, or SLA negotiation, are dealt. Lastly, at the **semantic** level, adaptation aims at service discovery based on semantic descriptions and motivated by a particular artefact requiring external semantically defined services.

Different methodologies and techniques are proposed in the SA field in order to solve each of the aforementioned sources of mismatch [3] [15], like the use of mappings that establish correspondences between different names at the signature level, or the combination of signature mappings and protocol specifications at the behavioural level.

Detecting mismatches at any of the levels above, and the automatic generation of the required adaptors will help to achieve interoperability between cloud artefacts deployed through heterogeneous cloud providers.

4. RELATED WORK

Different proposals have been made to mitigate the absence of standardization. This section describes the cloud application migratability and interoperability proposals that are most closely related to ours.

In the scope of MDE for cloud applications, [10] proposes a meta-model for modeling cloud applications focused in the definition of cloud tasks as composable units, each one consisting of a set of actions that make use of services to provide a specific functionality. The approach detaches the application modeling process from specific cloud platforms; nevertheless, model transformations will generate code that will be coupled to a specific cloud platform. Our approach models cloud applications from a different perspective since it allows us to generate source code that is not coupled to the cloud. Instead, all cloud related data is included in a separate deployment plan, thereby favouring the code's cloud agnosticism and maintainability.

Another proposal based on MDE techniques is presented by [8] as a means of mapping models of existent cloud environments to legacy software models and transforming the result to cloud-specific code through a series of iterations and result evaluations. This approach is fully oriented towards legacy software; additionally, any subsequent changes will have to be integrated into the generated software, which may result more difficult to work with and understand by the developers. In our work a different approach for modeling the variability of cloud platforms, based on feature models, is used. The cloud migration possibilities are not limited to legacy software exclusively; both legacy and newly developed software can be migrated easily to the cloud. Furthermore our work does not couple the application's source code to any cloud platform.

Other proposals for combatting the vendor lock-in effect are based on the use of middlewares and intermediate software layers intended to create an abstraction between cloud platforms and the generated software. One of the closest to our work is mOSAIC[4], a reference initiative carried out as a Europe funded project. Its perspective of how cloud development should be accomplished matches our criteria. Cloud migratability, interoperability and the deployment of applications across more than one cloud is tackled through a robust solution based on an API and a middleware platform for cloud brokering [5]. Cloud applications developed using this approach will be coupled to the middleware platform,

[4]mOSAIC Framework (http://www.mosaic-cloud.eu)

whereas in our proposal the use of automatically generated adapters allows applications to be decoupled from the underlying platform. Maintainance of a handcrafted middleware will also be more complex than the one required for automatically generated software adapters.

A Service Oriented Cloud Computing Architecture called SOCCA is presented in [16], where cloud computing resources are componentized, standardized and combined in order to build a "cross-platform virtual computer" which operates upon an ontology mapping layer that is used to abstract the differences between cloud providers. SOCCA applications can be developed using the standard interfaces provided by the architecture or the platform unique APIs of a cloud provider. In both cases the developed applications will be coupled to a specific platform, thereby hindering their migration to alternate scenarios. In our approach the applications are not coupled to any platform; the use of SA allows us to easily migrate components between clouds without having to modify their source code.

5. CONCLUSIONS AND FUTURE WORK

In this paper we have presented an alternative approach to the existent standardization initiatives and middleware-based solutions for achieving cloud application migratability and interoperability.

This solution has been conceived with cloud providers and customers in mind, trying to give them maximum flexibility. The underlying technology and product is currently being developed by Gloin, a small Spanish company. Gloin presents this work at NordiCloud as part of the company's strategy, which includes the possibility of entering the Nordic market and finding partners or companies that may be interested in this solution.

Our future work includes the extension of the current capabilities, supporting more cloud platforms and programming languages, as well as exploring the possibilities of using dynamic adaptors in order to support run-time changes in cloud artefact compositions.

ACKNOWLEDGMENTS

This work has been partially funded by the Spanish Ministry of Science and Innovation under Projects TIN2011-24278, and TIN2008-05932, as well as by Junta de Extremadura and FEDER funds.

6. REFERENCES

[1] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, Apr. 2010.

[2] C. Canal, J. Murillo, and P. Poizat. Coordination and adaptation techniques for software entities. In J. Malenfant and B. Østvold, editors, *Object-Oriented Technology. ECOOP 2004 Workshop Reader*, volume 3344 of *Lecture Notes in Computer Science*, pages 133–147. Springer Berlin / Heidelberg, 2005.

[3] C. Canal, P. Poizat, and G. Salaun. Model-based adaptation of behavioral mismatching components. *Software Engineering, IEEE Transactions on*, 34(4):546 –563, july-aug. 2008.

[4] A. Celesti, F. Tusa, M. Villari, and A. Puliafito. How to enhance cloud architectures to enable cross-federation. In *Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on*, pages 337 –345, july 2010.

[5] B. Di Martino, D. Petcu, R. Cossu, P. Goncalves, T. Máhr, and M. Loichate. Building a mosaic of clouds. In *Euro-Par 2010 Parallel Processing Workshops*, volume 6586 of *Lecture Notes in Computer Science*, pages 571–578. Springer Berlin / Heidelberg, 2011.

[6] T. Dillon, C. Wu, and E. Chang. Cloud computing: Issues and challenges. In *Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on*, pages 27 –33, april 2010.

[7] M. P. D.K. Nguyen, Y. Taher and W. van den Heuvel. Service-based application development on the cloud. state of the art and shortcomings analysis. In *CLOSER 2012 Proceedings*, pages 395–400, 2012.

[8] S. Frey and W. Hasselbring. Model-Based Migration of Legacy Software Systems into the Cloud: The CloudMIG Approach. In *Proceedings of the 12th Workshop Software-Reengineering (WSR 2010)*, pages 59–60, May 2010.

[9] J. Guillén, J. Miranda, and J. M. Murillo. Decoupling cloud applications from the source - a framework for developing cloud agnostic software. In *CLOSER*, pages 70–75, 2012.

[10] M. Hamdaqa, T. Livogiannis, and L. Tahvildari. A reference model for developing cloud applications. In *CLOSER*, pages 98–103. SciTePress, 2011.

[11] N. Leavitt. Is cloud computing really ready for prime time? *Computer*, 42(1):15–20, Jan. 2009.

[12] N. Loutas, E. Kamateri, F. Bosi, and K. Tarabanis. Cloud computing interoperability: The state of play. *Cloud Computing Technology and Science, IEEE International Conference on*, 0:752–757, 2011.

[13] D. Petcu, G. Macariu, S. Panica, and C. Crăciun. Portable cloud applications - from theory to practice. *Future Generation Computer Systems*, Jan. 2012.

[14] B. Rochwerger, D. Breitgand, E. Levy, A. Galis, K. Nagin, I. M. Llorente, R. Montero, Y. Wolfsthal, E. Elmroth, J. Cáceres, M. Ben-Yehuda, W. Emmerich, and F. Galán. The reservoir model and architecture for open federated cloud computing. *IBM J. Res. Dev.*, 53(4):535–545, July 2009.

[15] R. Seguel, R. Eshuis, and P. Grefen. Generating minimal protocol adaptors for loosely coupled services. *Web Services, IEEE International Conference on*, 0:417–424, 2010.

[16] W.-T. Tsai, X. Sun, and J. Balasooriya. Service-oriented cloud computing architecture. In *Information Technology: New Generations (ITNG), 2010 Seventh International Conference on*, pages 684 –689, april 2010.