A Structured and Achromatic Process for Legacy to Cloud Migration: A Survey

Ahmad Jalili* and Omid Bushehrian

PhD Student, Department of Computer Engineering and Information Technology, Shiraz University of Technology, Shiraz, Iran.

Assistant Professor, Department of Computer Engineering and Information Technology, Shiraz University of Technology, Shiraz, Iran.

*Corresponding Author's E-mail: A.jalili@sutech.ac.ir

Abstract

Cloud computing is a computing model that provides computing resources on demand and charges only for the required and consumed resources (i.e., pay-as-you-go pricing model). Moreover, this computing paradigm leverages from most benefits such as high scalability, reduced IT costs, self-service on demand, and pay-as-you-go price models, that increasingly attracted the interest of the corporate world. Despite these attractions, in practice many organizations have found it difficult to adopt cloud-based solutions, particularly when faced with the need to migrate existing legacy applications to this new environment. One of the main obstacles faced by those organizations is the lack of a structured and general process to help application developers in selecting methods, tools and solutions for migration. For this purpose, this paper presents a structured and achromatic process for legacy migration to cloud environment. The process relies on a clear and simple step-by-step approach for migration operation and also introduces tools and techniques which can be applied in every step. Moreover, in this paper business model and scalability are considered.

Keywords: cloud migration; legacy-to-cloud migration; cloud computing; step by step migration process.

1. Introduction

Cloud computing is an emerging computing paradigm whose benefits (such as high scalability, reduced IT costs, self-service on demand, and pay-as-you-go price models) have increasingly attracted the interest of the corporate world. Those benefits have increasingly attracted the interest of all kinds of organizations, from industry to governmental and non-governmental institutions, which see the cloud as a great opportunity to reduce IT costs, streamline their business processes, and improve the overall quality of their products and services. In addition, a continuously increasing set
of cloud-based solutions is available to application owners and developers to tailor their applications exploiting the advanced features of this paradigm for elasticity, high availability and performance. Nevertheless, it is difficult for organizations and developers to apply these techniques and solutions without a coherent plan and step by step process. Moreover, due to the variety of tools, techniques and solutions; most organizations are faced with the problem in selecting an appropriate migration strategy for cloud migration. In other hand one of the main restraints faced by those organizations is the lack of a structured and general process to help application developers in selecting available tools and solutions.

To fill this gap, this paper presents a structured and achromatic process for legacy migration to cloud environment. The process relies on a clear and simple step-by-step approach for migration operations and also introduces tools and techniques which can be applied in every step. This process includes six primary phases and two additional phases where initially we accomplish a preliminary feasibility analysis of the viability migration operation. If valuable of legacy system to cloud is realized, project team must extracts capabilities of existing system and candidate target environment by proposed tools and techniques in both phase two and three. After collecting a set of essential information on current state and future state, the project team will need to process these results. Therefore in transformation phase, various model-to-model transformations are performed for providing abstract levels of system. Then in Migrate activity, using the specific SOA/cloud computing patterns and methods, the new architecture of the migrated system will be built. After these mention phases, obtained products should be assessed in terms of various parameters in rest of process. In this paper, first we briefly describe best frameworks, techniques and tools that presented in academic papers for reporting legacy to cloud-base migration in the next section. As well, limitations of each solution has been partially evaluated. Then in section 3, we describe and define our process with its phases and tools. Common tools for migration are compared in terms of their application in section 4, and in the following, section 5 assesses introduced approaches and finally, Section 6 concludes our research with an outlook to future work.

2. Relate Work

There have been many research efforts on migration to cloud in recent years. In this section, we will review related techniques, with a focus on selecting of newest papers. Hesselbring in [1] has introduced a model based migration approach called CloudMIG. This approach
represents a semi-automatic method for migration and aims to address shortcomings of today’s migration projects such as applicability, level of automation, resource efficiency and scalability. CloudMIG is composed of six activities in order to migrating an enterprise system to a cloud environment: initially by using the OMG’s ADM (Architecture-Driven Modernization) standards such as KDM (Knowledge Discovery Metamodel) and SMM (Software Metrics Metamodel) the architectural and utilization models of the legacy system is extracted. Then a target architecture and a mapping model is produced and analyzed statically (by using metrics such as LCOM or WMC) and dynamically (by Simulating the target architecture). Finally manual migration toward the target architecture is performed [2]. Nevertheless, there are open issues and problems which are not addressed in CloudMIG. It supports only the early migration planning phase. Moreover it is not intended for migrations to SaaS solutions, e.g., for replacing an on-premise email system with a web-based variant.

In [3], a novel approach for migration on cloud is proposed. This approach is a model-driven migration methodology which provides an innovative toolbox that supports developers in each migration task. ARTIST (Advanced Software-based Service Provisioning and Migration of Legacy Software) consists of three major phases: pre-migration, migration and post-migration. Pre-migration consists of legacy software analysis and goal setting. In the migration step, both the reverse engineering and forward engineering techniques are applied to analyze and model the non-cloud software, transform it into a higher level platform-independent model and further to build a cloud-compatible model. Post-migration step concludes the process by verifying the functionality of the migrated software against the legacy version through tests, and by collecting and analyzing measures in order to make sure that the goals set in the first step are met.

SMART (Service-oriented Migration and Reuse Technique) is defined as a process to help organizations to make initial decisions about the feasibility of reusing legacy components as services within a SOA or a cloud environment. The method involves five major composite activities: (1) establishing the stakeholder context, (2) describing the existing capabilities, (3) describing the target SOA state where a wide range of information about legacy components, the target SOA and the candidate services are gathered, Information-gathering activities for the three activities are directed by the Service Migration Interview Guide (SMIG). The SMIG is an instrument that is implemented as a data model and contains more than 60 categories of
questions that gather information about the legacy components, the candidate services, and the target SOA environment. (4) analyzing the gap between the existing capabilities and the target state, as well, SMIG contains questions that directly address the gap between the existing and target architecture, design, and code, and maps questions to answers to risks to mitigation strategies (5) developing a migration strategy and Finally, creating the Migration Alternatives Table and Service Migration Strategy which estimates the migration costs and risks and a list of migration issues. All these phases can — and should — be supported by reverse engineering techniques and tools. However, one of shortcomings of this method is most of these analysis are performed manually and based on the knowledge of the participating team [4,5]. One of the projects are supported by European Commission is REMICS that run from 2010 to 2013. The goal of REMICS is to providing a model-driven methodology and tools for legacy-to-cloud migrations in a service-oriented way. The baseline concept of REMICS is the Architecture Driven Modernization (ADM) and Knowledge Discovery Meta-model (KDM) [6]. In this framework, modernization starts with the extraction of the architecture of the legacy application (the “Recovery” activity) where as an initially phase, ADM used to recovering the legacy system artifacts, and then KDM used to analyze these artifacts and provide the information basis for migration source architecture.

In REMICS approach the source architecture is migrated into cloud-capable target architecture by applying SOA and cloud computing patterns such as architecture decomposition, legacy components wrapping and legacy components replacement with new discovered cloud services. Migration phase is supported by two additional concepts: Model-Driven Interoperability (MDI) which ensures that existing services can still interoperate with the services in the target architecture, and Validate, Control and Supervise activity which e.g. validates the target architecture against Quality of Service (QoS) requirements and test cases. Finally, REMICS project offers a meta-model and Unified Modeling Language (UML) profile called PIM4Cloud which can be used to describe the cloud deployment of target architecture from a designer perspective [7]. Recovery process using BLUAGE tool and the use of SoaML and forward engineering with Modelio tool. However, REMICS methodology does not take into consideration non-functional requirements (i.e. performance) and neither addresses architectural issues such as multi-tenancy or scalability [8].
3. THE STRUCTURED PROCESS

According to previous studies, it is concluded that a comprehensive legacy to cloud migration solution comprises the following major steps.

![Structured legacy to cloud migration process](image)

**Figure 1:** structured legacy to cloud migration process

3.1. Feasibility

One of the most important issues in the migration legacy system to cloud feasibility analysis. Despite importance of the activity, current methodologies have done it tardily or together other activities such as extraction and migration activities and thus waste of time and cost. On the other hand, ignoring such issues mean repeating errors which can imply extremely high expenditures, projects that never end and even worse, some (very expensive) failures. Therefore, we initially define an explicit decision point to determine if the legacy system is a good candidate for migration to cloud or not. If the legacy system is not a good candidate, stopping at this point will save time and money.

Problems that in this step has propounded include:

- solicit information about stakeholders (typically include the owners and current end users of the legacy system, and the potential end users of the migrated services),
- Characteristics of the target environment,
- Characteristics of the legacy system (language programing, interfaces, codes, platform, ...),
- Data concerning legacy components,
- Potential services and Risks and issues specific to the legacy system.
Based on the information collected about existing state and the future state, the method provides a preliminary feasibility analysis of the viability. It also provides preliminary estimates about the cost and the risks involved. Gathered information to support the analysis activity extracted from various resources and techniques where we have generally categorized the techniques into two:

- Reverse engineering and
- Knowledge-based.

By knowledge-based techniques, we mean the knowledge and experience of initial developers and/or maintainer, the end-user experiences obtained via interviewing, and using existing documentation to understand the legacy system and target cloud, for this purpose can be used from SMIG instrument where proposed by SMART [11]. Nevertheless, if resources of information were insufficient can be used from reverse engineering like source code analysis. When valuable of legacy system to cloud is realized, project team must extracts capabilities of existing system and candidate target environment to better recognition of artifacts. However a study of the technical and economic feasibility will be conducted as a prerequisite to the migration/modernization of the legacy system.

3.2. Describe Existing Capability

Legacy applications are subjected to evolutionary development and bug fixing. These activities leads to so-called “spaghetti code”. Furthermore, lack of up-to-date documentation and resources make the understanding of the code inevitably hard. In such a scenario, identifying existing capabilities and candidate services is a challenging task. Nevertheless, identifying existing capabilities is an important activity in the context of legacy to cloud migration as this activity enables reusability and leveraging the existing legacy features. This activity, that sometimes in some literatures well known Model Discovery, is a technical and deductive process of acquiring knowledge about the “as-is” situation of legacy applications where during this phase a minimum set of required initial models out of (some of) the legacy artefacts composing the system is generated. This activity notably implies analyzing the different available legacy artefacts (source code, binaries, documentation, users’ knowledge, configuration files, execution logs and traces) to identify the ones which are actually relevant to the considered migration scenario as well support to the discovery of models describing the
legacy system in order to have a better understanding of it. In this activity, the legacy system will be analyzed from both technical perspectives and business perspectives. Indeed, the first step in this phase is to analyze how mature the application is in terms of technology (i.e. architecture, programming language, identifying dependencies, database, integration with third party offerings, etc.) and business (i.e. current business model, maintenance and upgrades procedures, etc.).

Accordingly, this activity aims to firstly complete identification of legacy component and recognition of component interdependency, secondly identifying existing capabilities and candidate services, thirdly identifying cloud-enabled component. There are different techniques in order to generate describing and architectural models as well describe candidate service where we these techniques categorize into two approach:

- top-down approach and
- bottom-up approach.

In the top-down approach, initially a business process is modeled based on the requirements and then the process is subdivided into sub-processes until these can be mapped to legacy functions. The bottom-up approach utilizes the legacy code to identify services using various techniques such as information retrieval, concept analysis, business rule recovery, source code visualization [12].

In addition, this activity will be supported by automated Model Driven Reverse Engineering (MDRES) and knowledge discovery methods as well there are some tools to extract information and produce models such as BLUAGE tool [9], Modelio tool [10], Business Feasibility Tool and the Technical Feasibility Tool [13]. The information extracted from the phase should support the decision about the better strategy for legacy system migration.

3.3. Describe Target Environment

The third activity is intended to gather sufficient detail about the target cloud environment to support decisions about what services may be appropriate and how they will interact with each other and the cloud environment. This activity can be performed either in parallel with secondary or already be provided by the service provider. In general, the activity represents two aspects of the target architecture: (i) the functional aspect, and (ii) the technical aspect. From the functional aspect, the target architecture not only represents the “to-be” functionalities, but also focuses on various non-functional characteristics such as performance,
security, availability. From the technical aspect, decisions are made regarding the selection of the technology to be used (SOAP or REST), messaging and communication protocols, service description languages, and service registry. In addition in order to comply with the business process information their context and the cost of services they use are also considered.

There are a variety of techniques to extract such information namely SMART for extract information via SMIG-directed conversations with owners [11], CloudMIG uses cloud environment meta-model where this meta-model comprises entities like VM instances or worker threads for IaaS and PaaS-based cloud environments [14].

On other hand, other approaches have attempted to produce platform-independent models in extraction activity. For example, REMICS is the ability to push the abstracted system to any open technology with a focus on cloud computing and for this purpose it uses two Platform Independent Model (PIM) and Platform Specific Model (PSM) precepts [15].

However, Even if it is completely platform-independent models to be extracted architectural target environment imposes a series of constraints (architecture, Communication technology and also business rules) on the project. Therefore, we need a better understanding of the system and also extraction of transformation rules in order to accommodate future.

3.4. mapping & transformation

After collecting a set of essential information on current state and future state like architectural models and transformations rules, processing must be performed on them. With producing the necessary “processed” derived models, thus filtering only the information required for the remainder of the overall process and also these derived models conform to different meta-models and so represent “views” on the legacy system at different levels of abstraction, according to the requirements of the next tasks.

This activity relies on the use of various (chains of) model-to-model transformations from the previously obtained initial models to the final derived models to be provided as inputs of the Modernization tasks. As well during this phase, architectural constraints and transformation rules that already have been extracted and described must be considered. For this purpose uses a model-enabled matchmaking algorithm [16].
The main objective of these model-to-model transformations and matchmaking algorithm is to provide a variety of abstract views of the legacy system and migrated application and also assign and adapt existing architectural features and prospective models. One of the most important outcomes from this phase is the analysis of the feasibility of the modernization of the different parts of the components of the legacy applications and another outcome will be the migration strategy to be applied to the different components of the legacy system. The main aim for these transformations is to generate diverse models providing different views over the migrated application, including its source code.

3.5. Migrate

Once the legacy system has been well understood, decomposed into features and/or components and provided final derived models by model-to-model transformations, other tasks such as Model Driven Forward Engineering (MDFE) need to be applied aiming at transforming the legacy system and deploying the migrated one into the target Cloud. Indeed, these models will be the starting point for the Migrate activity. During this activity, the new architecture of the migrated system will be built by applying specific SOA/cloud computing patterns and methods like architecture decomposition, legacy components wrapping and legacy components replacement with new discovered cloud services.

Generally, there are many migration strategies described in the state of the art, for example the “The Legacy Application Modernization Benchmark Report” identifies four possible migration strategies [17]: modernize, extend, surround and replace. Modernize means to update the legacy to new technologies (new database, new programming language, etc). The implication is that the legacy stays in the same place and is changed to be updated. Extends increases the legacy to include new features (such as web service communication), while surround creates a parallel system in charge of the new features. Finally replace means to discard the legacy and build a new application that duplicates its functionality.

This activity aims at building and deploying the migrated component corresponding to selected legacy components (e.g. implementations of application features). It should be noted that based on the particular legacy application requirements and the before activities analysis results, the migration processes and their flow are explicitly customized. While, for this purpose with Relying on information gathered from previous activities, PIM models and
architectural descriptive models provide the most appropriate migration strategy for legacy component. There are different tools where support migration activity such as: Target Specification, Optimization and Deployment tools which have been offered by ARTIST and supports the entire migration phase [17].

3.6. Adaption & Validation

Delivering a product is not the same as delivering a service in terms of organizational processes. Existing processes need to be analyzed and redefined to adapt the organization to the new structure. Also, new processes related to the service delivery will have to be defined from scratch. As the application has been migrated, In order to truly take advantage of the benefits of Cloud and is also compatible with the constraints of the target environment several issues need to be considered:

- Compatibility between services and issues relating to the interoperability,
- Issues to multi-tenancy,
- SaaS application must Scale,
- Bill automatically,
- Keep the highest security standards.

Needless to say that one of the strengths of service-oriented architectures is when migrating to the service cloud one has the possibility to add new functionality to the original system by combining the migrated services in a way where they not combined in the original system. One challenge with service composition is the need for Compatibility between services and issues relating to the interoperability. This problem can be solved via creating mediation services by combining a SoaML model with the data format model and a behavioral model for the mediation [7]. Issues relevant to Scalability, be multitenant, bill automatically and keep the highest security standards have to be accompanied with a change in the way the company offering the service works: new roles, new service model, a new form of payment, etc., in order to guarantee business continuity.

Once a feature has been migrated, the feature behavioral equivalence (across the legacy and migrated components) and the fulfilment of migration goals (i.e. non-functional requirements) are asserted. If the assessment fails, the process rolls back to the migration activity, in order to reevaluate the migration requirements and the optimization/transformation strategies.
Otherwise, the process moves to the next phase with concerns to the current feature. This activity iterates until all features have been migrated. Considering the above mentioned issues as well as after the use of existing solutions, produced artifacts (migrated components, features, business model) must be validated against the original requirements. The validation process comprises two straight steps:

1) The specification of the system which was migrated like the original specification of the system,

2) The migrated system itself has to be validated.

Since the project uses a model-based approach, several different model-based validation techniques can be used here. In the “Migrate” activity, the Business Process and Rules of the Source Architecture have to be fully implemented in Target Architecture. Finally, the system rebuilt for a service cloud platform has to fully address the business goals, process, rules, and QoS. Model checking will be applied for the verification of source and target models for various properties: coverage, conformance to design conventions and etc. In addition to the validation of the migrated system a model-based testing approach will be applied to derive the test scenarios and execute test cases.

In addition to above mention techniques there are a set of tools to support validation activity such as: Metrino, RSTestingTool, Fokus!MBT

The migration process consists of understanding the legacy system in terms of its architecture and functions, designing a new SOA application that provides the same or better functionality, and verifying and implementing the new application in the cloud. REMICS proposes a methodology that covers these steps based on model-driven engineering tools and methods.

3.7. Supervise & Certification

The purpose of this activity is to provide elements to control the performance of the system and to modify that performance. The step allows a company to monitor at all times, the performance of the application once this has been released and provisioned as a service, so it can be improved in performance, reliability, resources used and beware of possible degradation. Therefore, the focus is to implement the supervise procedures and components. This may involve the introduction of additional requirements, components and validation
procedures. On other hand virtualization increases the need for supervision (application behavior and QoS observations) and control (corrective actions) because changes and adaptations at runtime occur in a more sizable way. For this purpose we need a set of tools to instrument and enable the management of such services at runtime. There are tools for this purpose such as Models@Runtime and CloudPerformanceMonitoring tool. Models@Runtime is a set of libraries that allow to interact with models at runtime and also an appropriate approach for management, even self-management [3,7].

In addition to mention issues one of the major problems that new service-based software providers have to face is the reluctance of customers to consume new software offered as a service. Providers need to demonstrate their consumers that the service they deliver is of good quality, secure and trustable. Service consumers need to be sure that the SaaS applications they use and consume reach the minimum level of quality that is expected. To solve this, ARTIST [3] proposes a Certification Model that analyses: The organization (processes, products, financial aspects, and service continuity); the service offered (security, administration, support, QoS, SLA, service operational maturity); the application (functionality, usability, maintenance).

3.8. Evolution

Evolution activity is a crosscutting activity where the first propounded by ARTIST [16]. The main purpose of this phase is to provide a central place to store, archive and organize MDE artefacts such as models, meta-models and transformations and other information produced. This activity fosters reuse of artefacts and eases the evolution of software to a different cloud provider if needed. Since in this activity all intermediate results and artefacts attained in the whole process will be stored in a repository. Therefore, we offer this activity starts with three last activity of process and together with transformation activity in order to optimal operation. These artefacts will be made available to the public via a marketplace allowing to publish and consume artefacts as well as to search and browse the available artefacts and will allow users to comment on and rate them.

4. Comparison of tools

As mentioned previously, each one of the expressed approaches and methods employed a set of tools for migration operation. In this section, we briefly explain several common tools
where can be used for legacy to cloud migration. Then tools are compared in term of their application in each step. Blu age Software a solution for both reverse and forward engineering fully based on MDA and ADM principles and focuses on agile model transformation [9]. Blu Age software suite comprises of four modules: Forward Engineering, Reverse Modeling, Database Modernization and Continuous Integration Chain (Deliver). This software is exploited either in recovery phase and migrate phase in REMICS project which in the recover phase will take code as input (Cobol, PL/SQL, etc.) and will provide UML models as outputs, then these UML models are extracted for legacy code base by the Reverse Modeling module leveraging the functional specifications of the legacy code base for regeneration into a new programming language framework such as Java EE and .Net Enterprise business applications. Generally The Blu Age software suite is an agile and automated solution for application modernization that allows full rewriting of legacy applications with modern languages (JEE, .NET) for Private/Public/hybrid Cloud deployment. Using this software the project’s budget and time to go live can be reduced by up to 50%.

Modelio is a comprehensive modeling tool and environment with migration and recovery capabilities distributed under an open source license. Modelio is able to support many activities during de migration phase such as the component identification, the service architecture modelling, the cloud deployment modelling and code generation. In the recovery side it is able to support the recovery of java code. Modelio supports a wide range of models, diagrams and standards such as UML2, BPMN2, SoaML, XML and providing model assistance and consistency checking features. Generally Using Modelio you will get a highly professional modeling environment, having a wide modeling coverage and strong extensions capabilities [10].

Another tools those can be used in migration operation and Model Discovery is MoDisco. It is an Eclipse GMT project for model-driven reverse engineering. The objective is to allow practical extractions of models from legacy systems [18]. MoDisco proposes a generic and extensible meta-model-driven approach to model discovery. MoDisco approach has more benefits compared to already existing reverse engineering tools. Because MoDisco proposes a unified approach to model-driven reverse engineering and a metamodel driven methodology. This way, we are able to work in the modeling world, coming from a heterogeneous world to a homogeneous one. A general MoDisco process usually relies on two main phases: Model
Discovery that allows extracting low-level models out of the various artifacts composing a legacy system; Model Understanding that allows obtaining higher-level models out of the previously produced low-level models of the legacy system.

Table 1 depicts comparison of tools in term of their usage in each phase.

5. Evaluation of presented approaches

We reviewed some techniques for migration and in this section we investigate general procedures of each one from the perspective of supporting our proposed process. It worth to mention that each migration technique employs distinct methods. In other words, some techniques may not implement all phases or they might just focus on one phases (or several phases) more than other phases. Some of them may implement phases using their developed tools or run some phases manually. In table 2, we show which steps are covered by each technique and no details are given in this section.

Table 1: Comparison of tools

<table>
<thead>
<tr>
<th>Tools</th>
<th>Feasibility</th>
<th>Describe Existing Capability</th>
<th>Describe Target Environment</th>
<th>mapping &amp; transformation</th>
<th>Migrate</th>
<th>Adaption &amp; Validation</th>
<th>Supervise &amp; Certification</th>
<th>Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMIG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MoDisco (Model Driven reverse engineering techniques, Source code analysis)</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>knowledge-based techniques (interviewing, Existing Documents)</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TALE</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>BLUAGE</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Modelio</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Business Feasibility Tool</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Technical Feasibility Tool</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Target Environment</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Specification Tooling</td>
<td></td>
<td></td>
<td>x</td>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
6. Evaluation of presented approaches

We reviewed some techniques for migration and in this section we investigate general procedures of each one from the perspective of supporting our proposed process. It worth to mention that each migration technique employs distinct methods. In other words, some techniques may not implement all phases or they might just focus on one phases (or several phases) more than other phases. Some of them may implement phases using their developed tools or run some phases manually. In following table, we show which steps are covered by each technique and no details are given in this section.

<table>
<thead>
<tr>
<th>approaches</th>
<th>Feasibility</th>
<th>Describe Existing Capability</th>
<th>Describe Target Environment</th>
<th>mapping &amp; transformation</th>
<th>Migrate</th>
<th>Adaption &amp; Validation</th>
<th>Supervise &amp; Certification</th>
<th>Evolution</th>
</tr>
</thead>
<tbody>
<tr>
<td>CloudMIG</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ARTIST</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SMART</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>REMICS</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cloud Adoption Toolkit [19]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>CDOSim [20]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>
CONCLUSION

In this paper we present a structured and achromatic process for legacy migration to cloud environment. The structured process relies on a clear and simple step-by-step approach for migration operation that includes eight steps. For each phase, we introduced a variety of tools that can be used for specific operation. General procedures of each technique and common tools are investigated from the perspective of supporting our proposed process. The objective of this study was to consolidate existing research on legacy software migration to the cloud regarding the claimed benefits and the provided evidence of migration. As to future work, we advise to evaluate the structured process in industrial case studies.

References


