ARTIST: Model-Based Stairway to the Cloud

Javier Troya[1], Hugo Brunelière[2], Martin Fleck[1], Manuel Wimmer[1], Leire Orue-Echevarria[3], and Jesús Gorroñogoitia[4]

[1] Business Informatics Group, Vienna University of Technology, Austria
{troya,fleck,wimmer}@big.tuwien.ac.at
[2] AtlanMod Team (Inria, Mines Nantes & LINA), Ecole des Mines de Nantes, France
hugo.bruneliere@inria.fr
[3] Tecnalia Research and Innovation, Bilbao, Spain
leire.orue-echevarria@tecnalia.com
[4] ATOS Research and Innovation, Madrid, Spain
jesus.gorronogoitia@atos.net

Abstract. Over the past decade, cloud services emerged as one of the most promising technologies in IT. Since cloud computing allows improving the quality of software and, at the same time, aims at reducing costs of operating software and hardware, more and more software is delivered as a service in the cloud. However, moving existing software applications to the cloud and making them behave as software as a service is still a major challenge. In fact, in addition to technical aspects, business aspects also need to be considered. The ARTIST EU project (FP7) proposes a comprehensive model-based modernization approach, covering both business and technical aspects, to *cloudify* already existing software. In particular, ARTIST employs MDE techniques to automate the reverse engineering and forward engineering phases in a way that modernized software truly benefits from targeted cloud environments. In this paper we describe the overall ARTIST approach and present several lessons learned.

Keywords: ARTIST, Migration, MDE, Cloud

1 Introduction

Dealing with paradigm transitions has always been a recurring problem in software engineering. Nowadays, one of the most popular technological and business trends is to deploy applications on the cloud. This notably allows making pieces of software and related offered services available more easier and dynamically to a wider audience. This also provides some interesting new capabilities, such as improved scalability in contexts where traditional software running in on-premise environments was previously not efficient enough. While the most recent applications may have been designed with cloud deployment in mind, a large majority of the already existing software has been developed in such a way that it is not directly fully cloud-compatible. We refer to this as *legacy* software. As a consequence, there is currently a real need for concrete solutions supporting companies in evolving their legacy applications in order to make them deployable on the cloud, as well as to exploit the full potential and services provided by the cloud.

As an answer to this problem, the ARTIST EU collaborative project [4] aims at facilitating the migration and modernization of legacy software assets and businesses to the

cloud. To this intent, it provides a generic customizable model-based methodology and corresponding open source tooling for migrating such applications to the cloud. Covering the traditional reverse engineering and forward engineering phases (i.e., the actual migration), it also addresses (pre-)migration feasibility analysis from both technical and business perspectives as well as (post-)migration verification and certification.

The remainder of the paper is structured as follows. Section 2 gives an overview of the ARTIST project from an general point of view. Then, Section 3 introduces the main objectives of the project and its outcomes. Section 4 details the main innovation aspects brought by ARTIST, and also some encountered obstacles. Section 5 discusses the related work and collaboration with related EU projects. Finally, Section 6 concludes the paper by summarizing the main achievements and presents the ongoing exploitation of the results.

2 Project's Overview

ARTIST, standing for "Advanced software-based seRvice provisioning and migraTIon of legacy SofTware", is an EU Integrated Project (IP) which is part of the Seventh Framework Programme for Research and Technological Development (FP7). It is a still ongoing project that started on October 1, 2012 for a total duration of three years, thus ending in coming September 30, 2015. The project has a total budget of €9,690,258, for a total EC funding of €6,953,705. It directly involves 10 partners coming from 7 different countries which are Spain, France, Germany, Austria, Italy, Greece and Belgium. Academics in the project are coming from internationally recognized institutions: Inria, Fraunhofer, Tecnalia, Vienna University of Technology and Institute of Communication and Computer Systems. Industrial partners vary from innovative SMEs or tool vendors (Sparx Systems, ATC, Spikes) to large service companies (Atos, Engineering). The project is composed of a total of 13 work packages (WPs). WP1-WP4 are organizational WPs, while WP5-WP11 are technical WPs. Finally, WP12 is concerned with the use cases development and WP13 with the tools and methodology evaluation on the use cases. All the information on the ARTIST project is publicly available from the project website[5], including related material as well as access to the ARTIST Open Source Release.

Figure 1 gives an overview of the approach designed and developed in the context of ARTIST. The ARTIST model-based approach covers three main phases:

- **Pre-Migration**. It occurs prior to the actual realization of the migration. It principally consists in ensuring that the migration is feasible and/or desirable, from both a business and technical perspective.
- **Migration**. It can be triggered after the pre-migration phase in case of successful assessment. It is composed of two sub-phases:
  - **Reverse Engineering**. It deals with better understanding the initial application thanks to the (semi-)automated discovery of models describing it as accurately as possible and at different levels of abstraction (from low-level code models up to higher-level design models including, e.g., architectural models).

[5] http://www.artist-project.eu/

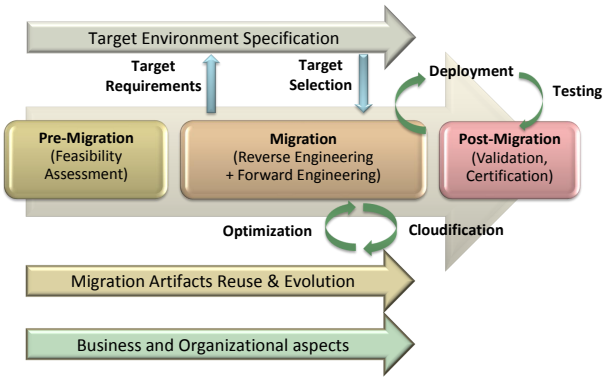

Fig. 1. Overview of the ARTIST approach

- **Forward Engineering**. It reuses the previously obtained models in order to perform the required cloud-oriented adaptations and optimizations onto (parts of) the initial application, with the final objective of sofware (re-)generation.

– **Post-Migration**. It starts after the actual migration has been fully performed in order to evaluate the resulting cloud-compatible pieces of software. It notably consists in validating that the migrated application behaves similarly to the original one once deployed to the cloud, and certifying the compliance of the migrated solution to common cloud practices.

In parallel to these activities, ARTIST also provides support for better handling the cloud target environment identification and selection process (if relevant, in some cases the target cloud platform could be imposed for various reasons). Complementarily, it comes with a dedicated repository for storing and retrieving useful (modeling) artifacts produced in past migration projects (e.g., common metamodels, generic model transformations or skeletons of extensible transformations, code generators, etc.).

3 Objectives and Expected Outcomes

ARTIST provides both an overall model-based methodology and corresponding open source tooling to apply it on real migration scenarios. These ARTIST outcomes are fully reflected within the official ARTIST Open Source Release[6], which gives free public access to the project results in a transparent manner.

Thus, relying on the model-based approach shortly summarized in Section 2, ARTIST provides a set of concrete outcomes that can be practically used and deployed in the context of migration-to-the-cloud projects. Figure 2 summarizes these main outcomes.For a more detailed description and related references (e.g. already published articles), see Section 4.

The first ARTIST asset is its generic and customizable methodology that comes under the form of 1) a complete handbook literally describing it and 2) a Methodology

[6] http://www.artist-project.eu/open-source-package

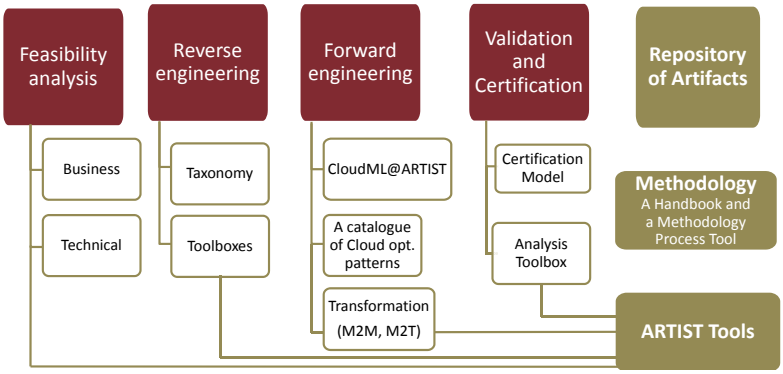

Fig. 2. The main ARTIST outcomes

Process Tool (MPT) allowing to plan, model and follow its underlying processes within the context of various migration projects. In order to practically implement this methodology, ARTIST offers the so-called ARTIST Tools as a second main asset. Interestingly, a large majority of its components is part of the ARTIST Open Source Release.

This ARTIST Tools asset is currently composed of individual components or tool sets covering four fundamental phases of the ARTIST approach. The pre-migration activity of analyzing a given migration feasibility prior to its actual realization can be made thanks to the combined use of the Maturity Assessment Tool (MAT), Business Feasibility Tool (BFT) and Technical Feasibility Tool (TFT). To support Model-Driven Reverse Engineering (MDRE), ARTIST proposes a taxonomy of legacy artifacts guiding engineers in the process of doing a very first analysis of the existing application and its internal structure/content. Then two toolboxes, the Model Discovery Toolbox (MDT) and Model Understanding Toolbox (MUT), allow performing initial model discovery and further model understanding activities, respectively. Having obtained all the required models of the original application at the appropriate levels of abstraction, forward engineering can be realized. In this regard, ARTIST provides different valuable artifacts. The first one is the CloudML@ARTIST language, defined as a UML profile, that is intended to support the identification and definition of the cloud target. The second one is a catalogue of common cloud-specific optimization patterns that can be reused (and eventually completed) in the context of various migration-to-the-cloud projects. The last one is a set of already existing model transformations and code generators that are either completely generic or to be customized to particular migration scenarios. Finally, the migrated software is analyzed to see if the behavior of the legacy software is preserved and if the non-functional requirements are compliant. A certification of the migration (e.g., as a consultancy service) can also be realized.

The ARTIST Repository is another significant outcome of the project. It provides a centralized way of dealing with all the reusable artifacts that can be relevant in such a migration context (cf. the reusable modeling artifacts above-mentioned for instance).

An empirical evaluation of the methodology and the tools is currently being conducted by the use case providers during this last year of the project [4].

4 Main Innovations and Encountered Obstacles

Even though all tasks and work packages are inter-related and their outcomes are used together in the overall ARTIST solution, each one investigates its own line of research. For this reason, the ARTIST project has advanced the state-of-the-art in different fields, unavoidably facing some obstacles and barriers. In this section we present the advances realized in the context of the tasks involving the phases presented in Section 2, and express the difficulties we have found. In particular, we cite some of our works from the more than 30 research papers already published at the time of writing.

4.1 Advances in Different Fields in the Main Three Phases

The *Modernization assessment* task covers the pre-migration phase mentioned in Section 2. In [1, 19, 20], we presented an innovative analysis combining technical and business dimensions in order to assess the maturity of an application and the convenience of migrating it to the cloud. It is based on quantitative indicators always ensuring the company's business continuity. In [2], we conducted a practical application of this pre-migration phase in a particular scenario.

Task *Legacy Product Analysis by Reverse Engineering* corresponds to the reverse engineering phase of the migration process (cf. Section 2). In [10], we presented the MoDisco open source MDRE framework which is used as our overall approach for both discovering initial models from the system artifacts and further understanding them. In the context of ARTIST, we especially advanced on the *model discovery* of behavioral UML2 Activity models from source code, whose implementation is still ongoing. In [5, 6] we presented JUMP, a framework which offers the possibility to discover UML profiles out of annotations at code level and which we integrated within the MoDisco approach. This framework advances the field by finding an effective mapping between Java and UML, generating UML profiles from annotation-based libraries. As for the *model understanding* phase, in [4] we presented an approach to obtain only the parts of a metamodel that we are interested in. This is realized by a type-safe restructuring of snippets that are generated from base metamodels. Also, in [12], we presented an automatic approach to obtain a component model from a class diagram using search-based optimization techniques.

The *New software generation by forward engineering* task represents the forward engineering phase. In order to optimize the application with regards to its non-functional properties, we presented in [12, 14] a search-based software engineering (SBSE) approach to select the proper set of optimization patterns, out of a catalogue of these patterns (cf. Section 3), to apply. In such works, we proposed to optimize the model of the application through in-place transformations. Thus we presented the MOMoT framework which provides several algorithms for local and global searches of rule applications guided by single and multiple objectives formulated in terms of models. We presented in [7] how CloudML@ARTIST facilitates expressing cloud-based deployments directly in UML, which is especially beneficial for migration scenarios where reverse-engineered UML models are tailored towards a selected cloud environment. Since the process of forward engineering is driven by model transformations, we proposed in [8]

the concept of *patch transformations*, as created in the context of co-evolution, where only the part that has changed needs to be re-executed in a model transformation.

Task *Migrated product testing, validation and certification* represents the post-migration phase, where the quality of the modernized software is to be evaluated. In [13], we studied how FUML can be used to study the non-functional properties of UML models, without needing to translate the latter into any other formalism. We presented in [18] an approach for integrating existing software libraries with FUML models, so that they can be considered in model simulation. Also aiming at comparing models for the legacy and cloud versions of the applications, we propose in [16] a generic semantic differencing approach that can be instantiated to realize semantic diff operators for specific modeling languages. Finally, we investigated on how to find bugs in model transformations [11], which play a central role in the forward engineering process. We came up with an approach to systematically mutate model transformations [21], which is an important step in the process of identifying bugs.

4.2 Encountered Obstacles

The heterogeneity of programming languages and frameworks present in already existing software, as well as the plethora of existing platforms for deploying applications to the cloud, have been the main obstacles that we have found in our project. The purpose of the ARTIST project is not to define a specific migration process for a particular technology, but rather to propose a semi-automatic generic software migration approach. However, even if the approach is generic, we still need to rely on some technical components that can be quite specific. Furthermore, coming up with such a generic approach is not easy when actual inputs can vary significantly from some applications to others.

For these reasons, we pragmatically dediced to focus on particular technologies for instantiating specific parts of our generic process. For instance, we have been able to obtain UML profiles from Java libraries. We would need to slightly modify the implementation in order to also consider C# code. As another example, for the generation of code from UML models, we have focused on Java and C# as key languages for our industrial partners. However, other target programming languages may also be considered in the future.

In any case, these limitations are precisely the reason why we decide to follow a model-based approach. Raising up the level of abstraction allows us to reason about the software application properties in a platform-independent manner, and consequently to reason about adaptations for the cloud without being polluted by too low-level technical aspects. These adaptations are therefore abstracted away from any technology and can be generically (re)applied in different scenarios.

5 Related Work and Projects

Since cloud computing is a relatively novel computing paradigm, several ongoing research and European projects are currently dealing with the many different issues regarding cloud systems modeling. For instance, MODAClouds [3] and PaaSage [15] also propose, among their objectives, some model-based migration support. Currently,

we are collaborating with MODAClouds and PaaSage to come up with a common modeling language for cloud software by merging the languages that have been created in the three projects.

The SeaClouds project [9] takes care of different aspects of the cloud development life-cycle, such as an open, generic and interoperable foundation to orchestrate parts of cloud-based applications. It provides services to monitor, manage and migrate the underlying providers (both public and private clouds) and thus leverages SLA policies in order to guarantee the required performance and QoS on multi-cloud environments. We have already collaborated with this European project in the study of the analysis of non-functional properties of systems [17].

6 Conclusions and Ongoing and Future Exploitation

The ongoing ARTIST project intends to provide relevant support for making easier the process of migrating already existing (legacy) applications to the cloud. It is currently resulting in a general model-based methodology and corresponding open source tooling allowing to implement it in the context of real industrial migration projects.

We are now in the process of preparing the project's follow-up in terms of further exploitation of relevant results, both for the ARTIST consortium as a whole and from an individual partner perspective. As already mentioned, ARTIST has a committed open source exploitation strategy where tools are built by contributors to the open source community. At the same time, the commercial partners in the project have high aspirations for the results. There is a strong potential for partnerships based on geographic coverage, skills and IPR synergies as well as links to the partner's existing portfolios. The partners have been thinking about a way to formalize collaboration in exploitation, such as pooling investment in development, marketing and cross fertilization of opportunities (respecting the open source distribution of software and their license terms).

Consequently, the consortium plans to form a so-called "ARTIST Club" based on a legal agreement which controls the use and ownership of branding associated with the project. In this way, the software can be used under the terms of the license while organizations cannot market services based on them under the ARTIST brand. Ultimately, then, the ARTIST Club serves as a marketing umbrella through which a greater presence can be reached for less resources than through diluted individual marketing investments. At the time of writing, the ARTIST Club contract is being drafted and project participants are finalizing exploitation plans based on opportunities through it.

Acknowledgement

This work is co-funded by the European Commission under the ICT Policy Support Programme, grant no. 317859 (ARTIST project).

References

1. Alonso, J., Orue-Echevarria, L., Corera, Z., Gorronogoitia, J., Karaboga, B.: ARTIST Technical Feasibility Tool: Supporting the early technical feasibility assessment of application cloudifications. In: Proc. of ICSEA (2014)

2. Alonso, J., Orue-Echevarria, L., Escalante, M., Gorronogoitia, J., Presenza, D.: Cloud modernization assessment framework: Analyzing the impact of a potential migration to Cloud. In: Proc. of MESOCA (2013)
3. Ardagna, D., di Nitto, E., Mohagheghi, P., Mosser, S., Ballagny, C., D'Andria, F., Casale, G., Matthews, P., Nechifor, C.S., Petcu, D., Gericke, A., Sheridan, C.: MODAClouds: A model-driven approach for the design and execution of applications on multiple Clouds. In: MISE@ICSE. IEEE/ACM (2012)
4. Bergmayr, A., Bruneliere, H., Canovas Izquierdo, J., Gorronogoitia, J., Kousiouris, G., Kyriazis, D., Langer, P., Menychtas, A., Orue-Echevarria, L., Pezuela, C., Wimmer, M.: Migrating Legacy Software to the Cloud with ARTIST. In: Proc. of CSMR (2013)
5. Bergmayr, A., Grossniklaus, M., Wimmer, M., Kappel, G.: Bridging java annotations and UML profiles with JUMP. In: Proc. of the Demonstrations Track MoDELS. CEUR Workshop Proceedings, vol. 1255 (2014)
6. Bergmayr, A., Grossniklaus, M., Wimmer, M., Kappel, G.: Jump—from java annotations to uml profiles. In: Proc. of MoDELS, LNCS, vol. 8767. Springer (2014)
7. Bergmayr, A., Troya, J., Neubauer, P., Wimmer, M., Kappel, G.: UML-based Cloud Application Modeling with Libraries, Profiles, and Templates. In: Proc. of CloudMDE@MoDELS. CEUR Workshop Proceedings, vol. 1242 (2014)
8. Bergmayr, A., Troya, J., Wimmer, M.: From out-place transformation evolution to in-place model patching. In: ASE. ACM (2014)
9. Brogi, A., Ibrahim, A., Soldani, J., Carrasco, J., Cubo, J., Pimentel, E., D'Andria, F.: SeaClouds: A European Project on Seamless Management of Multi-cloud Applications. SIGSOFT Softw. Eng. Notes 39(1), 1–4 (2014)
10. Brunelière, H., Cabot, J., Dup´e, G., Madiot, F.: Modisco: A model driven reverse engineering framework. Information and Software Technology 56(8), 1012 – 1032 (2014)
11. Burgueno, L., Troya, J., Wimmer, M., Vallecillo, A.: Static Fault Localization in Model Transformations. IEEE Transactions on Software Engineering 41(5), 490–506 (May 2015)
12. Fleck, M., Troya, J., Wimmer, M.: Marrying Search-based Optimization and Model Transformation Technology. In: Proc. of NasBASE (2015)
13. Fleck, M., Berardinelli, L., Langer, P., Mayerhofer, T., Cortellessa, V.: Resource Contention Analysis of Cloud-based System through fUML-driven Model Execution. In: Proc. of NIMALP@MoDELS. CEUR Workshop Proceedings, vol. 1074 (2013)
14. Fleck, M., Troya, J., Langer, P., Wimmer, M.: Towards Pattern-Based Optimization of Cloud Applications. In: Proc. of CloudMDE@MoDELS. CEUR Workshop Proceedings, vol. 1242 (2014)
15. Koller, B.: Model Based Cloud Application Development using PaaSage. Innovatives Supercomputing in Deutschland 11(1) (2013)
16. Langer, P., Mayerhofer, T., Kappel, G.: Semantic Model Differencing Utilizing Behavioral Semantics Specifications. In: Proc. of MoDELS, LNCS, vol. 8767. Springer (2014)
17. Moreno-Delgado, A., Durán, F., Zschaler, S., Troya, J.: Modular DSLs for Flexible Analysis: An e-Motions Reimplementation of Palladio. In: Proc. of ECMFA. LNCS, vol. 8569, pp. 132–147. Springer (2014)
18. Neubauer, P., Mayerhofer, T., Kappel, G.: Towards Integrating Modeling and Programming Languages: The Case of UML and Java. In: Proc. of GEMOC@MoDELS. CEUR Workshop Proceedings, vol. 1236, pp. 23–32 (2014)
19. Orue-Echevarria, L., Alonso, J., Escalante, M., Schuster, S.: Assessing the Readiness to Move into the Cloud. In: Cloud Computing. Springer (2013)
20. Orue-Echevarria, L., Escalante, M., Alonso, J.: An Assessment Tool to Prepare the Leap to the Cloud. In: Cloud Computing. Springer (2013)
21. Troya, J., Bergmayr, A., Burgueño, L., Wimmer, M.: Towards Systematic Mutations for and with ATL Model Transformations. In: Mutation Workshop @ ICST (2015)