MDE-based Automated Provisioning and Management of Cloud Applications

Anirban Bhattacharjee*
*Department of Electrical Engineering and Computer Science
Vanderbilt University, Nashville, Tennessee, USA
Email: anirban.bhattacharjee@vanderbilt.edu

***Abstract*—Digital innovations in the modern era are driven by cloud-native applications, cloud architectures, and cloud-based development. In this realm, continuous architectural design, framework/infrastructure (re-)configuration, deployment, migration, and resource monitoring to iteratively tune the application and underlying resources as per business demand are required in the form of a self-service cloud deployment and management platform. To address these challenges while ensuring broader applicability using vendor-agnostic approaches, this doctoral research envisions an intelligent and context-aware, model-driven, automated service provisioning technique, where manual efforts and domain expertise are alleviated to a large extent. In this context, this paper describes a self-service cloud-based application deployment and management ecosystem based on a Model-Driven Engineering (MDE) approach.**

I. Introduction

Deployment and management of cloud-based applications in both homogeneous or heterogeneous cloud environments are often hampered by the intricacies and variabilities in the deployment and configuration processes stemming from the diversity in both the software stacks that control the cloud platforms and the underlying platform resources. Application provisioning is important since it enables the service providers to deploy their service's application components in the cloud provider's resources in accordance with the service's Quality of Services (QoS), availability, business policy, and cost model requirements [33], [28]. This complexity restricts the cloud application development and deployment knowledge to a narrow set of experts who possess both the domain knowledge and application-specific knowledge to explore the power of these techniques. Even with advances in DevOps community tools, they demand domain expertise and hence creating full-blown deployable models are error-prone and time-consuming.

To address these challenges, this doctoral dissertation envisions an intelligent and context-aware, model -driven, automated service provisioning platform where manual efforts and domain expertise are minimal. To that end, in this paper, we propose a model-driven engineering (MDE)-based ecosystem [29] to automate the process of deployment and management for cloud applications and emphasize on extensibility, (re-)usability and scalability. Our approach emphasizes vendor-neutrality in order to be broadly applicable to diverse cloud platforms with their control software stacks. In the remainder of this paper, we explore the challenges faced in realizing the MDE approach for deployment and management of cloud applications, the current state of the art, and propose solutions while providing some preliminary results.

II. Problem Formulation

Consider as an example a distributed, data-intensive application deployed in the cloud. It requires significant efforts for the designers -

1) To configure and deploy the application to use the available big data frameworks [31],
2) To set up the virtual machines to host the frameworks and finally, and
3) To establish the connection properly among application components.

Moreover, adding application component which requires a different framework exacerbates these problems. To address these problems, we envision an automated solution and outline the key requirements of such an approach and their associated technical challenges:

A. Requirement 1: Infrastructure Design Provisioning and Configuration Automation

Need to abstract away the architecting phase by pre-defining the configuration of applications and infrastructure as intuitive modeling artifacts, which will speed up the deployment and migration process.

- ***Challenge 1: Capturing the Application and Cloud Specifications in the Metamodel***, where all the deployment and infrastructure complexities of the application and cloud specification are identified and defined.
- ***Challenge 2: Defining a Language for Model Transformation***, where the DSML will transform the abstract business model to target model (i.e. infrastructure code) using a knowledge base.
- ***Challenge3: Verification of the Business Model***, where all relationships and constraints of application components are checked and satisfied.

B. Requirement 2: Support for Continuous Integration, Migration, and Delivery

Changes in business strategies and demands lead to evolution of the business model, and hence the platform should be integrated with adaptation techniques.

- ***Challenge 4: Extensibility and Reusability of the Application Components***, where the addition, upgradation or

migration of application components can be done at the model level, and should support collaboration and version control.
- ***Challenge 5: Extensibility of the Platform***, where the addition of a new application type should be performed in a modularized way, and should be only a one-time effort.

C. Requirement 3: Dynamic Modeling and Provisioning

Continuously monitoring the infrastructure of the system and applications are required, and hence the model should always reflect the current up-to-date state.

- ***Challenge 6: Context-Aware Provisioning***, where optimization of cloud resources based on cost and resource availability should take place iteratively for potentially differing QoS needs, policies, etc.
- ***Challenge 7: Model@Runtime***, where the changes stemming from self-adaptation of the provisioning techniques should be reflected back at the model level dynamically [4].

III. Related Work

We explore existing research efforts directly aligned with our research goals. The script-centric DevOps community provides toolchains such as Chef [26], Puppet [21], or Ansible[1] to speed up deployment for developers [18]. Cloudify and Apache Brooklyn[2] enable cloud application orchestration of topology templates in a vendor-agnostic way according to the Topology and Orchestration Specification for Cloud Applications (TOSCA) [27] specification. However, we need declarative high-level modeling abstraction on top of these tools to reduce the burden of domain expertise.

To satisfy Requirement 1, multiple pattern based approaches are proposed [2], [9], [8], [22] where a model-based pattern describes a set of capabilities and functional and non-functional properties of application service deployment in cloud infrastructures [15], [1]. Alternatively, requirement solvers [25], [7] are also proposed to synthesize infrastructure configuration in a declarative fashion. OpenTOSCA [14] and CELAR [12] exercise the combination of MDE and TOSCA specification to automate deployment of cloud applications from users' partial business relevant topology. The components can be deployed and managed in the right order [5]. Later in [6] a context-aware provisioning and management plan along with detection and correction of the unintended error due to software conflicts and dependency is proposed. However, their platform is not based on any standardization, and their planner does not use any DevOps community tools. Hence, their approach might lack extensibility and robustness.

In relation to Requirement 2, CHAMPS [19] focuses on Change Management process by which IT systems are efficiently modified to minimize reconfiguration impact. Similarly, various standard modeling environments as an object-oriented formalism for configuration problems is proposed in [10], [13]. For Requirement 3 [24], [17], dynamic resource management on enterprise application vendors is proposed by various frameworks based on queuing model [34], [32], or by dynamic QoS Control [11], [16], [30].

[1]https://www.ansible.com/

[2]https://brooklyn.apache.org/

IV. Proposed Solution

We propose a composable ecosystem called CloudCAMP to deploy, monitor and manage cloud application intelligently. The workflow of the proposed approach is shown in Fig. 1. The contributions in this research include the following tasks:

- *Task 1: A Deployment Framework Metamodel*, to automate the infrastructure design and implementation by capturing variability and commonality endpoints for the application and cloud specifications along with their SDK and APIs, to provide abstractions based on TOSCA-specifications. This metamodel details will be stored in the knowledge base of the platform.
- *Task 2: Model-to-Infrastructure-as-code Transformation DSML*, as the basis for the deployment time and runtime composition and orchestration through generative programming. The DSML will also be integrated with predefined constraints to verify the correctness of business model.
- *Task 3: An Integrated Repository*, that allows storage, query, and manipulation of models, templates, metadata, parameters, software dependency knowledge base and provenance in a scalable data store.
- *Task 4: A Model@Runtime framework*, that permits reflection of the current model by capturing and monitoring the streamlined system and application information across heterogeneous resources at runtime.
- *Task 5: Integration of Adaptive Resource Provisioning Engine*, to provide the QoS guarantees while scaling the application vertically and horizontally, based on policy, economic issues, etc.

The approach to integrating diverse sets of deployment and management building blocks for a specific application and cloud providers offers significant advantages over traditional methods – such as DevOps or orchestration frameworks – by applying a blend of generative programming and micro service architectures. Concrete implementation our proposed generative approach will be demonstrated in a cloud-based MDE environment called WebGME [23].

V. Current Status

Our CloudCAMP approach to integrating a diverse set of deployment and management building blocks for a specific application and cloud providers offers significant advantages over traditional methods – such as DevOps or orchestration frameworks – by applying a blend of generative programming and micro service architectures.

A concrete implementation of our MDE-based, generative approach called CloudCAMP is being developed within a cloud-based MDE environment called WebGME [23]. A preliminary version of CloudCAMP supports few application

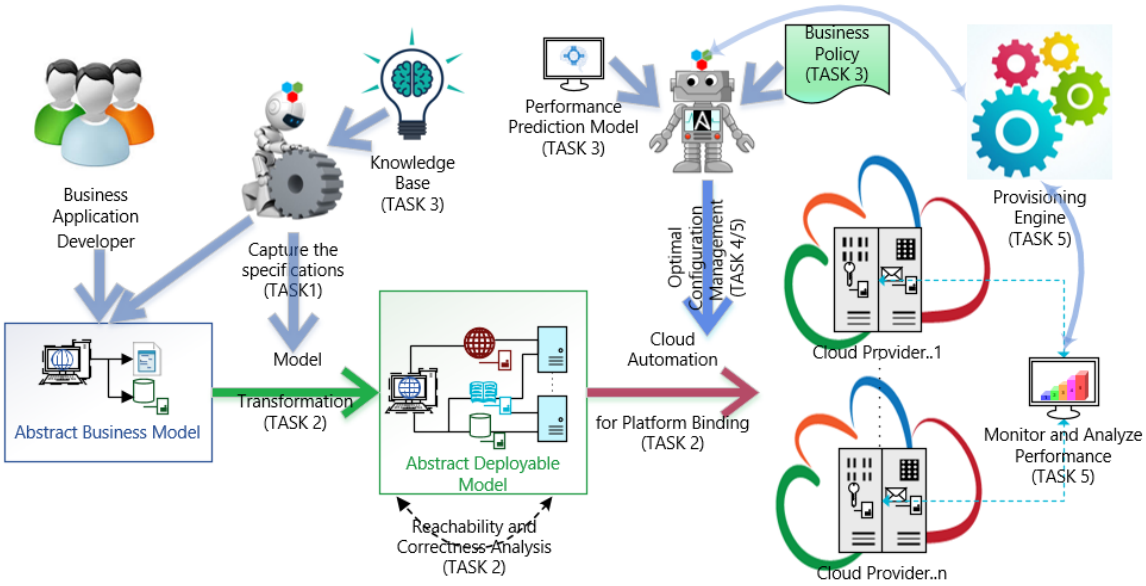

Fig. 1: The complete workflow of the proposed approach

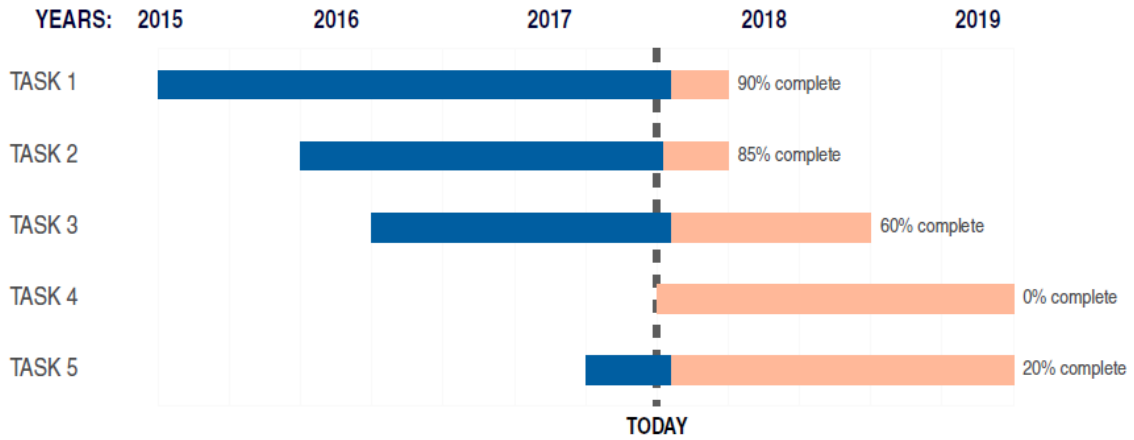

Fig. 2: THESIS Defense Timeline

types in an OpenStack cloud environment. The DSML transforms the business model to an infrastructure-as-code based on the metamodel and pre-defined knowledge base, which is a MySQL database. We have integrated a monitoring tool to check the VMs and applications health. Presently, our DSML can realize the CloudCAMP metamodel and generate Ansible specific infrastructure code.

The development of CloudCAMP comprises of the design and evaluation of the expressivity of the metamodel and the DSML with all the product-line features. Figure 2 provides a timeline for this research, conducted under the supervision of Dr. Aniruddha Gokhale, Associate Professor, Vanderbilt University.

VI. Plan for Evaluation and Validation

To demonstrate the deployment and management ecosystem, we are developing the CloudCAMP platform based on Generic Modeling Environment (GME) [20] as Task 1 and 2.

Our research evaluation plans call for validating the framework in the context of our ongoing and upcoming industry and federally sponsored research projects. The source code is available at https://anirban2404.github.io/DeploymentAutomation/. For example, one of our new projects is building a cloud-based machine learning ecosystem which illustrates many of the needs addressed by our research. We also plan to validate our framework in cloud-based, collaborative education services for STEM education [3].

VII. Expected Contributions

CloudCAMP will ease the deployment and management of applications based on deployment abstraction and intelligent resource optimization. The following key contributions are expected from this research:

The following key contributions are expected from this research:

1) *Automated generation of infrastructure code from an abstract business model in a standardized manner*. It can reduce deployment and migration time of application components enormously. Our CloudCAMP framework can be extended, and the application components are reusable and modularized.
2) *Business model verification*. Business model will be verified via a set of pre-defined constraints, which collected by reverse engineering and prototyping.
3) *Model@Runtime framework*. We will monitor and reflect the current system information in the model-level, so the business user can understand application design state and can override the architecture if necessary.
4) *Integration of Resource Management Techniques*. To build the context-aware self-service framework, we will integrate the resource management algorithm, which will trigger the addition or migration of application components to scale the system based on demand.

REFERENCES

[1] Ardagna, D., Di Nitto, E., Casale, G., Petcu, D., Mohagheghi, P., Mosser, S., Matthews, P., Gericke, A., Ballagny, C., D'Andria, F., et al.: Modaclouds: A model-driven approach for the design and execution of applications on multiple clouds. In: Proceedings of the 4th International Workshop on Modeling in Software Engineering. pp. 50–56. IEEE Press (2012)

[2] Arnold, W., Eilam, T., Kalantar, M., Konstantinou, A.V., Totok, A.A.: Pattern based SOA deployment. Springer (2007)

[3] Barve, Y., Patil, P., Bhattacharjee, A., Gokhale, A.: Pads: Design and implementation of a cloud-based, immersive learning environment for distributed systems algorithms. IEEE Transactions on Emerging Topics in Computing PP(99), 1–1 (2017)

[4] Blair, G., Bencomo, N., France, R.B.: Models@ run. time. Computer 42(10) (2009)

[5] Breitenbücher, U., Binz, T., Képes, K., Kopp, O., Leymann, F., Wettinger, J.: Combining declarative and imperative cloud application provisioning based on tosca. In: Cloud Engineering (IC2E), 2014 IEEE International Conference on. pp. 87–96. IEEE (2014)

[6] Breitenbücher, U., Binz, T., Kopp, O., Leymann, F., Wieland, M.: Context-aware provisioning and management of cloud applications. In: International Conference on Cloud Computing and Services Science. pp. 151–168. Springer (2015)

[7] Di Cosmo, R., Eiche, A., Mauro, J., Zacchiroli, S., Zavattaro, G., Zwolakowski, J.: Automatic deployment of services in the cloud with aeolus blender. In: International Conference on Service-Oriented Computing. pp. 397–411. Springer (2015)

[8] Eilam, T., Elder, M., Konstantinou, A.V., Snible, E.: Pattern-based composite application deployment. In: 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops. pp. 217–224. IEEE (2011)

[9] Fehling, C., Leymann, F., Retter, R., Schumm, D., Schupeck, W.: An architectural pattern language of cloud-based applications. In: Proceedings of the 18th Conference on Pattern Languages of Programs. p. 2. ACM (2011)

[10] Feinerer, I.: Efficient large-scale configuration via integer linear programming. Artificial Intelligence for Engineering Design, Analysis and Manufacturing 27(01), 37–49 (2013)

[11] Gambi, A., Toffetti, G., Pautasso, C., Pezze, M.: Kriging controllers for cloud applications. IEEE Internet Computing 17(4), 40–47 (2013)

[12] Giannakopoulos, I., Papailiou, N., Mantas, C., Konstantinou, I., Tsoumakos, D., Koziris, N.: Celar: automated application elasticity platform. In: Big Data (Big Data), 2014 IEEE International Conference on. pp. 23–25. IEEE (2014)

[13] Guillén, J., Miranda, J., Murillo, J.M., Canal, C.: A service-oriented framework for developing cross cloud migratable software. Journal of Systems and Software 86(9), 2294–2308 (2013)

[14] Hirmer, P., Breitenbücher, U., Binz, T., Leymann, F., et al.: Automatic topology completion of tosca-based cloud applications. In: GI-Jahrestagung. pp. 247–258 (2014)

[15] Homer, A., Sharp, J., Brader, L., Narumoto, M., Swanson, T.: Cloud design patterns: Prescriptive architecture guidance for cloud applications (2014)

[16] Huber, N., Brosig, F., Kounev, S.: Model-based self-adaptive resource allocation in virtualized environments. In: Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems. pp. 90–99. ACM (2011)

[17] Huber, N., Brosig, F., Spinner, S., Kounev, S., Bahr, M.: Model-based self-aware performance and resource management using the descartes modeling language. IEEE Transactions on Software Engineering (2016)

[18] Humble, J., Molesky, J.: Why enterprises must adopt devops to enable continuous delivery. Cutter IT Journal 24(8), 6 (2011)

[19] Keller, A., Hellerstein, J.L., Wolf, J.L., Wu, K.L., Krishnan, F.: The champs system: change management with planning and scheduling. In: Network Operations and Management Symposium, 2004. NOMS 2004. IEEE/IFIP. vol. 1, pp. 395–408. IEEE (2004)

[20] Ledeczi, A., Maroti, M., Bakay, A., Karsai, G., Garrett, J., Thomason, C., Nordstrom, G., Sprinkle, J., Volgyesi, P.: The generic modeling environment. In: Workshop on Intelligent Signal Processing, Budapest, Hungary. vol. 17, p. 1 (2001)

[21] Loope, J.: Managing Infrastructure with Puppet. " O'Reilly Media, Inc." (2011)

[22] Lu, H., Shtern, M., Simmons, B., Smit, M., Litoiu, M.: Pattern-based deployment service for next generation clouds. In: Services (SERVICES), 2013 IEEE Ninth World Congress on. pp. 464–471. IEEE (2013)

[23] Maróti, M., Kecskés, T., Kereskényi, R., Broll, B., Völgyesi, P., Jurácz, L., Levendovszky, T., Lédeczi, Á.: Next generation (meta) modeling: Web-and cloud-based collaborative tool infrastructure. MPM@ MoDELS 1237, 41–60 (2014)

[24] Morin, B., Barais, O., Jezequel, J.M., Fleurey, F., Solberg, A.: Models@ run. time to support dynamic adaptation. Computer 42(10) (2009)

[25] Narain, S., Levin, G., Malik, S., Kaul, V.: Declarative infrastructure configuration synthesis and debugging. Journal of Network and Systems Management 16(3), 235–258 (2008)

[26] Nelson-Smith, S.: Test-Driven Infrastructure with Chef: Bring Behavior-Driven Development to Infrastructure as Code. " O'Reilly Media, Inc." (2013)

[27] OASIS: Topology and orchestration specification for cloud applications. http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.pdf (2013), oASIS Standard

[28] Peiris, C., Sharma, D., Balachandran, B.: C2tp: a service model for cloud. International Journal of Cloud Computing 1(1), 3–22 (2011)

[29] Schmidt, D.C.: Model-driven engineering. COMPUTER-IEEE COMPUTER SOCIETY- 39(2), 25 (2006)

[30] Shekhar, S., Chhokra, A.D., Bhattacharjee, A., Aupy, G., Gokhale, A.: Indices: Exploiting edge resources for performance-aware cloud-hosted services. In: Fog and Edge Computing (ICFEC), 2017 IEEE 1st International Conference on. pp. 75–80. IEEE (2017)

[31] Tekiner, F., Keane, J.A.: Big data framework. In: Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on. pp. 1494–1499. IEEE (2013)

[32] Urgaonkar, B., Pacifici, G., Shenoy, P., Spreitzer, M., Tantawi, A.: An analytical model for multi-tier internet services and its applications. In: ACM SIGMETRICS Performance Evaluation Review. vol. 33, pp. 291–302. ACM (2005)

[33] Yassa, S., Chelouah, R., Kadima, H., Granado, B.: Multi-objective approach for energy-aware workflow scheduling in cloud computing environments. The Scientific World Journal 2013 (2013)

[34] Zheng, T.: Model-based Dynamic Resource Management Multi Tier Information Systems. Ph.D. thesis, Carleton University Ottawa (2007)