



Cloud migration process—A survey, evaluation framework, and open challenges



Mahdi Fahmideh Gholami^{a,*}, Farhad Daneshgar^{a,b}, Graham Low^a, Ghassan Beydoun^c

^aInformation Systems, Technology and Management Australian School of Business, University of New South Wales, West Wing Room 2106, Quadrangle Building, Sydney 2052, Australia

^bBangkok University, Bangkok, Thailand

^cSchool of Management, Leadership and Information Systems, University of Technology Sydney, Australia

ARTICLE INFO

Article history:

Received 15 December 2015

Revised 17 April 2016

Accepted 27 June 2016

Available online 28 June 2016

Keywords:

Cloud migration

Legacy application

Evaluation framework

Migration methodology

Process model

Cloud computing

ABSTRACT

Moving mission-oriented enterprise software applications to cloud environments is a crucial IT task and requires a systematic approach. The foci of this paper is to provide a detailed review of extant cloud migration approaches from the perspective of the process model. To this aim, an evaluation framework is proposed and used to appraise and compare existing approaches for highlighting their features, similarities, and key differences. The survey distills the status quo and makes a rich inventory of important activities, recommendations, techniques, and concerns that are common in a typical cloud migration process in one place. This enables both academia and practitioners in the cloud computing community to get an overarching view of the process of the legacy application migration to the cloud. Furthermore, the survey identifies a number challenges that have not been yet addressed by existing approaches, developing opportunities for further research endeavours.

© 2016 Elsevier Inc. All rights reserved.

1. Introduction

Many enterprise software applications that support IT functions are characterized by the need for high computing capability, scalability, and resource consumption (Buyya et al., 2009; Armbrust et al., 2010). In recent years, cloud computing initiatives have received significant attention towards addressing these requirements through offering services in various forms such as SaaS (software as a service), PaaS (platform as a service), and IaaS (infrastructure as a service) which are universally accessible, acquirable, and releasable on the fly, and payable on the basis of service usage amount. Given these advantages, many IT-based organisations have been interested in moving their legacy assets to cloud environments. It is estimated that the global cloud computing market will grow from \$40.7 billion in 2011 to \$241 billion in 2020 (Ried and Kisker, 2011). So far, a significant collection of research has been devoted to this topic by both academia and practitioners ranging from pure technical-centric solutions related to the using of cloud services, to research around the social and non-technical impact of the cloud as a new emerging paradigm. However, studies that focus on designing approaches offering a process model (method-

ology) for the cloud migration have not yet received much attention. Several studies such as Mohagheghi et al. (2010), Chauhan and Babar (2012) and Jamshidi et al. (2013) suggest that a well-defined process model for supporting the migration (or development) and maintaining working legacy applications to the cloud is a key concern. A rigorous approach is particularly important when moving large scale and complex legacies which have been in operation and stored critical data over the years. Moving legacies to the cloud raises many concerns such as security, interoperability, and vendor lock-in.

Legacy applications often predate the cloud computing and thus have been developed without taking into account the characteristics of cloud environments. The complexity of migration is exacerbated by the fact that some legacy applications may have been developed without taking into account the unique requirements attributed to cloud environments such as elasticity, multi-tenancy, interoperability, and cloud service/platform selection. Such requirements raise new challenges that entail to improve conventional software (re-engineering) development methodologies or to choose ones that address these specific requirements towards making a legacy application cloud-enabled. Various projects and studies in the cloud computing community have been suggested in order to enable legacy applications to utilize cloud services. A well-structured methodology can aid developers to carry out an effective and safe application migration, instead of struggling to understand “what” and “how” to carry out such a transition in an ad-hoc

* Corresponding author.

E-mail addresses: m.fahmidehgholami@unsw.edu.au (M.F. Gholami), f.daneshgar@unsw.edu.au (F. Daneshgar), g.low@unsw.edu.au (G. Low), beydoun@uow.edu.au (G. Beydoun).

manner which may latter result in poor and erroneous migration and maintenance overhead. A methodological approach can be claimed as promising mean for tackling the cloud migration complexities and move from an ad-hoc cloud migration to a structured and step by step quality methodology. In this spirit, [Laszewski and Nauduri \(2011\)](#), who are the designer of a methodology for moving Oracle legacy applications to the cloud, mention: *Like any software development project, migration projects require careful planning and good methodology to ensure successful execution.* A similar recommendation is stated in the final report of REMICS project, which is a three years research project supported by the European Commission and focuses on a methodological support for moving legacy applications to cloud platforms ([Benguria et al., 2013](#)). The above report mentions that *in the beginning it [legacy migration] was motivated by the lack of documentation, but in the last years it has been motivated by adaptation to new technologies. Each new technology has required new and renewed approaches and technologies to address the migration process in a more effective way.*

As will be elaborated in [Section 2.3](#), there are several surveys in the literature each focuses on different aspects of cloud migration such as interoperability, techniques and tools for migration, and cloud architecture design. Although these surveys provide a partial understanding of certain aspects of the legacy to cloud migration, they do not provide a complete picture of how the cloud migration is to be carried out and organised from the perspective of the process model. There is not yet a rigorous analysis of the extant material on this aspect of the cloud computing. For this reason and regarding the fact that the interest for legacy application migration to the cloud grows, there is a need to contribute a survey that distills existing cloud migration approaches by identifying their common characteristics and varying motives, concomitant activities, and empirical findings. This survey will differ from existing related surveys ([Section 2.3](#)) by focusing on the process aspect of the cloud migration to understand what essential activities and concerns are involved during such a transition. By comprehensively reviewing existing cloud migration approaches, we thus position this survey as the newest reference point for the cloud computing research and practice. Accordingly, the current study will attempt to answer the following research questions:

- **RQ1.** What are the existing approaches proposing a migration model for moving legacy applications to cloud environments in the literature?
- **RQ2.** What is the current state of these approaches w.r.t. the proposed evaluation framework introduced in [Section 3](#)?
 - RQ2.1.** what generic criteria, as typically expected for a software development methodology, are supported by these approaches?
 - RQ2.2.** what cloud-specific criteria are supported by these approaches?

RQ1 is motivated by the need to describe the state of the art of cloud migration approaches. This gives readers an overall understanding of approaches' core idea, their objectives, and a concise description of them. RQ2 was formulated to characterise as well as highlight the focus of approaches with respect to two dimensions. (See [Section 3](#)). More specifically, RQ2 aims to answer two sub-research questions RQ2.1 and RQ2.1.

RQ2.1 assesses generic criteria that any process model would need to address regardless of its application genre. RQ2.2 is related to the evaluation of cloud-specific aspects of available migration approaches. This decomposition is a first step in the synthesis of the evaluation framework which we will later use to identify and highlight a rich collection of key activities and recommendations that existing approaches include. In summary, the contributions of this paper are the following:

- To provide a deep understanding of the current state of migration approaches proposed in the literature, understand insightful activities and recommendations to be learned,
- To help both researchers and practitioners in the cloud community if they want to capture key facets of existing approaches and select or discard one or collection of them that may suit their needs for a particular migration exercise, and
- To give a broad view of research challenges, specifically concerned with process models for the legacy to cloud migration that need to be investigated by researchers. Hence, a gateway to new research opportunities can be opened.

This paper is structured as follows: In [Section 2](#), we give a general review of terms related to the cloud migration, key challenges that need to be addressed in a typical migration process, and the related work to this paper. [Section 3](#) describes proposed evaluation framework designed for the purpose of this paper. [Section 4](#) presents the research methodology that was adopted to conduct the current study. [Section 5](#) reports the findings of the review after identifying the existing approaches from the literature. [Section 6](#) discusses the remaining challenges and promising directions for future research. [Section 7](#) presents the limitations of this survey. Finally, [Section 8](#) concludes this paper.

2. Background and related work

As with all new areas of study, an etymological analysis is instructive. This is first undertaken in this section to give some clarity as to what a cloud migration and methodology mean in the context of cloud computing ([Section 2.1](#)). This section then identifies technical and organisational concerns of the legacy to cloud migration ([Section 2.2](#)) and provides a review of surveys related efforts ([Section 2.3](#)).

2.1. Etymology

- *Cloud migration and methodology.* In software engineering (SE) a software development methodology can be defined as a *systematic way of doing things in a particular discipline* ([Gonzalez-Perez and Henderson-Sellers, 2008](#)). Another definition can be borrowed from [Avison and Fitzgerald \(2003\)](#): *a recommended collection of phases, procedures, rules, techniques, tools, documentation, management and training used to develop a system.* A methodology organises the coordination of development team members, integration of project activities, and specifies when certain activities, which contains sequence and input/output artifacts, should be carried out.

Migration of legacy applications to the cloud signifies that an organisation has already in place existing software applications earmarked to take advantages of cloud services. A common understanding of the term *cloud migration*, as offered by [Chauhan and Babar \(2012\)](#), is the reengineering process of legacy applications for becoming cloud-enabled. That is, migration to the cloud is a kind of software reengineering where the target application will be able to interact or become integrated with cloud services. Another definition, offered by [Andrikopoulos](#), views the cloud migration process as a set of architectural adaptations/modifications required to ensure a legacy application becoming cloud-compliant ([Andrikopoulos et al., 2013](#)). Similarly, [Kwon et al.](#) pose the term *cloud refactoring* in which code transformation mechanisms are used to integrate legacy applications and cloud services ([Kwon and Tilevich, 2014](#)). Another yet broader and workable definition, which covers both technical and non-technical aspects of the cloud migration is suggested by [Pahl et al. \(2013\)](#) as: *A cloud migration process is a set of migration activities carried to support*

an end-to-end cloud migration. Cloud migration processes define a comprehensive perspective, capturing business and technical concerns. Stakeholders with different backgrounds are involved.

One can envisage a cloud migration methodology as an extended traditional software development methodology to enhance its capability to support moving legacies (or development) to the cloud.

- *Legacy Application.* This paper focuses on approaches addressing the migration of legacy applications to cloud environments. As such, we also analyse the term *legacy*. In software engineering literature many definitions can be found for the term legacy applications. One of the earliest definitions is the following: *large software systems that we don't know how to cope with but there are vital to our organization* (Bennett, 1995). Similarly, Stonbraker mentions that *a legacy application is any system that significantly resists modification and evolution* (Brodie and Stonebraker, 1995). Sneed states they are *information systems that have been in use for years* (Sneed, 2006). Others emphasise technological aspects. E.g.1, Stone distinguishes legacies as those that are not Internet-dependent (Stone, 2001). E.g.2, Dedek defines it as *an aggregate package of software and hardware solutions whose languages, standards, codes, and technologies belong to a prior generation or era of innovation* (Dedek, 2012).

A common finding in all above definitions is a dialectic tacit dependence and acknowledgment of the worthiness of the legacy applications. Holland explicitly mentions that *legacy applications encapsulate the existing business processes, organization structure, culture, and information technology* (Holland and Light, 1999). Likewise, legacies have been characterised as *massive, long-term business investment, and crucial to an organization* (Bisbal et al., 1999). Legacies are one of the major components of organisations, represent business services and repository of knowledge of organisations, and they can provide a significant competitive advantage with a positive return and contributing to the organisation revenue and growth (Bennett, 1995; Sneed, 1995; Erlikh, 2000).

From the technical point of view, the term legacy application is often collocated with a very old generation of technologies, standards, protocols and programming languages such as FORTRAN, COBOL, or C languages, and old indexed database and file systems. They are often associated with old mainframe applications, with hardware support and operational costs that are responsible for enormous transaction processing and supporting thousands of users and concurrently accessing numerous resources. Nevertheless, modern client-server software applications, which have been developed using the latest tools and technologies available in the marketplace such as .Net Framework and J2EE, but currently do not satisfy new business requirements are considered as legacy (Khadka et al., 2013; Sneed, 2006). A common architecture style of enterprise applications is 3-tiered, i.e. a user interface tier, business logic tier, and data tier. Each tier can have multiple components which can be deployed in different servers and collaborate together.

- *Different types of legacy migration to the cloud.* Taking into account the major cloud service delivery models, i.e. IaaS (infrastructure as a service), PaaS (platform as a service), SaaS (software as a service), one can view there are several possibilities in order to make legacies cloud-enabled to utilise cloud services. In this survey, these are called the variant types of the legacy to cloud migration. From this angle, a cloud migration methodology can then be viewed as a systematic process model to perform one or more migration type(s). Inspired by the classification introduced in Andrikopoulos et al. (2013), Table 1 presents definitions and examples of such

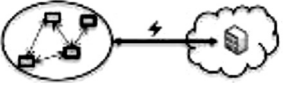
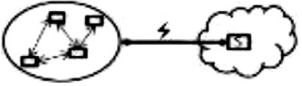
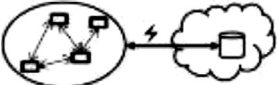
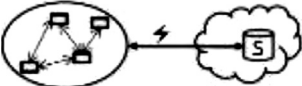
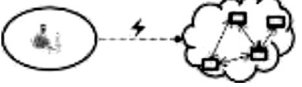
variants. Each migration type may raise different concerns. For example, if a migration type II is intended, where a legacy application is reengineered to SaaS, then multi-tenancy aspects such as application customisability and resource provisioning are needed to be properly addressed in the legacy application. On the other hand, in the case of encapsulating an application into a virtual machine and deploying it in the cloud (migration type V), enabling the feature multi-tenancy might be of less or not concerned. As another example, the migration of a legacy relational database to a NoSQL cloud database service (type IV) may raise incompatibility issues between the functionalities of the legacy relational database tier and equivalent ones offered by NoSQL cloud database. Cloud migration approaches may be designed to define a process model for a particular migration type while ignoring the others.

2.2. Key concerns in application migration to cloud environments

It is suggested that since cloud environments are not mature and secure enough, some legacies such as safety critical software applications (e.g. military, aviation, and aerospace) might not benefit directly from the cloud (Marston, Li et al. 2011, Andrikopoulos et al. 2013). Other applications such as some business enterprise applications can be reengineered to be cloud-enabled. Moving such applications to the cloud seems similar to conventional legacy application reengineering. However, such a transition should also satisfy special concerns attributed to the cloud. Drawing on the general literature on the cloud computing (Fox et al., 2009; Brebner, 2012; Rimal et al., 2009; Guo et al., 2007; Nathuji et al., 2010; Toosi et al., 2014; Dalheimer and Pfreundt, 2009; Ristenpart et al., 2009), we identified six cloud intrinsic key concerns as follow: (i) resource elasticity, (ii) multi-tenancy, (iii) interoperability and migration over multiple-clouds, (iv) application licensing, (v) dynamics and unpredictability, and (vi) legal issues. These concerns trigger considerations that an application owner should consider them in the migration process. The remainder of this section delineates these six concerns.

- Resource elasticity.** The cloud environment can be viewed as an infinite pool of resources such as CPU, memory, storage, and network bandwidth which can be acquired and released by applications based on demand (Fox et al., 2009; Brebner, 2012). Nevertheless, running an application on the cloud does not provide elasticity per se, rather the application needs to optimise resource usage in the case of fluctuation in workload. Many legacies might not have been implemented with a support of dynamic scaling up/down of resources. They assume that the elasticity is supported by providing more powerful physical servers. Inevitably, the legacy architecture needs refactoring and modifications to support the elasticity. Addressing the resource elasticity is concerned in the migration types I, II, and V.
- Multi-tenancy.** In the cloud, each service consumer is called a tenant (Rimal et al., 2009; Guo et al., 2007). Multi-tenancy is an ability to use the same instance of a resource at the same time by several tenants. On the side of cloud service provider, multi-tenancy maximises the resource utilisation and profit since only one application instance is required to deploy in the cloud. On the cloud consumer side, each consumer feels that he/she is the only users of the application. Tenants can customise application components, such as user interface appearance, business rules and sequence of workflow execution, and last but not least the application code. However, migration from a single-tenant architecture to multi-tenant one raises several issues, specifically in the case of applying the migration type II. As tenants are

Table 1
Definitions of the migration types.

Migration	Symbol	Migration type definition
	<div style="display: flex; justify-content: space-around;"> Legacy app. Cloud enabled app. </div>	
Type I		Deploying the business logic tier of legacy application (e.g. WS-BPEL), which offer independent and reusable functionalities, in the cloud infrastructure by applying the delivery model IaaS. In this migration type, the data tier remains in local organization network. Deploying an image-processing component of an application in E2C, is an example of this migration type.
Type II		Replacing some components or whole legacy application stack with an available and fully tested cloud service, by applying the service delivery model SaaS. The Salesforce CRM application is a typical example of SaaS, which can be integrated with other applications via its interfaces.
Type III		Deploying the legacy database in a cloud data store provider through IaaS delivery service model. The components related to business logic tier are kept in local organisation network and the database is deployed in public cloud data store e.g. Amazon Simple Storage Service (S3), Amazon Elastic Block Store (EBS), Dropbox, Zip Cloud, and Just Cloud.
Type IV		Converting the data, schema, and modifying data tier of legacy data tier to a cloud database solution provider e.g. Amazon SimpleDB, Google App Engine data store, or Google Cloud SQL.
Type V		Deploying the whole application stack in the cloud infrastructure via service delivery model IaaS. The application is encapsulated in a single Virtual Machine (VM) and then is run in the cloud. Hosting a Web application and its Web server as a VM on E2C is an example of such migration.

dedicated to the same instance of an application, there is often a risk that tenant's QoS is negatively affected by other tenants. For example, the performance of a tenant that uses one core of a multicore processor may significantly be reduced when another tenant runs an adjacent core and performs a massive workload (Nathuji et al., 2010). The tenant isolation for QoS satisfaction (e.g. performance, security, availability and customizability) should be addressed in a cloud application.

- (iii) **Interoperability and migration over multiple-clouds.** The cloud environment is proliferated with numerous services which bring a wide range of possible building blocks to develop a fully-fledged cloud application. Interoperability becomes an issue if an application is built via a composition of cloud services that are developed by different providers. Each provider may use different underlying technologies and proprietary APIs to develop its services (Toosi et al., 2014). These issues face developers to heterogeneities across the application tiers, which imply a certain level of development effort, specifically in migration types I, II, III, IV, and V. Advancements in the cloud computing is still on on-going track and there is not a common standard for development cloud services. Due to incompatibilities between cloud platforms, moving applications from a provider to another provider is a challenging process.
- (iv) **Application licensing.** The advantage of the elasticity offering by the cloud may raise a licensing issue in particular for the migration types I, II, and V. For example, assume an organisation has contracted for K number of application licenses and pays a fixed annual fee. Once encapsulated into a virtual machine and run in the cloud, multiple instances of the application are created by a server based on the workload fluctuation. As such, the restriction on K instances is unintentionally violated. It seems that traditional licensing model for commercial applications is not workable

for cloud-based software. In some cases, a dynamic licensing mechanism is implemented in the application by its owner (Dalheimer and Pfreundt, 2009). Alternatively, the licensing issue can be resolved via a negotiation between the application owner and provider.

- (v) **An unpredictable environment.** Cloud services may not be accessible all the time due to some reasons such service outage, network failure, transient problems in network, and service middleware failure. As an example, the sudden crash of Amazon EC2 cloud in 2011 caused the website of several high-profile companies down for hours (Blodget, 2011). Data loss was minor though it could be very harmful. Developers should empower the application with proper countermeasures to deal with such behaviours though in some cases unpredictability can be out of the control of either application owner or cloud provider.
- (vi) **Legal issues.** Beyond the technical aspects as mentioned above, organisations have a common concern about the exact location where their application components will be hosted, executed, and data processed (Ristenpart et al., 2009). In the cloud, there is no longer an assumption about the location of an application. It may move between different servers based on the server workload and load balancing mechanism defined by the cloud provider/consumer and network traffic. An application owner might not be able to determine the exact location of the code execution. While cloud providers are also a target for malicious cyber-attack, the application security can be compromised (Hay, Nance et al. 2011). To protect application components for confidentiality, developers need to provide mechanisms in the application to ensure the security of sensitive data within legal boundaries, specifically those components may be relocated between servers in different geographical areas, specifically in the case of applying the migration types I, II, III, IV, and V.

2.3. Related surveys

While there are several published surveys on different aspects of cloud computing, to the best of our knowledge, there is not a survey devoted to review the literature on existing approaches, proposing in a process model for moving legacy applications to the cloud. Perhaps, the closest studies to this paper are those related to migrating legacies to SOA because SOA and cloud computing share similar characteristics such as using services as basic blocks to build reliable and secure applications (Yi and Blake, 2010).

Razavian and Lago (2015) report a systematic literature review of SOA migration approaches. Their key goal of the survey is to identify commonalities and differences between 75 identified approaches and propose a reference model of typical activities that are carried out for the legacy to SOA migration. Khadka et al. (2013) provide a historical review of methodologies for the legacy to SOA migration. The objectives of this review are (i) to reach a broad understanding of existing process models for legacy evolution to SOA (ii) identify available techniques to perform migration activities, and (iii) identify the existing issues and possible directions for future research. Through evaluating 121 primary studies using an evaluation framework, inspired from three traditional reengineering methodologies namely Butterfly (Bisbal et al., 1997), Renaissance (Warren and Ransom, 2002), and Architecture-Driven Modernization (ADM) (Khusidman and Ulrich, 2007), the authors conclude that there is still a lack of adequate automation level and techniques for determining the decomposability of legacy applications, investigating organisational perspective of migration, and postmortem reports on after-migration experience. In another attempt, Lane and Richardson (2011) present a survey of process models to develop service-based applications with a focus on dynamic adaptation. Subsequently, they developed a meta-model giving an overarching view of development processes of 75 identified methodologies. On the basis of evaluation results using this meta-model, they found that increasing the automation of development process using model-driven development techniques is the most common theme in the reviewed methodologies. They also argued that existing methodologies suffer from a lack of real empirical validation. Even though the abovementioned surveys are helpful, they are silent to address the cloud-centric challenges stated in Section 2.2.

Jamshidi et al. (2013) identified: (i) the main drivers which motivate organizations to move their legacies to the cloud, (ii) different types of migration activities might be performed, (iii) techniques and tools, and (iv) existing gaps in the literature. Twenty-one studies on the cloud migration were evaluated against a characterisation framework including contribution type, evaluation method, means of migration, migration type, migration tasks, intents of the migration, migration tool support, and constraints. In another review, the REMICS consortium presents the state of the art with respect to modernisation methodologies and tools (Barbier and Hein, 2011) that support the automatic transformation of legacy application components to cloud environments in a model-driven fashion. However, none of the preceding surveys place particular focus on approaches for moving legacy applications to the cloud. A number of other review papers were also identified but they fall outside the scope of this survey. For example, Girish and Guruprasad (2014) focuses on six data frameworks for the cloud migration. Furthermore, Medina and García (2014) reviews research related to live migration mechanisms of virtual machines which move from one host to another over cloud network. Other studies such as Singh and Chana (2012) and Taher et al. (2012) report general challenges of cloud-based application development in terms of cloud-based architecture, component-based development and reusability, quality, design, and security.

The survey provided in the current study is different from the existing reviews in three salient aspects. Firstly, this survey limits its focus on all extant approaches proposing a (complete or partial) migration process model or framework for the cloud migration, and hence is more specific than the above-mentioned surveys. None of the reviewed surveys (see Table 2) provides an in-depth discussion on the features and migration activities proposed in the existing approaches as well as useful experience of applying these approaches in practice. Secondly, this survey provides a meticulous analysis of existing approaches through an evaluation framework, which encompasses 28 criteria classified into two dimensions i.e. generic and cloud-specific ones. The proposed framework has been derived from an extensive literature review and validated through a Web-based questionnaire survey of 104 experts from academia and experts in the field of cloud computing. Since all related surveys fail to consider the important evaluation criteria that the proposed framework includes, the proposed evaluation framework is an important contribution of the current study. The characterisation framework of Jamshidi et al. (2013) does not include any generic criteria as offered by our proposed evaluation framework. For the cloud-specific dimension, although 11 criteria have been referred by their framework, there is no elaboration on assessment of approaches. Thirdly, given our different focus, none of the related work covers the papers that this paper reviews. We found that only 5 out of our 43 reviewed papers were covered by Jamshidi et al. (2013). Finally, this survey considers different and recently published approaches that are not covered in the other surveys. With respect to this, this survey can be viewed as complementary one to the above-mentioned surveys through investigating different and recent approaches.

3. Evaluation framework

We propose an evaluation framework leaning heavily towards assessing software development methodologies, allowing us to classify and characterise approaches applicable to the cloud migration, and answer to our research questions. The following meta-criteria, suggested by Karam and Casselman (1993), were used to define a fair evaluation framework that is: (i) sufficiently general to all methodologies, (ii) precise enough to characterise the similarities and differences of methodologies, (iii) and comprehensive enough to cover all important requirements of methodologies. Given that, a two-dimensional set of criteria was developed: generic criteria and cloud-specific ones as shown in Fig. 1.

For the generic criteria dimension, we reviewed and synthesised various existing frameworks that define criteria attuned to evaluate software development methodologies in SE. These frameworks were Karam and Casselman (1993), Wood et al. (1988), Ramsin and Paige (2008), Sturm and Shehory (2004) and Tran and Low (2005). Once the criteria in the above sources were analyzed and redundancy and overlapping among them were removed, 11 distinct criteria were derived for the purpose of this study including (1) Process Clarity, (2) Procedure and Supportive Techniques, (3) Tailorability, (4) Development Roles, (5) Modelling Language, (6) Traceability, (7) Work-Products, (8) Formality, (9) Scalability, (10) Tool Support, and (11) Domain Applicability. A detailed description of these criteria is presented in Appendix F. Section 5.3 motivates and elaborates each criterion by providing a detailed explanation for each, along with an evaluation result against existing approaches.

For the second dimension, the framework was expanded with cloud-specific criteria that were deemed important and relevant to legacy application migration to the cloud. This was initially inspired by the study introduced in Strauch et al. (2014) and

Table 2
Related surveys on SOA/Cloud migration.

Authors and title	Study type	Published channel	Aim	Publication year	Total reviewed	Time period of included studies
M. Razavian, P. Lago, "A systematic literature review on SOA migration"	Systematic literature review	Journal of software: evolution and process	To review and analyse existing SOA literature in order to identify commonalities and difference between designed migration approaches. It designs a new conceptual model including a classification of activities carried out for the legacy to SOA migration.	2015	75	2003–onwards
R.Khadka, A. Saeidi, "Legacy to SOA Evolution: A Systematic Literature Review"	Systematic literature review	Book chapter	To provide a broad understanding of process model required for legacy evolution to SOA, available techniques to conduct migration activities, existing issues, and possible directions for future research.	2012	121	2000–August 2011
S.Lanea,I.Richardsona, "Process Models for Service Based Applications: A Systematic Literature Review"	Systematic literature review	Information and Software Technology	To assess existing process models for service-oriented application development on the basis of a generic process metamodel and identify categorised activities within those process models focus area of existing methodologies.	2011	57	October 2009
P.Jamshidi, A.Ahmad, "Cloud Migration Research: A Systematic Review"	Systematic literature review	IEEE Transactions on Cloud Computing	To identify the main drivers which motivate organisations to migrate legacies to the cloud, existing approaches and different types of migration activities, techniques, and tools for migration.	2013	21	2005–2013
REMICS Consortium "State of the art on modernization methodologies, methods and tools"	Review	REMICS (Public deliverable)	To review traditional modernisation methodologies and tools and argue potential requirements for designing cloud migration methodologies.	2011	NA	NA
Girish, "Survey on Service Migration to Cloud Architecture"	Review	Journal of Computer Science and Engineering Technology	To identify commonalities and difference of some existing cloud migration methodologies.	2014	22	NA
V. Medina, J. Garc, "A survey of migration mechanisms of virtual machines"	Review	ACM Computing Surveys	To identify live migration mechanisms of virtual machines moving from one cloud to another.	2014	14	NA
S.Sukhpal, C. Inderveer, "Cloud Based Development Issues: A Methodical Analysis"	Review	Journal of Cloud Computing and Services Science (IJ-CLOSER)	To identify general issues in development cloud-based applications in terms of cloud-based architecture, component-based development and reusability, quality, design, and security.	2013	89	NA
Y. Taher, D. Nguyen, "On Engineering Cloud Applications - State of the Art, Shortcomings Analysis, and Approach"	Review	Scalable Computing: Practice and Experience	To present a review of the standardisation, methodology, software, and products that support development of service-based applications in the cloud and state existing gaps in the literature.	2012	NA	NA

La and Kim (2009). They proposed a small set functional and non-functional properties that should be addressed by an ideal cloud migration methodology. These studies, however, were not complete and well-articulated; nor enough attention was paid to domain-independence, validation, and generality. Hence, we strived to identify a coherent set of analysis criteria for inclusion in the evaluation framework and accordingly to assess cloud migration approaches. The derivation of the criteria was mainly influenced by various sources in the cloud migration literature and the reviewed approaches in this survey.

A criterion was included in the proposed framework if it had addressed at least one of the concerns stated in Section 2.2 and

also sufficiently generic to cover a variety of migration scenarios regardless of a particular domain. This resulted in defining 17 cloud-specific criteria including (1) Analysing Context, (2) Understanding Legacy Application, (3) Analysing Migration Requirements, (4) Planning Migration, (5) Cloud Service/Platform Selection, (6) Training, Re-Architecting Legacy Application (including (7) Incompatibility Resolution, (8) Enabling Multi-Tenancy, (9) Enabling Elasticity, (10) Cloud Architecture Model Definition, (11) Applying Architecture Design Principles), (12) Training, (13) Test and Continuous Integration, (14) Environment Configuration, (15) Continuous Monitoring, and (16) Migration Type, (17) Unit of Migration. These criteria helped us to contrast and compare cloud-centric aspects

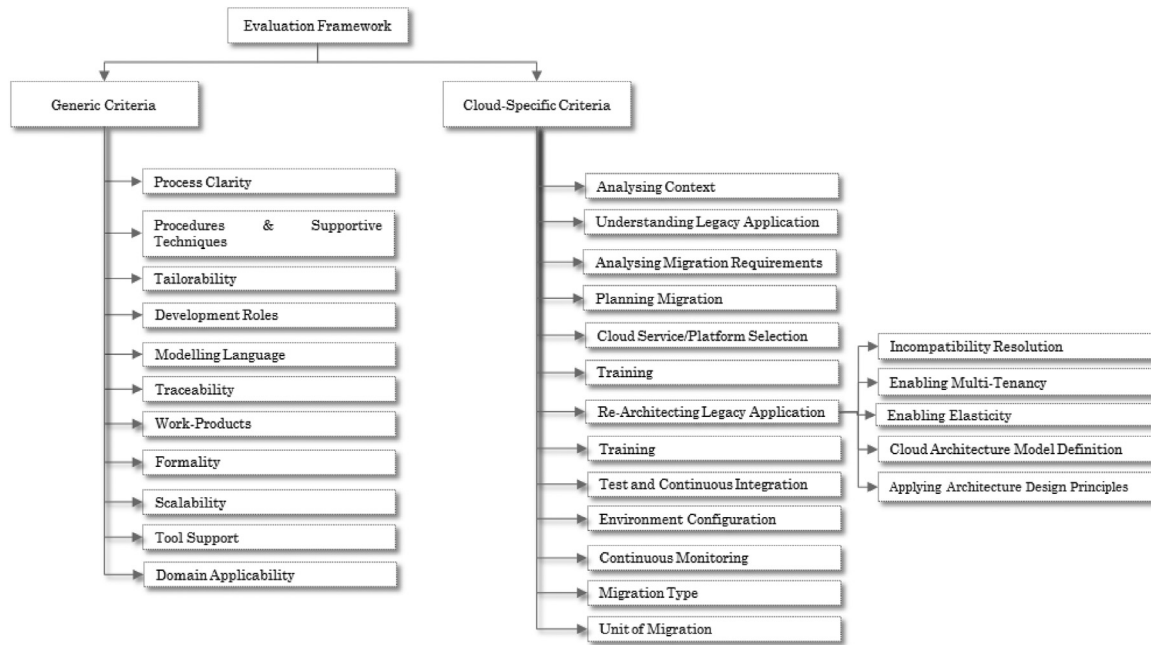


Fig. 1. Proposed evaluation framework for assessing various approaches (See Appendices 'F' and 'H' for the criteria definitions).

of existing approaches. Once these criteria were established, their importance and relevance to the cloud migration were assessed through a Web-based survey of 104 experts from academia and practitioners in the cloud computing field. This gave confidence that important criteria are covered by the framework. For further detail, we refer to the report (Fahmideh, 2015) which provides a detailed analysis of the criteria and suggested comments about the criteria by the survey respondents. The above report also includes the statistical analysis of importance of each criterion and the qualitative comments of experts to justify the importance of each criterion. The criteria are defined in Appendix H and described in Section 5.4.

Since the criteria were aimed to be measurable, evaluation questions were defined for each criterion, with four possible kinds of answer for each as suggested by Kitchenham et al. (1997): Narrative, YES/NO, Scale, or Multiple choices. For open questions a narrative answer was used; a YES/NO answer was given where a criterion was met or not; an answer that is described by scale point refers to a degree of criterion support provided by an approach. Scale points were used during the evaluation process to assign a value for the level of support of a criterion. The framework defines three levels of scale as fully-supported (●), partially-supported (◐), and not-supported (○). Finally, multiple choices

were defined when there was a possibility to choose one out of a number of answers (for example, the criterion migration type).

4. Survey

The systematic review procedure proposed by Kitchenham et al. (2009) was selected as a survey approach for the purpose of the current paper. It is a well-recognized procedure to identify, analyse, and interpret all related studies with respect to a topic of interest and includes three phases 'Planning', 'Conducting Review', and 'Documenting Results'. In the Planning phase, a protocol for the review is established which includes the defining of search strings, study sources, and study selection and inclusion criteria. These definitions are used to conduct the next phase, which is

the Conducting Review phase. During this phase, primary studies are identified from the literature, analysed against the inclusion and exclusion criteria, and necessary data items are extracted from them. In the last phase, the results of the review are documented. Appendix A describes the steps were undertaken in each phase. Consequently, 43 papers were identified, as the output of this step, for the review after applying the inclusion and exclusion criteria. Appendix B presents the list of the papers.

5. Results

Through analysing 43 identified approaches, we answered three research questions stated in Section 1. Section 5.1 presents overall demographic information about the studies including the year of publication, publication channels, authors' nationality, and the quality assessment of the papers. Sections 5.3 and 5.4 describe the analysis of the studies against each dimension of the proposed framework.

5.1. Overview of approaches

Quality assessment. While a new research that contributes to the literature is regarded as important, its findings might not be reliable if it suffers from a lack of appropriate research quality (Kitchenham et al., 2002). Nine criteria provide a measure of the extent to which a research could contribute to the literature. Table 1.1 (Appendix I) shows the results of the assessment. Given the defined questions in Appendix C, the grading of each criterion was based on three scales: 'fully supported', 'partially supported', and 'not-supported'. Fig. 2 depicts the distribution of the identified approaches in terms of their support for the criteria.

According to Table 1.1 the majority of the approaches have clearly stated a research aim. As many as 25 out of the 43 approaches have provided contextual information about the environment in which the research had been conducted. Seven approaches stated a general description of the research context whilst 11 ones did not state any information. As far as the criterion Research Design' is concerned, only 6 studies (14%) [S4], [S15], [S17], [S20], [S31] and [S36]) provide a clear description of the research

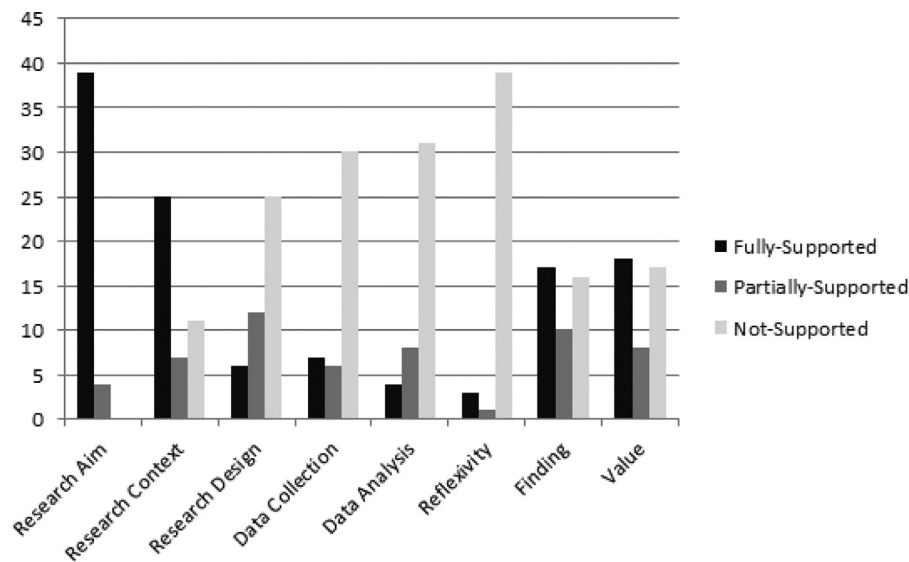


Fig. 2. Quality scores for the identified studies.

Table 3
Validation type.

Study	Validation type	Description	Number
[S1], [S5], [S6], [S7], [S8], [S9], [S10], [S12], [S14], [S15], [S16], [S17], [S20], [S22], [S23], [S25], [S27], [S29], [S30], [S37], [S39]	Case study	Case study has been used to investigate the approach within real-life or exemplar context.	21
[S1]	Questionnaire	Experts were asked to answer questions about the suitability and applicability of the suggested approach.	1
[S4], [S15], [S18], [S34]	(Semi-structured) interview	Experts were interviewed about the quality of the suggested approach.	4
[S18], [S34]	Focus group	A group of experts were asked about their opinions towards the approach in an interactive group setting.	2
[S2], [S3], [S32]	Example	Application of the approach has been demonstrated through an illustrative example.	3
[S11], [S36]	Simulation	A mathematic simulation used to assess the approach correctness.	2
[S7], [S13]	Theoretical evaluation	The approach has been validated using a set of high-level criteria.	1
[S26], [S33], [S35], [S43]	Industrial experience	The approach has been developed on the basis of gained experience in an industrial experience.	4
[S19], [S21], [S24], [S27], [S28], [S31], [S38], [S40], [S41], [S42]	Not stated	The approach did not specify any applied validation.	10

design of the study. As many as 12 of the 25 studies did not sufficiently describe their research design. Additionally, an overall view of the scores in Table I.1 (Appendix I) reveals that a large portion of the studies have not addressed the criteria 'Data Collection', 'Data Analysis', and 'Reflexivity'. More exactly, as many 30 and 31 studies ignored to report how data had been collected and analysed for the validation of the proposed approach. It can be seen that 39 studies (90%) did not report how researchers have been involved with validation environment. From this observation, one may conclude that the research methodology has been viewed as a subsidiary task by the designers of approaches. We discuss this issue in Section 6. Back to this table, it can be seen that 17 approaches explicitly stated their contributions to the literature. Only two studies (5%) [S15], and [S31] achieved a full score on the quality assessment, and studies [S1], [S7], [S21], [S23], [S38], [S42] received the lowest score in this review.

For the criterion 'Validation Type', the studies were classified according to the applied validation type as shown in Table 3. The majority of approaches applied the case study technique (21 studies), followed by techniques such as interview (4), industrial experience (4), example (3), focus group (2), simulation (2), theoretical evaluation (1), and questionnaire (1). Four of the studies (i.e. [S1], [S15], [S18], and [S27]) used a combination of validation types. For example, [S1] reported a case study in addition to a questionnaire. Of 43 approaches, 10 (23%) did not report any information about the validation.

5.2. RQ1 what are existing approaches proposing a migration model for moving legacy applications to cloud environments?

Appendix D synthesises a description of the approaches, ascendingly sorted based on published year. The appendix does not aim to present a critique of the existing approaches, rather it abstracts the approaches and gives a broad understanding of their perspectives to the cloud migration process, and facilitates for further investigation, elaboration and improvement of the approaches. In Appendix D, the third column presents the theoretical foundations used to design the approaches. According to this column, the motivation of the existing approaches varies between different streams as presented in Table 4 and described in the following.

Table 4
Classification of approaches based on theoretical foundation.

Foundation	Approach	Number
No specific foundation	[S2], [S3], [S6], [S7], [S8], [S9], [S10], [S12], [S16], [S17], [S18], [S19], [S21], [S22], [S23], [S24], [S25], [S27], [S28], [S29], [S30], [S33], [S34], [S35], [S36], [S37], [S40]	28
Optimization	[S11]	1
IT Capability Maturity Framework	[S4]	1
Agile Scrum	[S1]	1
Model-Driven Development	[S14], [S15], [S26], [S41], [S43]	5
Software Product Line	[S13], [S15]	2
Software Patterns	[S20], [S42]	2
Supply Chain Lifecycle	[S38]	1
Ontology-Based Reengineering	[S39]	1
Conceptual Model	[S31]	1
Total		43

- *Model-driven development*. The most applied paradigm is the model-driven development with 7 approaches. Its primary goals are portability, interoperability, reusability of applications as well as increasing development speed. The approaches have adopted model-driven development to transform the legacy application models (e.g. codes and architecture) to platform-independent models, configure them to generate platform-specific cloud applications using model transformation techniques. As an example, REMICS [S26] is a model-driven methodology with a special emphasis on cloud-enabled legacies that can be run on multiple clouds.
- *Software product line (SPL)*. The idea of SPL relies on techniques for developing a set of similar applications satisfying the requirements of a specific domain. In [S13] authors propose a five-phased process model which incorporates software product line techniques such as domain analysis and variability modelling for various requirements into SaaS migration (Type II). The approach highlights commonality and variability in legacy applications and customises them for various cloud platforms. Likewise, the approach suggested in Guillén et al. [S15] uses a combination of SPL and model-driven development approaches. It defines the notion of *cloud artefacts* which are high-level models of target cloud application. Cloud artefacts are produced based on cloud variability models and transformed to multiple cloud platforms.
- *Agile development*. Incorporating Agile practice such as light process, short release, and continuous testing into the cloud migration approaches process have received attention in the cloud community. In the approach proposed by Krasteva and Stavru [S1], authors pose whether legacy modernisation processes can benefit from Agile practice. They suggested a Scrum-based extension of REMICS migration methodology.
- *Software patterns*. Patterns are reusable and good-enough abstract solutions for recurring problems during the software development. They can be used for developing applications through an appropriate composition of their instances. Applied in the cloud migration, Jamshidi et al. [S20] proposed 15 fine-grained patterns as a sequence of architecture refactoring activities to integrate legacy components with cloud services.
- *Ontology-based reengineering*. Ontologies express an area of interest in a communicable and formal way. The ontology-based approach offered by Zhou and Yang in [S39] is a reengineering process whereby an ontology of enterprise legacy applications is built and then decomposed into potential service to be migrated to the cloud.

- *Supply chain lifecycle*. Lindner et al. [S38] state that the cloud migration forms a supply chain model of different interconnect cloud service providers and consumers. A service is provided at the start of the supply chain and a consumer at the end uses this service. A proper cloud supply chain model is required to understand requirements of both providers and consumers during this end-to-end migration process. Given that, they define a breaking down all of the activities and steps, which aids an organisation to migrate legacies to the cloud.

5.3. RQ2.1 What generic criteria, as typically expected for a software development methodology, are supported by these approaches?

Appendix E presents the extent to which each approach supports five criteria Process Clarity, Procedure and Technique, Modelling Language, Tailorability, and Tool Support. These criteria were measured using three levels of scale (See Appendix F). The following subsections detail the definition of each criterion in the first dimension of the evaluation framework and justify the situation in which the criterion is important. The evaluation results against this dimension also reported. For those studies that provided a full support of criteria, we narrowed our analysis to identify any interesting recommendations and reported experience for presenting in this survey.

5.3.1. Process clarity

Approaches differ in clarifying and specifying the detail of phases and activities. Some of them offer the most prescriptive ones whilst others, deliberately or not, provide a short description of the activities and leave a space for an arbitrary interpretation of how activities can be carried out. The degree of prescription is a helpful criterion for developers or organisations who are not familiar with the cloud computing technology and seek a step-by-step methodic guidance on what phases and activities should be followed to accomplish migration. According to Table E.1 (Appendix E), 35 out of 43 (81%) approaches were rated as fully-supported as they provide a description of their suggested migration activities. Seven out of 43 approaches provided a general or short description of migration phases and activities with no depth, rated as partially-supported. The only study which was rated as not-supported was the conceptual process model suggested by Jamshidi and Paul [S31], listing 20 key activities for a migration process as a result of a literature review on the cloud migration but without providing any details of them.

Table 5
Identified development roles.

Role	Approach	Responsibility
Business Analysts	[S8], [S18]	Elaborating requirements, motivations for the cloud migration, objectives to be achieved, and cost analysis.
Systems Analysts	[S8]	Analysing technical incompatibilities between legacy application components and cloud services.
Project Managers	[S8]	Identification of potential cloud platforms/services.
System Architects	[S8], [S18]	Identifying a potential cloud hosting solution, analysing technical incompatibilities between legacy application and a chosen cloud platform, design potential architecture solutions, evaluating QoS of cloud, and incremental architecture scoping and definition.
Developers	[S8]	Implementing or re-factoring the legacy architecture for moving the legacy application to a target cloud platform.

5.3.2. Procedures and supportive techniques

It is desirable that the approach elaborates its prescribed activities by defining supportive techniques and examples in a way that developers can simply understand and apply the approach. Hence, if developers seek detailed advice on how to perform activities or producing models rather than top-level descriptions, those approaches that properly describe steps to conduct activities take precedence over others. Twenty-three of 43 approaches provide examples of how their prescribed activities can be performed. However, 14 approaches provide guidelines for some activities but not for the whole process. It was found that 6 approaches do not offer any advice on performing activities. Some examples demonstrate how approaches offer techniques to conduct activities (more detail on various supportive techniques is described in [Section 5.4](#)). To resolve incompatibilities between legacy applications and cloud services, the suggested approach in [S35], proposes typical patterns such as emulators, proxies, and data aggregators in order to migrate the data tier of legacies to a cloud database solution. As another example, a common concern in designing a new cloud-based architecture for the legacy application is to define a proper deploying and distributing of legacy components in cloud servers. With respect to this, the approach proposed by Leymann et al. [S11] refers to the legacy deployment in the cloud as a *Move-to-Cloud* Problem and transform it into a graph partitioning problem. They use existing optimisation algorithm (e.g. simulated annealing) to find optimum distribution of legacy application components on different cloud platforms.

5.3.3. Tailorability

Like any software development projects, a methodology should be fine-tuned to characteristics and contingencies of a given cloud migration scenario at hand. Hence, the thought of a universal cloud migration methodology can meet all migration scenarios is viewed fallacious. While methodology adherence can be beneficial, constructing situation-specific methodologies or tailoring existing ones that meet project characteristics at hand should not be overlooked. As an example, according to [Louridas \(2010\)](#) there is a difference between the US and EU for addressing the ultimate to data protection in the cloud. That is, in the US, a cloud provider is responsible for completely data protection whilst in EU, the cloud consumer is responsible to ensure if the cloud provider satisfies data protection requirements. Ignoring this fact may affect the application security in the cloud and raise exposure to vulnerabilities. As such, security-related activities to protect legacy data tier should be incorporated into the migration process.

Given that the tailoring effort is an important part of the cloud adoption, it deserves to examine if an approach provides mechanisms or guidelines for its tailoring to be properly applicable. The review showed that most of the existing approaches, except for [S9], [S10], and [S26] which partially inform a need for methodology tailoring, are based on a one-size-fits-all assumption. Proposed methodologies in [S9] and [S10], jointly, refer to tailoring as *methodology extensibility*. REMICS [S26] methodology is a set of method fragments that can be further selected and assembled regarding characteristics of a project at hand. ARTIST methodology

[S43] provides a tool which allows the customisation and instantiation of the base methodology regarding the characteristics of a given migration scenario. Nevertheless, none of them provide a guidance on the creation of a situation-specific methodology. Following the idea of situational method engineering ([Brinkkemper, 1996](#); [Henderson-Sellers and Ralyté, 2010](#)), Jamshidi et al. [S20] pose an assembly-based approach wherein a methodology is constructed through combining architectural migration patterns. But, there is no further elaboration to accommodate configuration and tailoring.

5.3.4. Development roles

It deserves approaches to define roles, required expertise, and responsibilities associated with these roles in the course of the migration process. This can be helpful for developers who have limited experience in the cloud migration and are not quite clear sure about these roles. Furthermore, depending on chosen migration type ([Table 1](#)), different roles may be required or existing ones to be tailored. For example, deploying the whole legacy application stack in the cloud (the migration type V) would need a role who can analyse application workload and data storage growth; whereas in the case of moving business tier to the cloud (the migration type I), a reverse engineer would be required to discover the logic of legacy code blocks. An interesting finding in this survey was that, except for the methodology proposed by Chauhan and Babar [S8], the majority of the existing approaches had not defined required roles. In another approach by Pahl [S18], roles are briefly described but have not been linked to the migration activities. [Table 5](#) shows the description of suggested roles by the existing approaches. More discussion on this issue is presented in [Section 6](#).

5.3.5. Modelling language

An approach may specify a particular notation and semantic rules for expressing the outcome of each migration activity, i.e. models (or work-products). Modelling will increase the automation and productivity of development process. If developers have already been using tools along with a supported modelling language, this criterion can be helpful to select those approaches that can be supported or integrated with existing tools. The modelling criterion in the evaluation framework examines if an approach uses a particular notion to describe outputs from the activities as well as the degree of support, i.e. partial or whole migration life cycle. In this regard, 29 approaches did not use or suggest any modelling languages. Twelve approaches apply a modelling language for some migration activities. Only REMICS [S26] and ARTIST [S43] provide a fully support this criterion for their entire lifecycles. Besides, the approaches vary in applying the type of modelling languages. For example, REMICS [S26] and ARTIST [S43] use UML. On the other hand, some approaches such as [S20], [S29], [S36], and [S40] keep modelling at the level of simple graphical diagrams and texts. [Table 6](#) shows the approaches that use a modelling language, whether with a partially-supported or full-supported, along with their aim to use.

Table 6

Approaches incorporating modelling language into their suggested migration process.

Study	Modelling language	Aim
[S1]	UML	Modelling in general
[S11]	UML	Representing a deployment model of the cloud-enabled application, i.e. re-arrangement and provisioning of the application components in the cloud.
[S12]	AST and SoaML	(i) AST (abstract syntax tree) for the source code representation (ii) SoaML for modelling individual service interfaces, service implementation, and the architecture of target application.
[S13]	UML	Modelling commonality and variability in the target domain for which a legacy application is reengineered to a SaaS application.
[S15]	Feature model and XSD	(i) Describing target cloud platform using feature model (ii) Describing cloud deployment model using XSD schema diagram.
[S17]	Simple Block Diagram	Representing a legacy architecture model and its corresponding cloud deployment model.
[S20]	Simple Block Diagram	Representing cloud migration patterns to modify legacy application architecture.
[S21]	Graph Modeling	Legacy source code representation
[S26]	UML	In general
[S29]	Simple Block Diagram	Modelling application architecture before and after adding a support for multi-tenancy and architecture decoupling.
[S36]	Simple Block Diagram	Modelling the deployment of the application in the cloud.
[S37]	UML	Using state chart to model how application workload is moved from a cloud to another cloud.
[S39]	UML and OWL	Representing the legacy application architecture using UML and transforming it to ontology (UML to OWL model transformation). Then partitioning the application ontology into potential service candidates.
[S40]	Simple Block Diagram	Modelling legacy application architecture and dependency tree.

5.3.6. Traceability

Traceability refers to the relationships between work-products in the whole process model in the sense that a model becomes a refinement of another model and all models can be traced to high-level requirements. This criterion, which pioneered by object-oriented methodologies, is important in order to understand the linkages of a work-product to its previous and next work-products and to manage software changes during the lifecycle. Approaches were examined if they specify mechanisms or guidelines to refine abstract models into more detailed models. One interesting observation in this review was that the traceability is weakly supported in the existing approaches. Only 5 approaches ([S8], [S11], [S13], [S26], and [S43]) define the chain of work-product changes as mentioned in the following. In the approach by Leymann, et al. [S11], models of a legacy application architecture are gradually enriched by specification of related to legacy deployment in the cloud. Five approaches provide support for the traceability between particular activities but not for whole migration process: [S12] (Legacy code model → architecture representation → architecture redesign), [S21] (Requirement analysis → Migration plan, Legacy code → Code model → Legacy architecture → Cloud-service architecture model), [S41] (Legacy model → Target architecture → Mapping model → Constraint violation), [S43] (Source code → model understanding). Like many other areas of software development, the traceability is a crucial concern and cloud migration approaches need to take it into account.

5.3.7. Work-products (artefacts)

An integral part of every methodology is to define necessary work-products as the outcome of each activity throughout the lifecycle. An approach that specifies work-products can be adopted easier by developers using modelling tools in daily development activities. The approaches were scanned to identify any recommended work-product. It was found that 27 of 43 approaches defined required work-products during the execution of the migration process. As approaches varied and played English words for the naming of work-products, the identified work-products were classified based on their similarity, resulting in 11 distinct work-products. In this regards, the first and third columns of Table 7, respectively, show the work-product and different naming used by the approaches. From this table, the most frequently recommended work-products are *Legacy Application Architecture Model* and *Cloud Architecture Model* as stated in 14 and 13 approaches, respectively. *Migration Plan* and *Cloud Provider Profile* are prescribed by 4 approaches.

5.3.8. Formality

This criterion examines the extent to which an approach provides mathematical, unambiguous, and precise mechanisms for the modelling language, representing work-products, and the relationships among them. The majority of the approaches do not support formal definitions. The approach by Leymann, et al. [S11] applies the graph partitioning problem and simulated annealing for selecting a subset of legacy components and optimizing the distributing them to different cloud servers. Moreover, Aakash and Ali Babar [S21] provide a set of mathematical operators for the transforming legacy codes and architecture models to cloud-enabled ones. But the definitions of the operators still required to be completed. Obviously, if developers seeking for an approach that supports mathematical modelling and reasoning, the current literature is silent.

5.3.9. Scalability

Scalability is the applicability of a methodology to be used for various migration project sizes. Some methodologies can be suitable for moving large and complex workloads from traditional data centres to cloud infrastructures whilst others might be best suited for a partial and light migration. A migration methodology should be examined if it is appropriate to handle the intended scale of migration. A methodology supports the scalability if it explicitly defines activities that are concerned with workload size, project size, the degree of interconnectivity of legacy applications, the number of legacies to be migrated. None of the reviewed approaches supports the scalability nor refers to it.

5.3.10. Domain applicability

A particular interest about the approaches is to understand the application domain for which they have been designed. Approaches were classified on the basis of their adherence to a particular domain. As a result, 18 domains were identified, as shown in Table 8. Twenty-three approaches did not clarify the applicable area that they might suit.

5.3.11. Tool support

To automate migration activities, an approach can offer its tool or alternatively refer developers to existing third-party tools in the cloud market. With respect to this criterion, 34 of 43 (79%) approaches did not offer any tools (Appendix E, Table E1). As tool support for the whole migration process may not be practical, approaches vary in their focus and provide a partial automation support. As an example, the tool suite (e.g. cost modelling, suitability analysis, energy consumption, and stakeholder impact analysis) offered by the model of Khajeh-Hosseini et al. [S6] aids an

Table 7
Methodologies prescribed work-products during migration process.

Work-product	Migration Type	Aim	Referred names in studies	Frequency
Requirement model	All	A set of computational requirements, motivations, and objectives for the migration to the cloud.	System Requirements [S8], Recovered System Requirements [S26]	2
Cloud provider profile	All	The profiles of potential cloud providers which can satisfy legacy application requirements. The profile characterises the providers in terms scalability (vertical/horizontal), availability, security, inter-operability, billing model, and storage capacity.	List of Potential Cloud Environment [S8], A shortlist of suitable suppliers [S4], Selected Cloud Environments [S8], Selected cloud offering model [S43]	4
Cloud architecture model	All	An architecture model which specifies the distribution and deployment of legacy application components in the cloud regarding criteria such as network latency, data transfer, regulations, privacy, and geographical locations of components. The model also indicates which application components are to be migrated to which cloud and which components are kept in the local network.	Cloud Distribution [S11], Cloud Deployment Model [S22], Cloud Deployment Model [S28], Cloud Deployment Architectures, Cloud Component Model [S36], Target Architecture [S41], Cloud Architecture Model [S29], Deployment Diagram [S11], Architecture Diagram [S11], Cloud-service Architecture [S21], Finalized Design Decision And Modified System Architecture [S8], Cloud outsourcing model [S4], Architecture [S18]	13
Interaction diagram	All	Representing how application components are interacting in the a-synchronised and loosely coupled cloud environments.	[S36]	1
Application variability models/templates	Type II	These models capture features/parameters and the variation points of the application components in the sense that application can be configured for different deployment settings and execution in cloud platforms. Variability models facilitate application interoperability between different cloud platforms.	Variability Models [S13], Cloud Variability Models/Feature Model (deployment model, code model) [S15], Variability Model [S29]	3
Migration plan	All	A plan to guide how to conduct the migration process regarding migration needs and feasibility analysis.	Migration Plan [S21], Roll-out plan [S38], Migration Plan [S42], Plan [S40]	4
Cloud risk management	All	Remedy actions to handle cloud risks.	A Cloud risk management strategy [S4]	1
Legacy architecture model	All	An architectural description of the legacy application, the dependency between the components, data model, and interfaces. This model helps developers in a better understanding of current state of the legacy application and provides an insight of required effort to resolve incompatibilities between legacy application and target cloud solution.	Legacy Architecture [S21], Knowledge Model [S26], Application profiling, Application Dependency Mapping, Application Architectures [S27], Application Model [S30], Source Code Ontology, Database Ontology, Enterprise Ontology [S39], Legacy Architecture [S40], Dependency Tree [S40], Taxonomy of legacy artefacts [S43]	12
License agreement	All	A signed contract with cloud provider(s) indicating a model for resource consumption cost.	SLA and Pricing document [S4], Pricing Model [S40]	2
Virtual machines	Type V	Specifications of running virtual models of application in the cloud.	Implementation Unit (Virtual Machines) [S11]	1
Certification model	All	The confidence of end-user about the quality of migrated application and its compliance with the cloud.	Certification Model [S43]	1

Table 8
Application areas of the migration approaches.

Study	Domain
[S1], [S4], [S5], [S7], [S11], [S13], [S16], [S18], [S19], [S21], [S22], [S23], [S24], [S28], [S30], [S31], [S34], [S35], [S38], [S40], [S41], [S42], [S43]	Not specified
[S2]	Commercial Medical Application
[S3]	e-Tourism
[S6]	University
[S20]	Customer Relationship Management
[S8], [S39]	Open Source System
[S9]	e-Science applications (scientific workflow)
[S10]	Enterprise Resource Management
[S12], [S27]	Oil and gas
[S14]	Document Processing
[S15]	E-bank
[S17]	Document management systems
[S25]	Telecommunication
[S26]	ERP
[S29]	Medical Communications System
[S32]	Stock Market
[S33]	Digital Publishing
[S36]	e-Commerce
[S37]	Strategy Management Systems (SMS) based on Linux systems

Table 9

Approaches providing tool for migration activities.

Approach	Activity	Migration Type	Availability	Aim
[S6]	Context analysis	V	Not-specified	Supporting decision making during the suitability analysis of migration in terms of operational cost, organisational change, energy consumption, and stakeholder analysis.
[S9]	Incompatibility resolution	IV	Available	Supporting the data tier migrating to the public cloud and the refactoring of the application components.
[S11]	Design cloud solution	V	Not-specified	Allowing developers to model application components and define relevant criteria for the splitting components in the cloud.
[S39]	Legacy application understanding	All	Available	Extracting an ontology of source code, data, and architecture of the legacy application.
[S26]	Legacy application understanding	II	Available	Discovering legacy application components.
	Resolving incompatibilities between legacy and cloud services			Resolving interoperability problem between legacy components and third-party cloud services.
	Design cloud architecture			Generating a platform specific implementation model from the cloud architecture model.
	Test			Creating test cases and test scenarios.
	Deployment			Generating a platform specific implementation model from the architecture model by taking into account the constraints relevant to the deployment of applications in the cloud.
[S40]	Elasticity (resource provisioning)	V	Available	Configuring amazon services (e.g. storage, resource).
	Resolving incompatibilities between legacy and cloud services			Integration legacy components with amazon cloud services.
	Network configuration			Configuring network and dynamic resource provisioning (e.g. elasticity, virtualization, auto-scaling).
[S43]	Context analysis	II	Available	Analysing business and technical migration feasibility.
	Legacy application understanding			Representing high-level and platform-specific models of the legacy application.
	Incompatibility resolution			Transforming legacy codes to the cloud target platform.
	Test			Supporting functional and non-functional testing.
	Deployment			Deploying application components in the cloud.

Table 10

Migration types and the classification of approaches based on the migration types.

Migration	Approach	Number
Type I	[S1], [S3], [S5], [S15], [S17], [S18], [S35]	7
Type II	[S1], [S5], [S8], [S12], [S13], [S15], [S17], [S18], [S26], [S34], [S35], [S41], [S43]	13
Type III	[S1], [S14], [S15], [S17], [S18], [S34], [S35]	7
Type IV	[S5], [S8], [S9], [S10], [S15], [S17], [S18], [S19], [S25], [S29], [S34], [S35]	12
Type V	[S6], [S8], [S11], [S15], [S16], [S18], [S22], [S23], [S25], [S28], [S29], [S30], [S33], [S34], [S35], [S36], [S40], [S42]	18
In general	[S2], [S4], [S7], [S20], [S21], [S24], [S27], [S31], [S32], [S37], [S38], [S39]	12

organisation to explore concerns in the early phase of the migration. Among them, the cost modelling tool is the most mature one which examines the cost of different deployment options of a legacy application in the cloud. In general, 7 approaches provide tools for some activities in their suggested methodology.

Currently, two methodologies argue for the inclusion of a wide range of tools for the entire migration process: (i) ARTIST methodology [S43] proposes Eclipse-based suite which is tightly integrated with its activities. Since produced work-products are stored in a shared repository, they can be accessed and modified other tools; and (ii) REMICS methodology [S26] includes a set of tools related to the areas such as requirement management, knowledge recovery from legacy applications, re-transformation of application components to cloud architecture, and model-based testing. Finally, as far as availability of an approach's tool is concerned, it can be attributed as publicly available, proprietary, or not specified. Table 9 shows whether an approach fully or partially offers tools. A full description of the activities in this table is presented in Section 5.4.

5.4. RQ2.2 what cloud-specific criteria are supported by these approaches?

The goal of the second dimension of the framework is to characterise existing approaches in support of various activities attuned to the cloud migration process. Table G.1 (Appendix G) shows the

extent to which each criterion is satisfied by each approach on the basis of three scale points including 'fully-supported', 'partially-supported', and 'not-supported' (See Table H1 in Appendix H). The rest of this section qualitatively elaborates scores in Table G.1. We also highlight recommendations and advice offered by these approaches as a result of applying them in real scenarios.

5.4.1. Migration type

As mentioned in Section 2.1, there are several possibilities to make legacy applications cloud-enabled (Table 1). This criterion indicates the migration type for which an approach has been designed. Table 10 shows the classification of approaches based on the migration variants. Along with the migration type, 18 out of 43 approaches define support for moving the whole application to the cloud, i.e. Type V. The idea is to encapsulate application stack into virtual machines, deploy them in IaaS infrastructure, and add an additional layer to orchestrate the virtual machines. Twelve approaches provide support for moving the data tier of legacy applications to a cloud database solution along with defining modifications to the application to access the data layer in the cloud (Type IV). As many as 13 and 7 approaches, respectively, propose activities for the migration type I (i.e. deploying business logic of an application in the cloud) and type III (i.e. deploying a database in a cloud data storage). Thirteen approaches focus on defining a reengineering process to make legacy components cloud-enabled using existing application development APIs offered by PaaS or

Table 11

Classification of approaches based on the unit of migration.

Unit of migration	Study	Number
Organisation	[S4], [S6], [S38]	3
Whole application stack	[S1], [S2], [S5], [S7], [S8], [S11], [S12], [S13], [S14], [S15], [S16], [S17], [S18], [S21], [S20], [S22], [S23], [S24], [S25], [S27], [S28], [S29], [S30], [S31], [S32], [S33], [S34], [S35], [S36], [S37], [S39], [S40], [S41], [S42], [S43], [S44]	36
Data tier	[S9], [S10], [S19]	3
Business logic tier	[S3]	1

SaaS delivery models (Type II). Some studies are not classified under any particular migration types; rather they suggest an overall process model for the migration. Twelve approaches were fallen in this group. The migration types can be used simultaneously to make an application cloud-enabled. For example, the business logic tier and data tier may be migrated to Google App Engine and Amazon Relational Database Cloud Service, respectively. This is why some approaches are repeated in the second column of Table 10.

5.4.2. Unit of migration

For many reasons such as regulations, privacy laws, and network latency organisations may not migrate the whole legacy application stack to the cloud; instead some legacy components are migrated whilst other are kept in local organisational network and cloud services are offered to them. Hence, it deserves to investigate if an approach fit for moving a particular tier or whole application stack to the cloud. Given that, this criterion indicates which application tier is concerned for migration by the approach. Table 11 shows the result of classification of the approaches with respect to the unit of migration. According to this table, the majority of approaches focus on moving all the application tiers. Studies [S4], [S6], and [S38] view all legacy assets in an organisation as a single unit for migration and propose a general approach for moving all legacies to the cloud. Only study [S3] investigates running a business logic tier, described using Web Service Business Process business logic tier Execution Language, in the cloud via IaaS service delivery models.

5.4.3. Analysing context

Transition to the cloud is not merely a technological improvement of existing applications but also it is a substantial change in the way legacy applications hereinafter operate, provide IT functions, and to be maintained. As an important first activity in the cloud migration process, a feasibility and business value analysis of the cloud adoption as an IT strategic weapon to empower legacies should be conducted. In this regard, this criterion can be used to examine if an approach defines activities for the context analysis. In this survey, 27 approaches (62%) did not provide any suggestions on the context analysis, and 7 approaches (16%) only refer to this activity without any elaboration on how to perform it (Appendix G).

It was found that 9 approaches out of 43 (20%) explicitly include the context analysis. For example, Conway and Curry report their experience in the validation of IVI Cloud Computing Life Cycle [S4], an Agile migration process model, within a number of organisations. They observed those organizations that did not fully understand the impact of the migration on the user community and IT support staff and failed to plan accordingly either lost key resources or experienced resistance from the IT and user community – both during and after the migration. In the Cloudstep, a model proposed by Berserra et al. [S2], authors point the idea [context analysis] is to anticipate the detection of potential organisational constraints that might affect the cloud migration decision, before carrying out any further analysis of the application itself.

In the migration approach proposed by Khajeh-Hosseini and Greenwood [S6], the main focus is on the early phases of the mi-

gration process and it suggests three kinds of activities named as Technology Suitability Analysis, Energy Consumption Analysis, and Stakeholder Impact Analysis. They state that *understanding the significance and the extent of the organisational changes associated with cloud adoption is a difficult challenge. We argue that enterprises need to understand the breadth of changes and the effort required to make these changes in order to understand their benefits.* The ARTIST methodology [S43] defines an activity called Business Feasibility Analysis of Legacy Migration which is to *provide not only economic information (ROI, payback, etc.) but also what are the main risks to be faced with the migration and the organizational processes affected by the uptake of the new business model.* Ahmad and Babar in their proposed framework [S21] stress two concerns during conducting context analysis, i.e. (i) determine the type of the application is to be migrated to the cloud since some applications may not benefit from the cloud such as safety-critical or embedded applications, (ii) effort and cost that required for the migration regarding perceived benefits. Table 12 summarises the identified concerns from 43 reviewed approaches that should be taken into account when performing a context analysis. With respect to these concerns, the Cloudstep [S2] advises these concerns are not mutually independent, rather their influence on each other should be investigated prior actual migration execution.

5.4.4. Understanding legacy application

It is common that the knowledge about legacy applications be outdated, imperfect, and undocumented. Identifying any characteristics of legacies that may influence migration process is important. To this aim, items such as recapturing an abstract As-Is representation of legacy architecture, functionalities, different types of dependencies to other applications, interaction points and message follows between legacies, and the quality of code blocks for reuse and adaptation are important. Wu et al. in [S33] report their experience in moving an online digital library search engine (CiteSeerX) to Amazon EC2 and found that *an up-to-date and complete documentation of legacies can significantly reduce the length of learning and investigation time.*

The legacy understanding can be helpful in many ways: Some approaches such as [S12], [S21], and [S41] use produced high-level architecture models of the legacy application as a mean for automatic transformation to target cloud platforms. For example, CloudMIG [S41] proposes OMG's Knowledge Discovery Meta-Model (KDM) for extracting a meta-model of application and transforming it to a target cloud architecture. Based on the experience of moving legacy applications to IaaS, the approach of Sun and Li in [S16] incorporates legacy application understanding in order to get an estimation of required development effort for enabling legacies to utilize cloud services. In the approach by Rajaraajeswari et al. [S27] the activity *Application Profiling* is to identify the application usage data which helps to understand the size of workload to migrate to the cloud. This avoids from unexpected cost of cloud service usage when bills will be issued. Specifically, the approach emphasises the following data to collect (i) CPU and memory usage, (ii) storage data such as size and input/output operations per second, (iii) network data such as throughput and connections per second (iv) the node-level data such as number and type of

Table 12

Concerns should be investigated during context analysis.

Concern	[S2]	[S6]	[S4]	[S21]	[S27]	[S31]	[S23]	[S40]	[S43]
User resistance	✓	✓	✓				✓		
Loss of governance	✓								
Dependency on legacy application	✓			✓			✓		
Risk of unauthorised access	✓						✓	✓	
Legal restriction	✓				✓				
Physical location of IT resources	✓								
Energy consumption		✓							
Variation on responsibilities		✓							
Impact on organisational and daily activities		✓	✓						
Type of legacy application				✓	✓		✓	✓	
Required efforts and migration cost				✓					
Scalability (workload fluctuation)						✓	✓		
Technical suitability of cloud		✓					✓		✓
Financial suitability of cloud						✓		✓	✓

legacies. As shown in Table 10, 15 approaches (35%) explicitly define activities related to the legacy application understanding, targeted for the cloud. However, 22 studies (51%) did not offer any activities or guidelines regarding this criterion. Table 13 summarises a collection of useful recommendations, each stresses a particular aspect of legacy application understanding, identified from approaches along with supportive technique to perform.

5.4.5. Analysing cloud migration requirements and objectives

As the name implies, approaches may define activities to specify and model the expected functional non-functional requirements to be fulfilled through moving legacies to the cloud. Examining this criterion gives an insight if developers seek approaches that assist in capturing, analyzing cloud migration requirements and objectives, and suggesting possible trade-off. While common requirement engineering techniques (e.g. interview, prototyping, and workshop) available in SE literature still are applicable in the context of the cloud migration as it can be seen in approaches [S8] and [S26], some approaches have added a focus on expected elasticity and scalability application requirements in the cloud [S18], computing requirements of legacies [S19], inter-operability requirements for deployment in the cloud [S21], security and regulatory requirements [S23], and storage space requirements for the application in the cloud [S33]. According to Table 10, 15 (35%) of reviewed approaches define related activities for the requirement analysis. However, 7 (16%) and 26 (60%), respectively, partially or does not support this criterion.

5.4.6. Planning migration

Once cloud migration is accepted as a feasible decision by project stakeholders, a plan guiding the rest of migration process is required. Feedback from stakeholders is used to define a migration plan. Support for activities related to the planning has been included in 12 reviewed approaches (28%).

According to [S27], information obtained from the legacy of application understanding is an input source for the planning. For instance, if legacies X and Y are using the same database server, a possible migration plan would be a combined move or splitting the dependencies.

Approaches vary in recommended factors that need to be considered during the planning as mentioned in the following. Strauch, et al. propose a methodology for moving e-science applications to the cloud by acquiring services from Amazon AWS and Microsoft Windows Azure [S9]. Planning in this methodology is defined as a set of actions in order to resolve potential incompatibilities between legacy components and target cloud database solution without modifying the application business logic. Additionally, Pahl et al. [S18] take into account parameters such as cloud service provider capabilities, addressing contract with cloud providers, the

distribution of project team, the capabilities of migration team (e.g. technology, skills, and tools), and defining metrics and milestones for the planning. Furthermore, in the process model of Legacy-to-Cloud Migration Horseshoe, proposed by [S21], authors incorporate the influence of cloud provider selection and the migration type (Table 1) as main factors to develop a migration plan.

Finally, defining a proper rollback plan, i.e. switching to the previous version of the application at any stages of migration reduces the risk and exposure to organisation business. The approach [S4] highlights that *the option to roll back to an in-house version at any stage significantly reduced the risk and exposure to the business. Organisations that experienced difficulties in the transition to cloud computing missed vital steps in their planning*. Similarly, [S33] recommends defining *Backward Availability* for critical migration projects in the case new cloud application fails.

5.4.7. Cloud service/platform selection

Approaches can be assessed based on the extent to which they properly define activities to identify, evaluate, prioritize, and select a set of candidate cloud services that might suit the requirements of an IT-based organization and application. In justifying this criterion, Chauhan and Babar examined their suggested process model for moving an open source system to Amazon Web Service and Google App Engine [S8]. They found that a better alignment between the quality attributes of application and cloud services can reduce migration effort and cost. They conclude that suitability of cloud environments in term of security and sensitivity is pivotal to the success of application migration the cloud. Similarly, from a cost perspective, Tran et al. [S5] report a breakdown of activities performed during moving a .NET n-tier application to run on Windows Azure and highlight efforts required for deciding cloud providers is major. They underline the influences of selecting cloud platforms on the required effort for the rest of migration process. If the selected cloud platform technology is highly compatible with the legacy, fewer code modifications in the legacy are required.

In general, from 43 reviewed approaches the following important factors were identified for investigation when selecting cloud services/platforms: *variety of service models offered by provider, price model, the form of SLA, additional services such as backup, monitoring, auto-scaling, security mechanisms, implementation technologies which are supported by a provider such as programming languages, development platforms, allowing access to internal operational logs, physical location of application data* [S2], *sustainability* [S4], *commitment durability with cloud provider, business objectives* [S8], *degree of automation, storage encryption mechanisms, storage format and exchange, developer SDKs* [S9] and [S10], *required configuration effort, deployment speed, consistent development experience* [S17], and *traffic bandwidth at the client network* [S29].

Table 13

Summarised recommended activities by the existing approaches for legacy application understanding.

Approach	Activity name	Definition	Technique
[S1]	Recover activity area	Recover legacy application knowledge in a KDM format, which is as an intermediate representation supporting many programming languages, from which UML application models are generatable.	KDM
[S2]	Define application profile	Create a profile of legacy application including technical characteristic of the application that need to be considered for migration.	Not specified
[S7]	Discovery of application details from the source environment	Identify a topology graph of enterprise's IT including legacy application and then all outgoing hosted-on, depends-on, and connects-to edges starting from the application's entry node.	Deep dive
[S12]	Architectural representation of the legacy application	Reconstruct an architecture model of the legacy application and any business entities and metadata.	Not specified
[S16]	Discovery	Collect the configuration information of application including deployment model, the number of physical machines that have been deployed in the local network, the network topology of these physical machines. Use this information to determine the deployment scale in the target cloud environment.	Not specified
[S21]	Architecture recovery and consistency conformance	Recover an architecture model of legacy and check whether it is consistent with existing legacy source codes. This architecture model is necessary to find the design rationale that need to be preserved during modification/transformation of application for the cloud.	Graph-based modelling
[S23]	Identify user load parameters	Discover and document groups of users, and the pattern of their use, teams of contractors in foreign countries, access time to the application, a particular time in the day or month in which application usage is heavy.	Not specified
[S24]	Describe existing capability	Analyse different available legacy artefacts such as source codes, programming language, documents, users' knowledge, configuration files, execution logs and traces, integration with third party offerings, dependencies, database, upgrade procedures to identify artefacts to be considered in the migration project.	Top-down and bottom-up approaches
[S26]	Recover	Model "as-is" and "to-be" architecture and deployment models.	KDM
[S27]	Application dependency mapping profiling	Identify dependencies among applications in the local infrastructure and their usage data. This helps estimating the size of migration to the cloud. Based on the node on which the application is executed, the following data need to be identified: CPU and memory usage, data storage, network latency, and input/output operations per second, network data.	Not specified
[S37]	Discover	Identify static aspects of the legacy such as source configuration details, network IP, operating system type, and dynamic aspect such as CPU usage and dynamic workload characteristics of application.	Not specified
[S39]	Capture	Identify key concepts and relations in the legacy application.	Ontology
[S41]	Extraction	Extract a utilisation model of legacy application including statistical properties concerning user behaviour like service invocation rates over time or average submitted datagram sizes per request.	KDM
[S42]	Initial screening and analysis	Collect key data on existing workloads, applications and their dependencies, server utilisation, machine and operating system type, hardware and software characteristics.	Not specified
[S43]	Application discovery and understanding	Extract an overall logic of the legacy application and any different available legacy artefacts relevant for the migration scenario.	Model transformation

Once a provider(s) is selected, a contract between the legacy application owner and the cloud service provider is signed. The application owner needs to specify the scope of expected service provisioning by cloud provider prior migrating application components to the infrastructure of a cloud provider. The negotiation is finalised with producing an SLA which specifies the boundary of usage and provision of cloud services. According to IVI lifecycle model [S4], *building a relationship with the cloud supplier was the key to success in many of the projects we studied*. As mentioned in Section 2.3, cloud environments may raise an application licensing issue meaning that by running the application in the cloud, many instances of applications might be created by the servers based on the workload, leading to an unintended licensing violation. From the reviewed studies, 4 approaches incorporate addressing licensing issues. For example, Amazon methodology [S40] recommends that for legacy applications that organisation is tightly coupled with complex third party applications, which have not been migrated to the cloud, developers should extend the legacies with implementing new components (e.g. VPN tunnel) in a way that cloud services can be indirectly offered to them. [S23] suggests enabling a license tracking mechanism through monitoring connections between the application and cloud resources.

5.4.8. Re-architecting legacy application

The aim of the re-architecting is to evolve the legacy architecture towards a new architecture which can utilize cloud services. The legacy architecture and requirement models which are outputs of legacy application understanding and requirement analysis (sections 5.4.4, 5.4.5), respectively are used during the re-architecting process. Several concerns are needed to consider as described in the followings.

- (i) **Define cloud architecture model.** One important aspect of re-architecting is to find suitable legacy components for migration and their optimum distribution on the cloud servers. Conway et al. [S4] state *this [component selection] will require an understanding of the current state, so that it can be compared to the desired future state*. Andrikopoulos et al. [S35] numerate several factors are taken into account for legacy component selection such as data privacy, expected workload profile, acceptable network latency and performance variability, the availability zone of cloud providers, the affinity of components in the cloud, and the geographical location of cloud servers. Among the reviewed approaches, Leymann et al. [S11] propose a concrete technique for a cloud

architecture design via transforming application architecture into a graph partitioning problem and using simulated annealing to find optimum distribution of legacy components on different cloud servers.

- (ii) **Enabling application elasticity.** Running applications in the cloud does not help to resource efficiency and scalability by itself. Rather applications should have mechanisms to acquire and release resources, effectively, under input workload (Vaquero et al., 2011). Often legacy applications have not been designed with a support for dynamic resource provisioning. Elasticity can be implemented by developing a new component either embedded in the application or separately hosted on a server node in order to continuously monitor the application/component resource usage variables and then performs appropriate action for the resource acquisition and release based on scaling rules/conditions related to specific workload threshold, events, or metrics. Only Ridha et al.'s approach introduced in [S22] defines generic activities for deploying existing applications with a support of the elasticity. The activities are as follow: (i) describing required computing resources, the life cycle of the application and its offered services, and scaling rules of the application and its services, (ii) automatically provisioning resources from selected clouds for the application, and (iii) monitoring and ensuring application performance. Two approaches rather provide high-level advice: Andrikopoulos et al. [S35] point out aspects that needed to be considered during enabling elasticity in an application in terms of what to scale, how to scale, when to scale, which scaling strategy to be used, and scaling latency. Likewise, with respect to the approach of Maenhaut et al. [S29], to handle elasticity it is required the components of an application to be decoupled and communicate in a-synchronised manner.

- (iii) **Enabling multi-tenancy.** According to Bezemer and Zaidman (2010) and Guo et al. (2007), re-architecting of a legacy application to support multi-tenancy includes the following aspects:

- The first aspect is security isolation. Enabling multi-tenancy, specifically in the case of applying migration type II (Table 1), raises security risk as different tenants use the same application instance and use the same database and shared resources. It is necessary to assure each tenant can only access to its data and to be protected from unauthorised access by other tenants which are running in the same cloud. As off-the-shelf database management systems might not support multi-tenancy (Jacobs and Aulbach, 2007), securing the data tier of application should be properly addressed. Furthermore, the code blocks of the application reflecting organizational business processes (e.g. BPEL) might need to be secured prior deploying in the cloud to protect from unauthorized access by other tenants. The confidentiality of code execution is assured, for example, through encryption mechanisms, in the sense that no other tenants will be able to access, read, or alter the code blocks within the running application instance in the cloud.
- The second aspect is application customisability. Each tenant may have different functional and non-functional requirements. For example, one tenant might want to have the ability to customise application user interface whilst other one need to change the sequence of application workflow and code customisation. Customisations are applied by a tenant should not affect other tenants (e.g. application availability). To realize application customisability, configuration points in the form of applica-

tion template should be defined in the code blocks of the application.

- The third aspect is fault isolation meaning that if a fault occurs in the same instance of a running application being used by multiple tenants in a shared manner, it should not be propagated to other tenants using that instance. The application should monitor its internal state to detect faults, prevent their propagation, and repair them in a timely manner.
- The fourth aspect is performance isolation which is to guarantee the performance of one tenant from the negatively being affected by the performance usage of other tenants in unforeseen behaviours. If this is satisfied in all situations, then the application is performance-isolated.

Given the above mentioned aspects of the multi-tenancy, only the approach proposed by Maenhaut et al. [S29] incorporates explicit activities to add a customisability isolation to legacy applications to move to a hybrid or public cloud environments. That is, enabling multi-tenancy requires the following steps: (i) decoupling databases, (ii) adding tenant configuration databases, (iii) providing tenant configuration interface, (iv) dynamic feature selection, (v) managing tenant data, users, and roles, and (vi) mitigating security risks.

Andrikopoulos et al. [S35] point out the concept *multi-tenant aware applications* and recommend addressing two aspects of multi-tenancy as (i) supporting message exchange isolation per tenant, and (ii) administrating and configuring the application per tenant. Other approaches only state multi-tenancy aspect without any proposals on required activity to perform.

- (iv) **Incompatibility resolution.** If legacy application components are to be bound to cloud services, potential incompatibilities between the legacy application and these services should be identified and accordingly resolved through adaptation mechanisms. With respect to the existing approaches, three forms of adaptations might be required to resolve incompatibilities.

- *Code refactoring.* A basic form of incompatibility resolution is through modifying legacy codes so that they can interact with the cloud services. This can be the modification of legacy application interfaces in order to remove mismatches (e.g. interface signature, operation ordering, operation names, message format, interaction protocols, and data type) with cloud service interfaces. Also, behavioural adaptation is performed if there is a mismatch in the granularity of message interaction between application components and cloud services. An explicit support for a code adaptation was found in 11 approaches, i.e. [S3], [S5], [S8], [S9], [S10], [S12], [S14], [S15], [S25], [S35], and [S43]. Other approaches such as [S7], [S13], [S17], [S20], [S21], [S24], [S26], [S29], [S30], [S31], [S33], [S39], [S40], [S41], and [S42] only explain general advice on the code adaptation.
- *Developing integrators/adaptors.* If the code refactoring, as mentioned in the previous item, is costly, then developing integrators/adaptors (also called wrappers) can be served as an alternative solution to remove incompatibilities. Adaptors provide an abstraction layer, keeping the legacy codes untouched and facilitating application interoperability and portability. In this regard, the ADM (Architecture-Driven Modernisation) approach of Zhang [S12] suggests developing wrapper layers (e.g. Web services) on legacy components. Some approaches include high-level classifications of the possible adaptors that developers should build to integrate legacies with cloud services. In this way, CMotion approach [S30] defines

Table 14

Summarised of resolution mechanisms to resolve incompatibilities between legacies and cloud services.

Study	Mechanism	Legacy tier	Definition
[S9]	Data store functionality extension	Data tier	As a cloud database solution may not support functionalities that legacy data tier offers (for example stored procedures), the data tier is extended with missing functionalities of cloud database solution.
[S9]	Local database proxy	Data tier	Cloud data stores might not offer horizontal scalability for data reads. To address this feature, the cloud data store is configured through applying a single master/multiple slave model where the master manages writing data and slaves are replicas for read operations. All data traversing from each data tier is conducted through a proxy.
[S12]	Web-Service generation	Application stack	An indirection access to cloud services is created via Web-Service technology.
[S14]	Adapter generation and injection	Business tier	The adaptor consists of a proxy with a concrete implementation of the service interface injected into the application as an instance of the component required for consuming the selected cloud service.
[S15]	Service adapters	Application stack	Developers should identify required adaptation points in the application and define intermediate artefacts which automatically injected between interaction points of legacy components and cloud services.
[S30]	Adaptor	Application stack	Adaptors are classified into two dimensions: 'when' and 'how'. The 'when' dimension denotes times that adapters are applied in the applications. The 'how' dimension represents how adaptors apply their changes.
[S20]	Hybrid refactor with cloud adaptation	Application stack	A component adapter (e.g., legacy façade) is adopted to provide an interaction point between the legacy components and those re-hosted cloud-based components.

two types of integrators focusing on aspects: *when* and *how*. The former is to define adaptors for transformation application components before deployment in the cloud whilst the latter is to define ways of applying changes, i.e. automatic vs. manual adaptors. A similar view has been also defined by Miranda et al. [S14]: static (or design-time) adaptation, which includes the adaptations required to be performed prior the application is run in the cloud; and dynamic (or run-time) adaptations at the runtime application execution in the cloud. Jamshidi in [20] presents a catalogue of patterns to enable legacy components to interact with different cloud services.

- *Data adaptation*. Migrating the legacy data tier to a cloud database solution may cause different mismatches. For instance, although cloud database solution SQL Azure is similar to traditional SQL Server, distributed transactions are not supported by SQL Azure as SQL Server does. This may imply a need for the data type conversion, query transformation, database schema transformation, and developing runtime emulators. The methodology suggested by Strauch et al. [S9] and [S10] provide a richest of *Cloud Data Patterns* in order to address incompatibility issues between traditional databases and cloud database solutions. The patterns are classified under three groups as (i) *functional patterns* (e.g. data store functionality extension, emulator of stored, procedures, and local database proxy), (ii) *non-functional patterns* (e.g. local sharding-based router), and (iii) *confidentiality patterns* (e.g. confidentiality level data aggregator, confidentiality level, data splitter, filter of critical data, pseudonymizer of critical data, and anonymizer of critical data). Additionally, the methodology suggested by Oracle [S19] defines the following activities for database migration: (i) database schema migration (involves migrating tables, indexes, and views in a database), (ii) data migration, (iii) database stored program migration triggers, and views, (iv) application migration, and (v) database administration script migration.

Table 14 summarises the list of identified mechanisms proposed by the existing approaches for developing integrators and data adaptors.

- (v) **Applying architecture design principles**. From the reviewed approaches, a few recommended architecture design principles was identified which need to be considered by develop-

ers when re-architecting a legacy application to target cloud. According to [S8], [S14], [S17], [S22], [S34], [S36], and [S42] these principles are as follows:

- *Application decoupling*. To take the advantage of dynamic deployment, one architectural requirement is to decouple application components so that they can interact in a transparent manner. The application decoupling enables independent elastic scaling of the components (i.e. dynamically adding/removing more instances of the same components), minimise time required for code refactoring and test in the case of switch to another cloud provider, handle a-synchronised communication, as well as simplify coping with the components failures.
- *Stateless programming*. To support dynamic deployment and independent component scalability, components should minimise using contextual data. Moreover, the co-existence of multiple running instances of the same components in the cloud environment requires a new session management mechanisms in order to track tenant's behaviours and ensure their security when running application instances are increased or decreased based on the server workload.
- *Transient fault handling*. Performance variability of cloud servers and network latency between local network and the cloud can have a negative impact on QoS of migrated application. Developers should implement mechanisms in the application to detect and handle transient faults that occur in cloud environments.

Again it should be noted that adopting the abovementioned principles mainly depends on a selected migration type (Table 1). Table 15 shows the situations in which a rearchitecting concern should be incorporated into the migration process is required to perform.

5.4.9. Training

As cloud platforms come with a particular set of tools and APIs, new skill set needs to be learned about selected cloud platforms. Some existing approaches recommend incorporating training activities for IT staff (developers and managers) as a part of the migration process. At the managerial level, there is a change from a traditional licensing model to pay-as-you-go or post-usage billing [S18]. On the other hand, developers need training on new programming concepts such as asynchronous interaction, distributed state and session management, caching, scale out across data centres and providers (scalability), and multi-tenancy [S35]. Tran et al.

Table 15
Re-architecting activities and migration types.

Re-architecting		Type I	Type II	Type III	Type IV	Type V
Component selection and distribution		Very likely	Very likely	Very likely	Very likely	Very likely
Enabling elasticity		Not likely	Very likely	Not likely	Not likely	Very likely
Enabling multi-tenancy		Very likely	Very likely	Not likely	Not likely	Very likely
Incompatibility resolution	Code adaptation	Very likely	Very likely	Very likely	Very likely	Maybe
	Developing Integrators/Adaptors	Somewhat likely	Somewhat likely	Not likely	Very likely	Not likely
	Data Adaptation	Not likely	Somewhat likely	Somewhat likely	Very likely	Somewhat likely
Applying architecture design principles	Application decoupling	Very likely	Very likely	Very likely	Somewhat likely	Somewhat likely
	Stateless programming	Very likely	Very likely	Somewhat likely	Somewhat likely	Very likely
	Transient fault handling	Very likely	Very likely	Very likely	Very likely	Very likely

state that all cloud services may not support some features offered by legacy technologies [S5]. For example, as mentioned in Section 5.4.8, SQL Azure does not support distributed transactions as SQL Server does. Hence, an understanding on handling this aspect of SQL Azure might be required. Sun et al. [S16] put one step further and define a process-based effort estimation approach to measure the capability of a development team on the basis of mastery of developers on conducting migration activities.

5.4.10. Test and continuous integration

Like many other software development endeavors, the test activity includes testing both functional and non-functional aspects of migrated application to the cloud. Additionally, in the context of cloud migration, various cloud-specific tests are to perform including security test, interoperability test, and workload test. Agile REMICS [S1] defines a phase called Validation Activity which, consists of the following steps for the migration type V: (i) define testing infrastructure, (ii) identify and refine requirements to be tested, (iii) generate acceptance testing, (iv) import models elements to be tested, (v) define testing procedures, and (vi) implement testing strategy.

Other approaches, however, provide general advice on conducting tests as described in the followings. For many reasons such as workload or user preference, a running application may move from a cloud server to another one. With respect to this, a lesson learned from applying the migration framework [S15] is to test application for interoperability in heterogeneous cloud environments. Therefore, a certain level of integration and interoperability testing is required. Important tests that should be taken into account are *Data Verification*, *Test Backup and Recovery Plan* before they are required [S23], *Test Network Connectivity* (connection between cloud services and local network), *Test Connection Speed* as there is network latency to receive a response from cloud servers located in different geographical areas [S33] [S40], *Test provider Performance Variability* and *Test Application Latency* due to the network performance variability [S35].

An important aspect of the test is to address continuous integration. Continuous integration is to add a new increment to all tier of the application in the cloud as well as any hosts in a smooth and coordinated manner. Cloud provides a new environment for automated testing by adding and removing servers for test and integrating new increments to working applications. This reduces the cost of deploying, maintaining, and licensing internal testing tools. Except for [S33], the existing approaches do not provide support for the continuous integration. Wu et al. in [S33] recommend a mechanism called *staging and production* in order to use separate servers for the test activity. In the staging environment, developers can test the application within a production-like environment (e.g. infrastructure configuration). That is, a deployment environment in which all tests are performed in order to detect bugs, performance,

and other issues before the application to be used by users. All releases of application are tested on the staging environment. Once the application completely is tested, it is pushed to the target environment, i.e. production, as an increment of the application.

5.4.11. Environment configuration

Once the application migrated to the cloud, the connection between the local network and the application in the cloud is still required. Ten out of 43 approaches ([S3], [S5], [S9], [S10], [S12], [S19], [S23], [S28], [S40], and [S42]) have comprised the following activities:

- Reconfigure organisation network setting such as ports, firewalls, and anti-virus, reachability policies, and connection strings to the database.
- Set privileges on application tenants/users to assure the security of application still is satisfied in the cloud environment.
- Create installation scripts and setup different third-party libraries and tools which may be used for monitoring and reporting runtime application behaviour, though this needs less effort if a cloud provider has already done it.

5.4.12. Continuous monitoring

The dynamic and unpredictable nature of cloud environments necessitate continuous monitoring of application and cloud resources to assure successful SLAs (e.g. [S1], [S4], [S22], [S23], [S26], [S28], [S33], [S38], [S40], and [S43]) as described in followings:

- *Measuring*. Collect critical data about the application health and performance, traffic patterns, vital signs such as CPU and memory usage, network traffic, and disk usage. This data are used to detect deviations from SLA and runtime application adaptation and optimisation. Measuring can be fulfilled either by developing a new component, which is integrated with the application and is responsible for this purpose, or it can be offered by cloud service provider [S1] [S4] [S22] [S23], [S28] [S33] [S40] [S43].
- *Updating patches and periodical backup*. During the application operation in the cloud, regular patch update and database backup should be addressed either by application owner or by the cloud provider [S28] [S33] [S40].
- *Metering and managing bills*. This includes two aspects. Firstly, billing management is required to be implemented in the application by its owner to track tenants/users are using the application service. A new component might be added to the application which tracks the tenants using the services and bill them accordingly. Secondly, the application owner is billed for the cloud services (e.g. infrastructure) used for developing and maintaining the application. Hence, the owner should monitor resource usage, track logs periodically, identify lifecycle patterns of each application instance, and detect suspicious high resource usage which may cause an extra cloud service usage fees [S38] [S43].

- *Replicating and synchronising.* Replication aims to support high business continuity and minimal downtime and is realised by running several application instances on a variable number of cloud infrastructures. The partitioning and replicating application component in the cloud servers may entail extending the application for a support of synchronisation between the components (local replicas and those hosted in the cloud) [S33].
- *Terminating idle instances.* Removing under-utilised application instances to increase utilisation of the overall application [S40].
- *Decommission legacy components.* Once the application gets operational in the cloud, the old legacy components may no longer be required and hence they are retired and the assigned resources to them are released [S23].
- *Withdrawing.* In some cases, the application owner wants to withdraw the application from cloud services. If so, the application should be removed from the cloud and other acquired resources (e.g. data storage, CPU, and network bandwidth) released to reduce costs [S26], [S28].

6. Challenges and future directions

It is not feasible that an approach to cover all the stated criteria in the framework; rather, it may focus on a subset of them based on its objective and leave other criteria unaddressed. Therefore, it was hard to rank approaches from lower priority to higher. The evaluation framework can be viewed as an indicator of the primary focus of an approach instead of judgment on its completeness. In other words, if an approach does not support a certain criterion, it does not necessarily mean that it cannot be useful; rather, it means that criterion is not its main focus.

While all the reviewed approaches have merit and form a rich source of necessary activities and recommendations to be learned, our deep analysis revealed that still there are challenges which are yet to address. Given the RQ2 and in the light of the evaluation results in Section 5, overcoming the following challenges may open new possibilities to ameliorate the state of the literature.

- **Approaches suffer from a sound research quality.** Like other fields of SE, a rigorous research methodical is of utmost significance in order to advance and better understanding the field. As discussed in Section 5.1, only 6 out of 43 (14%) approaches included a section in their paper to explain their research methodology. This lack is also true for the Data Collection and Data Analysis where only 7 and 4 approaches, respectively, address these criteria. This undermines the trustworthiness of the existing approaches. We believe that using proper research methodologies will improve the quality of individual approaches and enable to combine their adoption results. A good example of applying of a sound research methodology in the field of legacy migration can be seen in the study by Razavian (2013) where she applied an exploratory action research in order to get a deep understanding of what academia and practitioners perceive about the migration process of legacy applications to SOA and identified the categories of activities that are carried out during this transition. A similar exercise can be equally applied in the field of the cloud migration.
- **Approaches are not tailorable.** There has been a well-established acknowledgment that there is no universally superior methodology and the choice of a methodology is dependent on a project setting such as time, technical or human factors, and the characteristics of the software is being developed (Sommerville and Ransom, 2005). This strand continues as a crucial theme in the cloud migration. For example, one study suggests: *There is an immense need to identify correct process model for the deployment of cloud-centric environments in order to meet changed business requirement of clients. One solu-*

tion can never fit all problems; likewise, there is a need of customised cloud for individual businesses and dynamically changed requirements of clients (Mahmood, 2013). With respect to the criterion tailorability, as defined in the first dimension of the evaluation framework, any existing approach supports tailoring mechanisms to fine-tune its suggested model/activities to meet the characteristics of a given cloud migration scenario at hand. From the reviewed approaches, only 5 approaches state a need for the tailoring effort but without proposing any means to do that (Appendix E). To address this shortcoming, one can apply a method engineering approach (Brinkkemper, 1996), that is based on this thought that instead of looking for a universal software development methodology, developers should construct a new bespoke methodology through reusing and enhancing existing methodologies to meet project characteristics at hand. Such a harness in the context of cloud migration is to develop a repository of atomic and reusable method fragments along with mechanisms to construct situation-specific methodologies through assembling these fragments which fit a given migration scenario. In the context of SOA, Börner (2010) and Khadka et al. (2011), respectively, proposed method engineering approaches for construction situational service identification and methodology for the legacy to SOA migration. We believe this kind of research can be applied in the field of cloud migration.

- **Lack of adequate support for the multi-tenancy, elasticity, test, and continuous integration.** One interesting observation from the reviewed approaches is that only [S35] covers the multi-tenancy merely from the customisation aspect. The elasticity has been also supported only by one approach [S22]. The continuous integration is partially supported by two approaches [S23] and [S32]. It seems that there is a clear lack of support in the literature when there is a need to approaches supporting these criteria. In the cloud computing literature, one can find many ad-hoc techniques related to the above criteria. Future research is to properly abstract and structure these techniques in the form of activities (or reusable method fragments) and incorporate them into existing migration approaches.
- **Developer roles have not been sufficiently honed.** The definition and responsibilities of roles that might be involved in the migration process have not been well described in the existing approaches. Specifying roles will make clear for developers their exact responsibilities and lead to better organising the migration process. Besides a need for the elaboration of roles, the existing approaches do not incorporate cloud-specific challenges into the role definition. Rather they use the common SOA roles (i.e. service provider, consumer, and broker). However, an overview on blogs and white papers in online cloud communities shows that there are new development roles and skill associated to development of cloud applications. Stafford, who is executive editor of TechTarget, gives some examples of new roles such as *continuous integration skills for real-time testing and diagnostics, virtual infrastructure configuration, Hadoop on the cloud for handling big legacy data* (Stafford, 2013). More research on necessary roles should be conducted to characterise not only simply developer roles but also any stakeholders involved in the cloud migration. Gu and Lago (2007) proposed a role-driven migration process model for service-based application development. The model has two dimensions where a horizontal view explicitly shows the activities relevant to roles in SOA and vertical view shows the interaction and cooperation between the roles. This work is a good starting point to do a similar research about roles in the context of the cloud migration lifecycle and unfold cloud-specific roles.
- **Lack of a unified process model of the cloud migration.** Our observation from the reviewed approaches shows that they are

narrowed in focus and present heterogeneous viewpoints of the same legacy to cloud migration process while there is no established correspondence among them. Furthermore, we observed that the approaches are often combined with technical-centric concepts which are often not homogenous and sometimes limited to certain cloud-specific platforms. Irrespective of technical aspects of cloud migration, the question here is that how would developers grasp a quick and overarching view of what the cloud migration process entails? With many advantages that variety of cloud migration approaches offers, it would be also beneficial to have an overarching view of the legacy to cloud migration process. Integrating existing approaches into a unified and well-abstract cloud migration process model would be advantageous in terms of facilitating understanding of the legacy to cloud migration process and lucid knowledge transfer and interoperability across the community of cloud researchers and practitioners. The need for unified migration model models has been corroborated by researchers (Zimmermann et al., 2012; Hamdaqa and Tahvildari, 2012) but still no work with this focus exists in the literature. The reference process models for the cloud migration are still a gap in the literature and worthwhile pursuit.

- **Other problems.** Beyond the abovementioned problems, the criteria traceability, scalability, and formality have been weakly supported by the existing approaches. Furthermore, automation support in performing migration activities has not been sufficiently supported.

7. Threat to validity

There are three limitations in the survey presented in this paper: bias in publication selection and imprecise data collection.

Firstly, we focused on studies whose main objective was to suggest an approach (e.g. methodology, process model) for the legacy to cloud migration. This resulted in the identification of 43 primary approaches in the literature. Through snowballing technique, (searching for all references contained in the studies), we found that studies do not necessarily use the terms process model and methodology consistently; and even in some cases, they combined these with other objectives. For example, [S35] was a well-cited source and included very useful practice and considerations for the legacy to cloud migration; however, this study was not identified through our initial search strings. To reduce the likelihood of missing relevant papers, we broadened the search strings, as shown in Table A.1 in Appendix A, to cover more relevant papers. Additionally, we performed an extensive manual search in scientific databases and conference proceedings as enumerated in Appendix A. However, it is still possible that some related work may have been missed.

Secondly, although we have used the evaluation framework as a lens to extract data items from the approach, still we believe our data extraction can be imprecise since approaches do not clearly describe their suggested migration model. We frequently found that research methodology of approaches including validation techniques, contextual information, and data analysis had not been properly explained. This could affect the quality of our data extraction.

Thirdly, the evaluation results reported in this survey have been based on the available documents on the approaches. A more realistic evaluation could be performed through applying approaches in real-world migration scenarios. However, such an evaluation is out of the scope of this paper.

Finally, we acknowledge that despite conducting a rigorous review procedure for the ratings, still the rates rely on a subjective interpretation of the authors, hence the risk of bias is probable due to the interpretive and subjective nature of any survey paper. To

alleviate this threat, it should be noted that the rating was not performed in a single step; rather, it was made after a few refinements. More specifically, each identified paper was read and immersed within the mind of the researcher of the current survey, as recommended by Braun and Clarke (2006) and then compared with other identified papers. This helped us to get familiar with the depth of each study. In addition, in order to rate the approaches in a precise manner, we emphasised a proper extraction of all relevant constructs (e.g. phases, activities, work-products) from the identified 43 studies as a basis for the rating and with respect to the defined criteria. For the extraction of the constructs, the authors were actively involved for the identification of the constructs.

8. Conclusion

We presented a systematic literature review on the legacy to the cloud migration from the perspective of the process model. As far as RQ1 is concerned, we reviewed, evaluated, and characterised existing proposed approaches suggesting a methodological solution for moving legacies to cloud environments. This was carried out via conducting an exhaustive systematic literature review including manual and automatic search on major electronic scientific databases since 2007, screening many papers and eventually leading to 43 papers related to the migration legacy applications to the cloud. An evaluation framework was suggested to highlight important features, activities, and recommendations of the existing approaches. A particular strength of the framework is its potential application as a yardstick in selecting approaches or a subset of them as to satisfy specific requirements of a given cloud migration scenario at hand.

Regarding RQ2, several few research opportunities discussed in this paper. Firstly, we found that researchers have not employed significant research approaches such as interviews, focus groups, observations, surveys, design efforts, and archival materials so as design and evaluation of suggested migration approaches. While the cumulative learning and understanding of the cloud migration can be attained via sound research methodologies, this essence has been less incorporated in the developing of current approaches. Secondly, a central aspect of the cloud migration is that a fixed migration approach is not applicable for all given migration scenarios. Nevertheless, the literature review revealed that little work exists that provides a mean to design situation-specific approaches with respect to the characteristics of a migration project. In addressing this gap, we suggested applying the situational method engineering which is to identify method fragments from existing approaches and combine them to design bespoke migration approaches fit migration scenarios. Thirdly, existing approaches need to be empowered to with relevant method fragments to the aspects multi-tenancy, elasticity, test, and continuous integration as they have been weakly supported. Fourthly, the definition of new roles specific to cloud application development has not been explored in the existing approaches. As mentioned in Section 5.3.4, a methodology should specify producer roles that are required to carry out the migration process. Fifthly, recognising that there is a sheer volume of cloud migration research, which is currently dispersed and fragmented, we suggested a need for a generic reference process model aiming at integrating the existing literature. The fact that each year a considerable number of research papers are published in the cloud computing field, where each reports different solutions, experience reports, and recommendations to move legacy assets to cloud environments, itself is an evidence that the field has reached a maturity point where the development of such a generic reference model is mandatory. Sixthly, some criteria such as traceability, scalability, formality, and automation have been not been properly supported by the existing approaches.

The current state definitely calls for further enhancement of the cloud migration with more quality approaches.

Finally, the results of analysing existing approaches have made a rich inventory of important activities, recommendations, and concerns that are commonly involved in a typical cloud migration process in one place as described Sections 5.3 and 5.4. In our view, this inventory is a great contribution of this work which can be used by academia and practitioners to understand the essence of the cloud migration process. In addition, the results of the evaluation existing approaches and the proposed evaluation framework is helpful for practitioners to get an understanding of the applicability of each individual approach and on the other hand comparing the approaches to select one or a subset of them with respect to their migration project requirements. In this regard, the survey presents a basis for a well-informed decision. Another key contribution of this survey is to enable people in the cloud computing community to get an overall view of the current state of research on the methodological aspect of legacy application migration to the cloud including key concerns, activities, and criteria need to be taken into account. It is enjoyed by a novice who will engage in the cloud migration research and anyone who is interested in the methodological aspect of moving legacies to cloud environments. We hope that they expand cloud computing body of knowledge by addressing the identified challenges.

Acknowledgements

The authors would like to thank Professor Fethi Rabhi from the School of Computer Science and Engineering at UNSW for his constructive comments and suggestions for improving this paper. The authors also wish to acknowledge comments made by anonymous reviewers that greatly strengthened the manuscript.

Appendix A

Conducting literature review

Planning review

Step 1 pilot review and defining search strings. From the first pilot review, we discerned that authors do not necessarily use the terms process models, life cycle, or methodology to name their proposal. This was why our initial search strings missed some well-known paper related to cloud migration. To alleviate this issue, the search strings were refined based on the recommended guidelines described in Dieste and Padua (2007) in order to identify all relevant studies even though they could not be labelled as a pure software development methodology. The five steps were followed: (1) Defining main terms by decomposing the research questions; (2) Identifying alternative synonyms for the main terms; (3) Checking the search strings in any relevant papers that retrieved; (4) Incorporating alternative synonyms using the logical operator ‘OR’; and (5) Using the logical operator ‘AND’ to link the main terms. The terms “Cloud”, “Cloud Computing”, “Service Computing”, “Legacy”,

“Methodology”, “Process Model”, “Reference Model”, “Migration”, and “Framework” were set as the main keywords and based upon them, the different search strings were defined using the logical operator OR to include synonyms for each search string as well as the logical operator AND to link together each set of synonyms. Using the appropriate Boolean expressions, a set of search strings were generated as shown in Table A.1.

Step 2 selecting study sources. The following databases were searched against the predefined search strings: IEEE Explore, ACM Digital Library, SpringerLink, ScienceDirect, Wiley InterScience, ISI Web of Knowledge, and Google scholar. These databases cover the vast majority of published studies in the software engineering field. In addition, journals, conference, workshop proceedings and technical reports attributed to cloud computing and SOA areas were sought. These included IEEE Transaction of Cloud Computing, IEEE Transaction of Service Computing, Software Engineering for Cloud Computing, Cloud Computing International Conference, Cloud and Service Computing International Conference, International Conference of Service-Oriented Computing, Maintenance and Evolution of Service-Oriented and Cloud-Based Systems, International Conference on Cloud, Service-Oriented Computing and Applications, and International Conference on High Performance Computing and Communications.

Step 3 defining study inclusion and exclusion criteria. The criteria for selecting a study were as follows:

- Relevant to the first research question (RQ1) though it could be titled under different terms,
- Focused on the migration of legacy applications to cloud environments, and directly dealt with the challenges as stated in Section 2.2,
- Published between 2007 (the date of origination of cloud computing) and June 2015.

This survey focused on studies pertaining to approaches for moving legacy applications to cloud environments through proposing a process model or framework. The following exclusion criteria were set for:

- High-level frameworks suggesting conceptual models for outsourcing business processes to cloud environments and do not directly deal with legacy application migration were considered out the scope of this research. Consequently, these identified frameworks were excluded: Business Model Framework (CBMF), Linthicum Cloud Computing Framework (LCCF), Oracle Consulting Cloud Computing Services Framework, IBM Framework for Cloud adoption (IFCA), BlueSky Cloud Framework for e-Learning, and Hybrid ITIL V3 Framework for Cloud.
- Decision making frameworks aiding users to rank, evaluation, and selection cloud providers that fit legacy application requirements.
- Studies in languages other than English were excluded from the review.

Table A.1
List of search strings.

Search query (SQ)
SQ1: “Migration” OR “Cloud adoption” OR “Cloud migration” OR “migration to Cloud” OR “Legacy to Cloud migration” OR “Legacy migration to Cloud” AND [SQ2 OR SQ3]
SQ2: “Monolith application” OR “Legacy code” OR “Legacy system” OR “Legacy information systems” OR “Existing system” OR “Legacy component” OR “Legacy software” OR “Legacy application” OR “On-premise application” OR “Monolithic system” OR “Existing software” OR “Pre-existing software” OR “Legacy information system” OR “Legacy program” OR “Pre-existing assets” OR “Legacy architecture” OR “Legacy asset”
SQ3: “Methodology” OR “Software Development Methodology” OR “Development Process” OR “Process Model” OR “Reference Model” OR “Migration” OR “Framework”

Table A.2
Frequency of papers by year.

Year	N	%
2009	3	6.9
2010	3	6.9
2011	6	13.9
2012	8	18.6
2013	15	34.8
2014	7	16.2
2015	1	2.3
Total	43	~100

Conducting review

Step 1 study selection. This step dealt with the seeking of the defined search strings (Step 1 of Planning) over scientific databases (Step 2 of Planning). A set of primary papers was identified as the

result of the first iteration. Then the title, abstraction, and in the most cases, the content of each paper were scrutinised regarding the inclusion and exclusion criteria, as defined in Step 3 of Planning. Reference snowballing was conducted in the sense that studies were cited in the references or related work section of a paper, were fed into the conducting review process as new resources. In some cases, for example, study [S7] in Table Appendix B, that a paper was not online accessible, its main author was communicated in order to receive the paper. In the case of multiple publications from an author or a research community, the most recent and completed one was chosen for the review. Consequently, 43 papers were identified, as the output of this step, for the review after applying the inclusion and exclusion criteria.

Step 2 data extraction from the identified studies. Finally, the proposed criteria in the evaluation framework were used as a lens to extract key data from the reminder identified papers. These data items enabled us to capture the full details of the stud-

Table A.3
Distribution by publication channel.

Channel name	Publisher name	N
Book Chapter		
International Conference on Cloud Computing and Services Science	Springer	1
Computer Communications and Networks	Springer	1
Cloud Computing	Springer	1
	Total:3 %6.1	
Journal		
Software Practice and Experience	Wiley Online Library	3
Cloud Computing with e-Science Applications	CRC Press/Taylor and Francis	1
International Journal of Big Data Intelligence	Perpetual Innovation Media	1
International Journal of Cooperative Information Systems	World Scientific	1
Journal of Software Maintenance and Evolution	Wiley InterScience	1
Journal of Systems and Software	ScienceDirect	1
International Journal of Cloud Computing and Services Science	Institute of Advanced Engineering and Science	1
International Journal of Mechatronics, Electrical and Computer Technology	–	1
IEEE Computer	IEEE	1
Computing	Springer	1
International Journal on Advances in Software	GEOMAR	1
IEEE Transaction on Cloud Computing	IEEE	1
Information Technology	De Gruyter	1
	Total:15 %30.6	
Conference		
Services	IEEE	1
Service-I	IEEE	1
International Conference on Cloud Computing and Service Computing	IEEE	1
International Conference on Internet and Web Applications and Services	ThinkMind	1
International Conference on e-Business Engineering	IEEE	1
International Conference on Object Oriented Programming Systems Languages and Applications	Unipub	1
International Conference on Cloud Computing	IEEE	1
International Symposium on Service-Oriented System Engineering	IEEE	1
International Conference on Service-Oriented and Cloud Computing	Springer	1
International Service-Oriented Computing and Applications	IEEE	1
International Conference on Cloud Engineering	IEEE	1
World Congress on Services	IEEE	1
International Conference on Computer Software and Applications	IEEE	1
International Conference on High Performance Computing	IEEE	1
International Conference on Cloud Computing	IEEE	1
International Conference on Cloud Computing, GRIDs, and Virtualization	XPS	1
International Working Conference on Software Architecture	ACM	1
Symposium on Symbolic and Numeric Algorithms for Scientific Computing	IEEE	1
IEEE International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems	IEEE	1
	Total:19 %38.8	
Workshop		
International Workshop on Software Engineering for Cloud Computing	ACM	1
International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems	IEEE	1
International Workshop on Modeling in Software Engineering	IEEE	1
International Workshop on Service-Oriented Application	Springer	1
	Total:4 %8.2	
White Paper		
Amazon, Cloud Standards Customer Council,	–	2
	Total:5 %10.2	
Total all publication type: 43		

Table A.4
Frequency of papers by year.

Continent	Country	N	%
Europe	Germany	9	18.4
	Ireland	5	10.2
	UK	2	4.1
	Spain	2	4.1
	Denmark	2	4.1
	Italy	1	2
	Belgium	1	2
	Bulgaria	1	2
	Portugal	1	2
	Norway	1	2
	Greece	1	2
	Sweden	1	2
	Finland	1	2
	Total	28	56.9
North and South America	USA	6	12.2
	Brazil	1	2
	Canada	1	2
	Total	8	16.2
Asia	China	2	4.1
	India	2	4.1
	United Arab Emirate	1	2
	Korea	1	2
	Iran	1	2
	Total	7	14.2
Australia	Australia	2	4.1
	Total	2	4.1

ies under review. The extracted data from the studies and their analysis are presented in Sections 5.3 and 5.4. Apart from the data extraction using the framework, data items pertaining to research quality were extracted for further assessment as discussed in Section 5.1. The criteria were borrowed from Critical Appraisal Skills Programme (Greenhalgh and Taylor, 1997) and those suggested conducting empirical research in SE by Kitchenham et al. (2002). Appendix C presents these criteria.

Overview of approaches

Distribution by the year of publication. Table A.2 presents the frequency of the proposed approaches since 2009. N shows the

number of publications per year. For example, until 2009 there have been 4 publications. The highest publication rate is in 2013 with 17 papers (34.7%). As shown in this table there has been a continuous growth in publications from 2009 to 2013. The number of publications has decreased in years 2014 with 7 papers.

Distribution by publication type. Table A.3 presents a classification of publications from academia and industrial sectors. It indicates that the most of the contributions are related to conferences with 19 out of all 43 publications. Journals are the second-ranked publication channel with 15 publications in total. Among them, the journal of *Software Practice and Experience* published the most papers related to cloud migration approaches with 3 papers. White papers, workshops papers, book chapters, and dissertations are placed as third, fourth, and fifth regarding the number and percentage of the total publications, respectively.

Distribution by authors' nationality. Table A.4 shows the geographical distribution of the identified papers. The value N in this table indicates the total number of time authors from a country published a paper on the topic of the cloud migration. According to this table, German stands at the first place for publication in this field with proposing 9 approaches (18.4% of total publication). USA and Ireland stand in the second and third places with 6 and 5 publications, respectively. UK, Spain, and Denmark are with 2 published papers, following with Italy, Belgium, Bulgaria, Portugal, Norway, Greece, Sweden, and Finland with 1 published paper. An ascending ordering of publications based on the continent reveals the cloud migration approaches in Europe with N=28, North and South America with N=8, Asia with N=7, and Australia with N=2. There were not any contributions from Africa in this survey. It should be noted that Cisco, IBM, Logicalis, Cloud Standards Customer Council, and Remics consortium were not considered in the country-based classification as they come from industry and dispersed over the world.

Appendix B

This appendix presents the list of identified approaches from the literature for the purpose of this survey.

Table B.1.

Table B.1
Studies included in final review.

Study ID	Authors and title	Abbreviation	Channel	Source	Year	Affiliation
[S1]	Krasteva, I., S. Stavru, et al., "Agile Model-Driven Modernization to the Service Cloud"	ICIW	Conference	ThinkMind	2013	Rila Solutions EAD, Bulgaria
[S2]	Beserra, P. V., A. Camara, et al., "Cloudstep: A step-by-step decision process to support legacy application migration to the cloud"	MESOCA	Workshop	IEEE	2012	Brazil
[S3]	Anstett, T., Leymann, F., Mietzner, R., "Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes"	Services - I	Conference	IEEE	2009	Institute of Architecture of Application Systems, University of Stuttgart, Germany
[S4]	Conway, G. and E. Curry, "The IVI Cloud Computing Life Cycle"	CCSS	Book Chapter	Springer	2013	Innovation Value Institute, National University of Ireland
[S5]	Tran, V., J. Keung, et al., "Application migration to cloud: a taxonomy of critical factors"	SECLLOUD	Workshop	ACM	2011	CSE, University of New South Wales, Australia
[S6]	Khajeh-Hosseini, A., D. Greenwood, et al., "The cloud adoption toolkit: supporting cloud adoption decisions in the enterprise"	SPE	Journal	Wiley Online Library	2012	Cloud Computing Co-laboratory, School of Computer Science University of St Andrews, UK

(continued on next page)

Table B.1 (continued)

Study ID	Authors and title	Abbreviation	Channel	Source	Year	Affiliation
[S7]	Binz, T., Breitenbücher, U., "Migration of enterprise applications to the cloud"	–	Journal	Information Technology	2014	University of Stuttgart
[S8]	Chauhan, M. A. and M. A. Babar, "Towards Process Support for Migrating Applications to Cloud Computing"	CSC	Conference	IEEE	2012	Software and Systems Group IT University of Copenhagen, Denmark
[S9]	Strauch, S., et al. "Migrating eScience Applications to the Cloud: Methodology and Evaluation"	–	Journal	CRC Press / Taylor and Francis	2014	Institute of Architecture of Application Systems, University of Stuttgart, Germany
[S10]	S. Strauch, V. A., D. Karastoyanova, F. Leymann, "Migrating Enterprise Applications to the Cloud: Methodology and Evaluation"	JBDI	Journal	Perpetual Innovation Media	2014	Institute of Architecture of Application Systems, University of Stuttgart, Germany
[S11]	Leymann, F., et al., "Moving applications to the cloud: An approach based on application model enrichment"	JCIS	Journal	World Scientific	2011	Institute of Architecture of Application Systems, University of Stuttgart, Germany
[S12]	Zhang, W., et al., "Migrating legacy applications to the service Cloud"	OOPSLA	Conference	Unipub	2009	SINTEF, Norway
[S13]	La and Kim 2009, "A systematic process for developing high quality saas cloud services"	–	Book Chapter	Springer	2009	Department of Computer Science Soongsil University, Korea
[S14]	Miranda, J., et al., "Assisting Cloud Service Migration Using Software Adaptation Techniques"	Cloud Computing	Conference	IEEE	2013	Dept. of Information Technology and Telematic Systems Engineering, University of Extremadura, Cáceres, Spain
[S15]	Guillén, J., et al., "A service-oriented framework for developing cross cloud migratable software"	JSS	Journal	ScienceDirect	2013	GloIn, Calle Azorín 2, Cáceres, Spain
[S16]	SUN, K., Li, Y., "Effort Estimation in Cloud Migration Process"	SOSE	Conference	IEEE	2012	IBM Research - China
[S17]	Rabetski, P. and G. Schneider, "Migration of an On-Premise Application to the Cloud: Experience Report"	ESOC	Conference	Springer	2013	Department of Computer Science and Engineering Chalmers University of Technology, and the University of Gothenburg Gothenburg, Sweden
[S18]	C., Pahl, H. Xiong, et al., "A Comparison of On-Premise to Cloud Migration Approaches"	SOCC	Conference	Springer	2013	IC4, Dublin City University, Dublin, Ireland
[S19]	Laszewski, T. and P. Nauduri, "Migrating to the Cloud: Oracle Client/Server Modernization"	Book	–	Elsevier	2012	USA
[S20]	Jamshidi, P., Pahl, C., "Cloud Migration Patterns: A Multi-Cloud Architectural Perspective"	WESOA	Workshop	Springer	2014	IC4 – the Irish Centre for Cloud Computing and Commerce, Dublin City University Dublin, Ireland
[S21]	Ahmad, Aakash, and Muhammad Ali Babar., "A framework for architecture-driven migration of legacy systems to cloud-enabled software"	WICSA	Conference	ACM	2014	IT University of Copenhagen, Denmark
[S22]	Ridha, Gadhagadhi, Khazri Saida, and Cheriet Mohamed. "OPENICRA: Towards A Generic Model for Automatic Deployment of Applications in the Cloud Computing"	IJ-CLOSER	Journal	Institute of Advanced Engineering and Science	2013	Multimedia Communication in Telepresence, Montréal (QC), Canada
[S23]	Council, C. S. C., "Migration applications to public Cloud Services: roadmap for success"	–	–	Cloud Standards Customer Council	2013	Cloud Standards Customer Council
[S24]	Ahmad Jalili, and Omid Bushehrian, "A Structured and Achromatic Process for Legacy to Cloud Migration: A Survey"	IJMEC	Journal	–	2014	Department of Computer Engineering and Information Technology, Shiraz University of Technology, Shiraz, Iran
[S25]	Quang Hieu, V. and R. Asal, "Legacy Application Migration to the Cloud: Practicability and Methodology"	SERVICES	World Congress	IEEE	2012	Khalifa University, UAE, United Arab Emirates
[S26]	Benguria, G., Elvesæter, B., Ilieva, S., "REuse and Migration of legacy applications to Interoperable Cloud Services"	–	REMICS	REMICS Consortium, Research Report	2013	SINTEF ICT

(continued on next page)

Table B.1 (continued)

Study ID	Authors and title	Abbreviation	Channel	Source	Year	Affiliation
[S27]	S., Rajaraajeswari, R., Pethuru, "Cloud Application Modernization and Migration Methodology"	–	Book Chapter	Springer	2013	Department of Master of Computer Applications, India
[S28]	Tang,K., Zhang J.M, Feng, C.H, "Application Centric Lifecycle Framework in Cloud"	E-Business Engineering	Conference	IEEE	2011	IBM Research China
[S29]	Maenhaut, p., Moens, H., Ongenae, V., Turck, F "Migrating legacy software to the cloud: approach and verification by means of two medical software use cases"	SPE	Journal	Wiley Online Library	2015	Ghent University, Belgium
[S30]	Binz, Leymann et al., "CMotion: A framework for migration of applications into and between clouds"	SOCA	Conference	IEEE	2011	Institute of Architecture of Application Systems University of Stuttgart, Germany
[S31]	Jamshidi, P., Ahmad, A., Pahl, C., "Cloud Migration Research: A Systematic Review"	TCC	Journal	IEEE	2013	School of Computing, Dublin City University, Ireland
[S32]	Ardagna, D., Casale., G., "MODACLOUDS: A Model-Driven Approach for the Design and Execution of Applications on Multiple Clouds"	MiSE	Conference	IEEE	2012	Politecnico di Milano
[S33]	Wu, J., et al., "Migrating a Digital Library to a Private Cloud"	ICE	Conference	-	2014	Information Sciences and Technology Computer Science and Engineering Pennsylvania State University, USA
[S34]	Pahl, C. and H. Xiong, "Migration to PaaS clouds-Migration process and architectural concerns"	MESOCA	Conference	IEEE	2013	the Irish Centre for Cloud Computing and Commerce, Ireland
[S35]	Andrikopoulos, V., Binz, T., Leymann, F., Strauch, S., "How to Adapt Applications for the Cloud Environment"	Computing	Journal	Springer	2013	Institute of Architecture of Application Systems, University of Stuttgart, Germany
[S36]	Bahga, A., Madiseti, V.K., "Rapid Prototyping of Multitier Cloud-Based Services and Systems"	IEEE Computer	Journal	IEEE	2013	Georgia Tech, USA
[S37]	C. Ward, N. Aravamudan, "Workload Migration into Clouds – Challenges, Experiences, Opportunities"	Cloud	Conference	IEEE	2010	IBM
[S38]	Lindner, M.A., McDonald, F., Conway, G., Curry E., "Understanding Cloud Requirements - A Supply Chain Lifecycle Approach"	-	Conference	XPS	2011	SAP Research, Palo Alto, USA
[S39]	Zhou, H., Yang, H.,Hugill, A., "An Ontology-Based Approach to Reengineering Enterprise Software for Cloud Computing"	COMPSAC	Conference	IEEE	2010	Software Technology Research Laboratory Leicester, UK
[S40]	Varia, J., "Migrating Your Existing Application to the AWS Cloud. A Phase-Driven Approach to Cloud Migration"	-	White paper	Amazon	2010	Amazon, USA
[S41]	Frey, S. and W. Hasselbring, "The cloudmig approach: Model-based migration of software systems to cloud-optimized applications"	-	Journal	GEOMAR	2011	Software Engineering Group University of Kiel, Germany
[S42]	Banerjee, J., "Moving to the cloud: Workload migration techniques and approaches"	HiPC	Conference	IEEE	2012	IBM Kolkata, India
[S43]	Menychtas, A., Santzaridou,C., Koussouris, G., Varvarigou, T., "ARTIST Methodology and Framework: A novel approach for the migration of legacy software on the Cloud"	SYNASC	Conference	IEEE	2013	Athens, Greece

Appendix C

Table C.1

Table C.1

Research quality criteria (adapted from Greenhalgh and Taylor (1997) and Kitchenham et al. (2002)).

Criteria name	Definition	Criterion type	Possible values
Research aim	Is there a clear statement of research objective?	Scale form	The research provides a clear statement of the research objective. The research provides a partial statement of the research objective. The research does not provide a statement of the research objective.
Research context	Is there an adequate description of the context in which the research was conducted?	Scale form	The research clearly describes the research context. The research partially describes the research context. The research does not describe the research objective.
Research design	Is there any appropriate research design to conduct the research?	Scale form	The research describes the research design. The research provides a partial description of the research design. The research does not describe the research design.
Data collection	Is there any clear description of data collection method to address research question?	Scale form	The research provides a clear description of data collection method. The research provides a partial description of the data collection method. The research does not provide a description of the data collection method.
Data analysis	Is there a clear and rigors description of data analysis?	Scale form	The research clearly describes the data analysis. The research partially describes the data analysis. The research does not describe the data analysis.
Reflexivity	Is there an appropriate relationship between research and participants to conduct research?	Scale form	The research clearly describes the data analysis. The research partially describes the data analysis. The research does not describe the data analysis.
Finding	Is there a clear description of research findings?	Scale form	The research provides a clear description of the research findings. The research provides a partial description of the research findings. The research does not provide a description of the research finding.
Value	Is there a clear description of contributions to research and practice?	Scale form	The research provides a clear description of the research contribution. The research provides a partial description of the research contribution. The research does not provide a description of the research contribution.
Validation type	Does a validation of the suggested methodology exist?	Descriptive	–

Appendix D

Table D.1.

Table D.1

Description of included cloud migration approaches in final review.

Year	Study ID	Aim
2009	[S12]	The methodology proposes seven generic activities to aid developers to move monolithic legacy applications to SaaS.
2009	[S13]	As conventional approaches such as object-oriented methodologies lack effectively support SaaS-specific engineering activities such as modelling common features, variability, and designing quality services, this paper presents a systematic process model for developing SaaS applications. It uses product-line engineering notions such as commonality and variability (C&V) modelling to maximise the reusability
2009	[S3]	This paper investigates how legacy business process models can be executed in the cloud using service delivery models such as IaaS, PaaS, and SaaS. It also describes modifications are required to execute BPEL processes in the cloud. The focus is on moving the business logic of legacy assets to the cloud whilst data are kept in the local data centre.
2010	[S39]	This paper proposes a five-step ontology development process to reengineer legacy enterprise applications for shifting to the cloud by building an ontology for enterprise software and then partitioning and decomposing it into potential service candidates. The approach helps to understand legacy software and identify potential service candidates that can meet the objectives of the cloud migration.
2010	[S40]	This paper suggests a phase-driven step-by-step methodology for moving enterprise applications to Amazon cloud. The methodology presents several scenarios of deploying applications in Amazon.
2010	[S37]	This paper proposes a framework, called Darwin, with an emphasis on accelerating heterogeneous source/target migrations to standard virtualised environments. The model helps for smooth migration of the legacy workload from the local environment to a new cloud environment in a cost effective way, with minimal disruption and risk.
2011	[S5]	The authors have a cost-oriented view to the migration process and based on this propose a taxonomy of the migration activities. They show a breakdown of costs and factors that impact on activities in a case-study in a .NET n-tier application which is migrated to run on Windows Azure. This taxonomy and cost factors contributed towards a cost estimation model of the cloud migration.
2011	[S11]	A methodology and corresponding tool chains to rearrange the legacy application components and classify them into groups of components so that each group can be provisioned separately to different clouds while preserving the desired properties of the whole application. The methodology has been described in terms of the various activities to be performed and artefacts to be created in order to move an application to the cloud.
2011	[S28]	An application centric lifecycle management framework to deploy a complex application in the cloud. This framework exploits a model-driven approach which captures information in models that can be used to automatically generate code and configuration.
2011	[S30]	This paper proposes a framework called CMotion to address the vendor lock-in problem of application migration into and between different clouds. It uses adaptors to enable previously incompatible technologies to work together. Proposing various adaptation techniques is the main idea behind this framework.
2011	[S38]	The authors define a cloud lifecycle process from the perspective of supply chain management. It demonstrates how the supply chain plays an active role to manage the process for migrating from an existing computing environment to a cloud hosting environment.
2011	[S41]	A model-driven methodology including a set of reengineering activities in order to translate legacy application architecture to a target architecture for IaaS and PaaS-based cloud environments.
2012	[S2]	This paper presents Cloudstep, a step-by-step process from a decision making viewpoint. It helps developers not only in selecting the cloud models and services best suited for their application, but also in carefully assessing the various risks and benefits occur during the migration process.
2012	[S6]	A cloud adoption conceptual framework is proposed to support decision makers in identifying uncertainty to deploy legacy applications in the cloud and analysing the cost of deployment options which can affect bandwidth and the cloud service provider's pricing scheme.
2012	[S8]	This paper proposes a process model on the basis of gained experience in moving an Open Source System (OSS), Hackstat, to two different cloud computing platforms. It provides guidelines, observations, and lessons learned regarding key issues involved in the migration such as analysis of target cloud environments against specific application's requirements, evaluation of platforms for cloud specific quality attributes of SaaS applications, the impact of a target cloud platform on each of the migration activities, and the potential influence of different services offered by cloud environments on migration activities and on the new architecture of a system to be migrated.
2012	[S16]	A process-based effort estimation approach is presented to assess the investment of the cloud migration process before it is undertaken with a focus on infrastructure-level migration.
2012	[S19]	The waterfall life cycle model has been used as a base to define a new methodology, concerning with activities such assessment, selecting service provider, migration effort estimation, training requirements, IT resource requirement for the migration project, capturing exhaustive amounts of information from the legacy systems, and database schema layout, testing, optimisation, and deployment.
2012	[S24]	A generic process model for the cloud migration with a focus on tools and techniques which can be applied in every step.
2012	[S25]	A methodology to assess the feasibility of migration to PaaS along with compatibility checklist to estimate the cost of migration as well as general solutions to address incompatibilities.
2012	[S42]	This paper offers a five-step migration methodology without re-architecting or re-engineering the existing applications. It also includes a set of high-level migration patterns emerged from the commonly found repeatable migration scenarios.
2012	[S32]	A model-driven approach with a focus on the interoperability of applications in multiple cloud environments. In this approach, application models are designed at a high level to abstract from the targeted cloud, and then semi-automatically translated into to multi-cloud platforms.
2013	[S1]	This paper presents how the model-driven modernisation can be enriched with Agile software development practices for a seamless execution of different migration activities.
2013	[S4]	A nine-step cloud life cycle which is concerned with challenges such as security, data ownership, interoperability, service maturity, and return on investment. It includes the recommended activities to address these concerns.
2013	[S14]	An adaptation process to resolve incompatibility and vendor lock-in issues between legacy applications and cloud services. The approach promotes the use of decoupling-aware mechanisms in the design of cloud applications.
2013	[S15]	An approach for developing cloud agnostic applications that may be deployed across multiple cloud platforms. In this approach information about cloud deployment and cloud integration is separated from the source code and managed by the framework. Interoperability between interdependent components deployed in different clouds is achieved by automatically generating services and service clients from legacy models.

(continued on next page)

Table D.1 (continued)

Year	Study ID	Aim
2013	[S17]	This paper presents an experience of the migration process to Microsoft Azure platform in the context of a small company. It recommends that activities and consideration should be taken into account in different migration scenarios, in particular from a performance and cost perspective.
2013	[S18]	The aim of the paper is to understand the core elements of cloud migration processes from an architecture perspective. Authors extract commonalities and difference between migration process activities for migration to IaaS, SaaS, and PaaS and present a range of concerns such architecture settings, costs, skills, and technologies.
2013	[S22]	This paper focuses on the design and the implementation of a new generic model for automatic application deployment, called OpenICRA. It aims to reduce application development complexity and to simplify cloud services deployment process.
2013	[S23]	The methodology provides a series of activities that developers should take into account to ensure successful migration of existing applications to IaaS. It details activities related to assessing applications and workloads, developing a business case and technical approach, addressing security and privacy requirements as well as managing the migration process.
2013	[S34]	A generic architecture-centric process for moving to PaaS. The focus is on programming and architecture design principles that need to be incorporated in a migration process to PaaS. Experts' opinions on two case studies have been used to formulate this process model.
2013	[S36]	A three-phased architecture-centric development process for moving (or developing) multitier applications to the cloud with a focus on architecture design principles such as scalability, performance, maintainability, portability, and interoperability.
2013	[S43]	The methodology is a model-driven migration methodology and considers both the technical and business aspects of the legacy applications. The methodology emphasises generating reusable artefacts from legacy models which can be transformed to multiple cloud platforms.
2013	[S35]	This paper categorises different migration types and identifies the potential impact and adaptation needs for each of these types on application tiers. It also investigates various cross-cutting concerns such as security that need to be considered for the migration of the application, and position them with respect to the identified migration types.
2013	[S27]	A five-step process to discover the optimal balance of performance, agility, sustainability, and cost.
2013	[S31]	Through using a systematic literature review, this paper introduces a conceptual process model called Cloud-RMM (Cloud - Reference Migration Model), which classifies key process areas related to the cloud migration.
2013	[S26]	The main objective of the REMICS (REuse and Migration of legacy applications to Interoperable Cloud Services) project is to specify, develop, and evaluate a tool-supported model-driven methodology for migrating legacy applications to interoperable service cloud platforms.
2014	[S33]	This paper presents activities that authors carried out to move CiteSeerX into a private cloud. It also reports a number of lessons learned from this migration experience, challenges encountered prior to and during the migration and post-migration issues and possible solutions.
2014	[S9]	This methodology consists of seven phases for moving the database tier of e-science applications to the cloud. It incorporates guidance on choosing a cloud database solution and required refactoring activities.
2014	[S10]	A step-by-step methodology for the migration of a legacy data tier to the cloud and the refactoring of the application architecture.
2014	[S21]	An architecture-centric process, named Legacy-to-Cloud Migration Horseshoe, which views the migration as a set of recurring problems that can be formulated as migration process patterns.
2014	[S20]	This paper proposes 15 fine-grained patterns of the migration that target multi-cloud settings and are specified with architectural notations.
2014	[S7]	This approach defines activities for modeling an overall picture of enterprise applications and deploying them in the cloud.
2015	[S29]	A generic model to migrate to a hybrid or public cloud environment, and the steps required to add multi-tenancy to legacies.

Appendix E

Table E.1.

Table E1
Result of evaluation of approaches based on the generic criteria Process Clarity, Procedure and Technique, Modelling, Tailorability, and Tool Support.

Study	Process Clarity	Procedures & Techniques	Modelling	Tailorability	Tool Support
[S1]					
[S2]					
[S3]					
[S4]					
[S5]					
[S6]					
[S7]					
[S8]					
[S9]					
[S10]					
[S11]					
[S12]					
[S13]					
[S14]					
[S15]					
[S16]					
[S17]					
[S18]					
[S19]					
[S20]					
[S21]					
[S22]					
[S23]					
[S24]					
[S25]					
[S26]					
[S27]					
[S28]					
[S29]					
[S30]					
[S31]					
[S32]					
[S33]					
[S34]					
[S35]					
[S36]					
[S37]					
[S38]					
[S39]					
[S40]					
[S41]					
[S42]					
[S43]					
Fully-Supported	35 (81%)	23 (53%)	2 (5%)	0	2 (5%)
Partially-Supported	7 (16%)	14 (32%)	12 (28%)	5 (11%)	7 (16%)
Not-Supported	1 (2%)	6 (14%)	29 (67%)	38 (88%)	34 (79%)
Total	43	43	43	43	43

Appendix F

Table F.1.

Table F.1
Generic criteria.

Criterion	Evaluation question	Criterion type	Description	Source
Process clarity	Does the approach provide a clear description of the suggested phases and activities?	Scale	<p>● The approach explicitly provides a clear description of activities for conducting the migration process.</p> <p>◐ The approach provides a general description for some activities but details are lacking.</p> <p>○ The approach provides either very partial definition or any definition for the activities.</p>	(Ramsin and Paige, 2008)
Procedures and supportive techniques	<p>(i) What are the techniques to perform each activity?</p> <p>(ii) Are examples and heuristics of the activities provided?</p>	Scale	<p>● The approach offers techniques/examples for activities.</p> <p>◐ The approach offers techniques/examples for some activities.</p> <p>○ The approach does not provide supportive techniques or examples for activities.</p>	(Karam and Casselman, 1993)
Modeling language	Does the approach specify a modelling or notational component to represent produced work-products during the migration process?	Scale	<p>● An existing modelling language or a new one is prescribed for all of the activities.</p> <p>○ An existing modelling language or a new one is prescribed for some of the activities.</p> <p>● Any modelling language has not been specified.</p>	(Karam and Casselman, 1993; Sturm and Shehory, 2004)
Traceability	Does the approach determine the sequence of modelling or dependencies between produced work-products?		<p>◐ The approach specifies the traceability of all work-products in the migration process.</p> <p>○ The approach specifies traceability for a subset of activities.</p> <p>● Traceability links between work-products have not been specified.</p>	(Karam and Casselman, 1993; Ramsin and Paige, 2008; Chitchyan et al., 2015)
Tailorability	<p>(i) Is the approach based on a one-size-fits-all assumption or configurable regarding a given migration scenario?</p> <p>(ii) Is the approach have been expressed in the form of modular method fragments or process components?</p>	Scale	<p>◐ The approach defines mechanisms to configure and modify its suggested process or modelling language.</p> <p>○ The approach provides a basis (e.g. repository of method fragments) so that the tailoring process is facilitated.</p> <p>● Any tailoring mechanisms is supported by the approach.</p>	(Karam and Casselman, 1993; Ramsin and Paige, 2008)
Tool support	<p>(i) Has the approach developed specific tools to perform migration process?</p> <p>(ii) Does the approach use or provide guidelines for developers to select existing third-party tools for modelling?</p>	Scale	<p>◐ The approach provides tools for the whole activities or integrated with existing third-party tools.</p> <p>○ The approach provides tools for some activities but is lacking for other.</p> <p>● The approach neither provides tool nor refers to existing available third-party tools for modelling work-products.</p>	(Karam and Casselman, 1993)

(continued on next page)

Table F.1 (continued)

Criterion	Evaluation question	Criterion type	Description	Source
Theoretical foundation	Is the approach has been inspired or developed based on the existing software engineering paradigms or practice?	Descriptive	–	(Karam and Casselman, 1993)
Work-products	What work-products are prescribed by the approach to produce in the migration process?	Descriptive	–	(Karam and Casselman, 1993; Sturm and Shehory, 2004)
Development roles	What development roles, who are responsible for performing migration activities or any stakeholders, are defined by the approach?	Descriptive	–	(Karam and Casselman, 1993; Sturm and Shehory, 2004)
Domain applicability	What are application domains for which the approach is applicable?	Descriptive	–	(Karam and Casselman, 1993)
Scalability	Does the approach or a subset of it is applicable to handle various migration sizes?	Boolean	Yes: The approach explicitly defines mechanisms to support various migration sizes and its scalability has been demonstrated in real project.	(Sturm and Shehory, 2004; Karam and Casselman, 1993; Chitchyan et al., 2015)
	Does the methodology define mechanisms or guidelines suitable to handle different migration sizes?		No: The approach does not support scalability	
	Is there any real evidence of applying the approach in the migration of legacy application with different sizes?		–	
Formality	Does the approach provide a degree of formality on technical aspects?	Boolean	Yes: The approach provides formal techniques for some activities.	(Karam and Casselman, 1993)
	Does the approach use unambiguous mathematical definitions for modelling or describing work-products?		No: Any formalisms is supported by the approach.	

Table G.1

Table G.1
Evaluation results based on the cloud-specific criteria.

[illegible]

Appendix H

Table H.1.

Table H.1

Cloud-specific criteria.

Criterion	Evaluation question	Criterion type	Possible evaluation result
Migration type	What kind of service delivery model and application tier are concerned by the approach?	Multiple	Types I, II, III, IV, V
Unit of migration	What tiers of application has the approach been designed for the migration?	Multiple	Whole Application Stack, User Interface Tier, Business Logic Tier, Data Tier
Context analysis	Does the approach define activities related to assess the suitability of migration to the cloud in terms of security, legality, cost, organizational change, and user resistance?	Scale	<p>● The approach explicitly provides a clear description of the activities for conducting a context analysis.</p> <p>◐ The approach provides a general description for context analysis activity but details are lacking.</p> <p>○ The approach provides either very partial definition or any definition for the context analysis.</p>
Legacy application understanding	<p>(i) Does the approach provide activities for recapturing the abstract representation of application architecture in terms of functionalities, data, constraints, structure of components, business and data tiers, design quality, complexity, and coupling?</p> <p>(ii) Does the approach identify and collect usage data of working legacy application and its resource utilisation model?</p>	Scale	<p>● The approach explicitly provides a clear description of the activities to discover knowledge about the legacy application.</p> <p>◐ The approach provides a general description for discovering knowledge about legacy application but details are lacking.</p> <p>○ The approach provides either very partial definition or any definitions for the legacy application understanding.</p>
Migration requirements analysis	Does the approach provide activities to acquire a set of requirements from multiple stakeholders about the target cloud-enabled application according to the current configuration setting of the legacy application? Requirements such as computational requirements, servers, data storage and security, networking and response time, and elasticity requirements	Scale	<p>● The approach defines activities related to elicitation and analysis of migration requirements.</p> <p>◐ The approach offers general guideline for requirement analysis.</p> <p>○ Migration requirement analysis is not addressed in the approach.</p>
Planning	<p>(i) Does the approach provide support for correct and safe sequence of steps which guide the rest of migration process?</p> <p>(ii) Does the approach provide guidelines for a proper roll-back plan to an in-house version of the legacy application?</p>	Scale	<p>● The approach defines activities related to migration planning.</p> <p>◐ The approach offers general guideline for planning.</p> <p>○ The approach does not support planning activity.</p>
Cloud service provider selection	<p>(i) Does the approach provide activities, guidelines, or concerns that should be taken into account to choose a cloud provider that meet requirements?</p> <p>(ii) Does the approach address licensing issues in cloud environments?</p>	Scale	<p>● The approach defines criteria or guidelines for selection cloud services are to be utilised by the legacy application.</p> <p>◐ The approach offers general guideline for cloud service selection.</p> <p>○ The approach does not support the activity of cloud provider selection.</p>
Training	Does the approach specify necessary skill or training requirements for developers and IT staff?	Scale	<p>● The approach explicitly defines an activity for developers and IT staff training.</p> <p>◐ The approach suggests considering training activities but does not define activities or guidelines for it.</p> <p>○ The approach does not incorporate training into its suggested process model.</p>
Component selection and distribution in the cloud	What are the activities or guidelines in the approach to assess and determine legacy components which are suitable to be migrated to the cloud?	Scale	<p>● The approach provides an explicit activity for selecting legacy components which are suitable for migration to the cloud and the way of arranging them in cloud platforms.</p> <p>◐ Support for component selection and distribution is implicit and confined to general advice.</p> <p>○ The approach does not support this activity.</p>

(continued on next page)

Table H.1 (continued)

Criterion	Evaluation question	Criterion type	Possible evaluation result
Incompatibility resolution (application refactoring)	(i) Does the approach identify possible incompatibilities and possible solutions to resolve these incompatibilities? (ii) What kind incompatibilities are concerned with the approach?	Scale	<p>● The approach defines activities to identify incompatibilities between legacy components and cloud services and suggests techniques to resolve them.</p> <p>◐ The approach concerns incompatibilities between legacy application and cloud services, however, does not define an explicit activity to identify and address these incompatibilities.</p> <p>○ The approach does not address incompatibilities.</p>
Enabling multi-tenancy	(i) Does the approach provide activities, techniques, or guidelines for detecting and handling faults which might incur in a tenant (tenant availability isolation)? (ii) Does the approach include activities for identifying commonality and variability in the target domain? Any specific techniques are offered for application customisability on the basis of particular (application customisability)? (iii) Does the approach provide mechanisms to protect tenants from each other in terms of access to data and security (security isolation)? (iv) Does the approach provide support to protect tenant's performance from being affected by other tenant's behaviour (tenant performance isolation)?	Scale	<p>● The approach defines activities to address multi-tenancy.</p> <p>◐ The approach concerns with multi-tenancy but there is not guideline to enable multi-tenancy in the legacy application.</p> <p>○ The approach does not address multi-tenancy.</p>
Enabling application elasticity	Is there any activity or guideline in the approach to define scaling rules, dynamic acquisition and release of cloud resources?	Scale	<p>● The approach defines activities to address elasticity.</p> <p>◐ The approach concerns with multi-tenancy but there is not guideline to enable elasticity in the legacy application.</p> <p>○ The approach does not address elasticity.</p>
Test and continuous integration	(i) What kind of test activities are defined in the approach? (ii) Is there any systematic support to apply changes to application parts which are host in the cloud?	Scale	<p>● The approach defines a detailed activity to conduct test and integration.</p> <p>◐ The approach provides general guidelines to conduct test and integration.</p> <p>○ The approach does not cover test and integration.</p>
Environment configuration	How approach provides support for re-configuring the running environment of the application including reachability policies to resources and network, connection to storages, setting ports and firewalls, and load balancer?	Scale	<p>● The approach defines a detailed activity to prepare cloud environment to deploy the application.</p> <p>◐ The approach provides general guidelines for configuring the cloud environment prior deploying application.</p> <p>○ The approach does not support environment configuration.</p>
Continuous monitoring	Does the approach provide mechanisms for continuously monitoring application components utilising cloud services e.g. CPU, Memory, Disk I/O, and Network I/O?	Scale	<p>● The approach explicitly provides an activity for continuous application monitoring when running in the cloud.</p> <p>◐ The approach provides general advice for application monitoring in the cloud.</p> <p>○ The approach does not support application monitoring.</p>

Appendix I

Table I.1.

Table I.1
Quality assessment of the studies.

Study	Research Aim	Research Context	Research Design	Data Collection	Data Analysis	Reflexivity	Findings	Value
[S1]								
[S2]								
[S3]								
[S4]								
[S5]								
[S6]								
[S7]								
[S8]								
[S9]								
[S10]								
[S11]								
[S12]								
[S13]								
[S14]								
[S15]								
[S16]								
[S17]								
[S18]								
[S19]								
[S20]								
[S21]								
[S22]								
[S23]								
[S24]								
[S25]								
[S26]								
[S27]								
[S28]								
[S29]								
[S30]								
[S31]								
[S32]								
[S33]								
[S34]								
[S35]								
[S36]								
[S37]								
[S38]								
[S39]								
[S40]								
[S41]								
[S42]								
[S43]								
Fully-Supported	39(90%)	25 (58%)	6 (14%)	7 (16%)	4 (9%)	3 (7%)	17 (39%)	18 (41%)
Partially-Supported	4 (9%)	7 (16%)	12 (28%)	6 (14%)	8 (18%)	1 (2%)	10 (23%)	8(18%)
Not-Supported	0	11 (26%)	25 (58%)	30 (70%)	31 (72%)	39 (90%)	16 (37%)	17 (40%)
Total	43	43	43	43	43	43	43	43

References

- Andrikopoulos, V., Binz, T., Leymann, F., Strauch, S., 2013. How to adapt applications for the Cloud environment. *Computing* 95, 493–535.
- Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., 2010. A view of cloud computing. *Commun. ACM* 53, 50–58.
- Avison, D., Fitzgerald, G., 2003. *Information Systems Development: Methodologies, Techniques and Tools*. McGraw Hill.
- Barbier, F., Hein, C., 2011. State of the art on modernization methodologies, methods and tools. REMICS Consortium, Research Report, March.
- Benguria, G., Elvesæter, B. & Ilieva, S. 2013. Deliverable D2. 6 REMICS Handbook, REMICS, Final Release.
- Bennett, K., 1995. Legacy systems: coping with success. *IEEE Software* 12, 19–23.
- Bezemer, C.-P., Zaidman, A., 2010. Multi-tenant SaaS applications: maintenance dream or nightmare? In: *Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE)*. ACM, pp. 88–92.
- Bisbal, J., Lawless, D., Wu, B., Grimson, J., 1999. Legacy information systems: issues and directions. *IEEE Software* 16, 103–111.
- Bisbal, J., Lawless, D., Wu, B., Grimson, J., Wade, V., Richardson, R., O'sullivan, D., 1997. An overview of legacy information system migration. In: *Software Engineering Conference, 1997. Asia Pacific... and International Computer Science Conference 1997. APSEC'97 and ICSC'97. Proceedings. IEEE*, pp. 529–530.
- Blodget, H. 2011. Amazon's cloud crash disaster permanently destroyed many customers' data. Available at <http://www.businessinsider.com.au/amazon-lost-data-2011-4> [Last access June 2015].
- Börner, R., 2010. Applying situational method engineering to the development of service identification methods. *AMCIS*.
- Braun, V., Clarke, V., 2006. Using thematic analysis in psychology. *Qual. Res. Psychol.* 3, 77–101.
- Brebner, P.C., 2012. Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (IaaS) cloud applications. In: *Proceedings of the Third Joint WOSP/SIPEW International Conference on Performance Engineering. ACM*, pp. 263–266.
- Brinkkemper, S., 1996. Method engineering: engineering of information systems development methods and tools. *Inf. Software Technol.* 38, 275–280.
- Brodie, M., Stonebraker, M., 1995. *Migrating Legacy Systems: Gateways, Interfaces & the Incremental Approach*. Morgan Kaufmann Publishers Inc.
- Buyya, R., Yeo, C.S., Venugopal, S., Broberg, J., Brandic, I., 2009. Cloud computing and emerging IT platforms: vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.* 25, 599–616.
- Chauhan, M.A., Babar, M.A., 2012. Towards process support for migrating applications to cloud computing. In: *2012 International Conference on Cloud and Service Computing (CSC)*, 22–24 Nov. 2012, pp. 80–87.
- Chitchyan, R., Rashid, A., Sawyer, P., Garcia, A., Alarcon, M.P., Bakker, J., Tekinerdogan, B., Clarke, S. & Jackson, A. 2015. Survey of aspect-oriented analysis and design approaches.
- Dalheimer, M., Pfreundt, F.-J., 2009. Genlm: license management for grid and cloud computing environments. In: *Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on*. IEEE, pp. 132–139.
- Dedeke, A., 2012. Improving legacy-system sustainability: a systematic approach. *IT Prof.* 14, 38–43.
- Dieste, O., Padua, O., 2007. Developing search strategies for detecting relevant experiments for systematic reviews. In: *Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on*. IEEE, pp. 215–224.
- Erlikh, L., 2000. Leveraging legacy system dollars for e-business. *IT Prof.* 2, 17–23. Available at Fahmideh, M.. A generic process metamodel for cloud migration.
- Fox, A., Griffith, R., Joseph, A., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A. & Stoica, I. 2009. Above the Clouds: A Berkeley View of Cloud Computing. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley, Rep. UCB/EECS, 28.
- Girish, L., Guruprasad, H., 2014. Survey on service migration to cloud architecture. *Int. J. Comput. Sci. Eng. Technol.* 5, 507–510.
- Gonzalez-Perez, C., Henderson-Sellers, B., 2008. *Metamodelling for Software Engineering*. Wiley Publishing.
- Greenhalgh, T., Taylor, R., 1997. How to read a paper: papers that go beyond numbers (qualitative research). *BMJ* 315, 740–743.
- Gu, Q., Lago, P.A., 2007. Stakeholder-driven service life cycle model for SOA. In: *2nd International Workshop on Service Oriented Software Engineering: in Conjunction with the 6th ESEC/FSE Joint Meeting. ACM*, pp. 1–7.
- Guo, C.J., Sun, W., Huang, Y., Wang, Z.H., Gao, B., 2007. A framework for native multi-tenancy application development and management. In: *E-Commerce Technology and the 4th IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services, 2007. CEC/EEE 2007. The 9th IEEE International Conference on*. IEEE, pp. 551–558.
- Hamdaqa, M., Tahvildari, L., 2012. Cloud computing uncovered: a research landscape. *Adv. Comput.* 86, 41–85.
- Henderson-Sellers, B., Ralyté, J., 2010. Situational method engineering: state-of-the-art review. *J. UCS* 16, 424–478.
- Holland, C., Light, B., 1999. A critical success factors model for ERP implementation. *Software, IEEE* 16, 30–36.
- Jacobs, D., Aulbach, S., 2007. Ruminations on Multi-Tenant Databases. *BTW*, pp. 514–521.
- Jamshidi, P., Ahmad, A., Pahl, C., 2013. Cloud migration research: a systematic review. *IEEE Trans. Cloud Comput.* 1, 142–157.
- Karam, G.M., Casselman, R.S., 1993. A cataloging framework for software development methods. *Computer* 26, 34–45.
- Khadka, R., Reijnders, G., Saeidi, A., Jansen, S., Hage, J., 2011. A method engineering based legacy to SOA migration method. In: *Software Maintenance (ICSM), 2011 27th IEEE International Conference on*, 25–30 Sept. 2011, pp. 163–172.
- Khadka, R., Saeidi, A., Idu, A., Hage, J., Jansen, S., 2013. Legacy to SOA evolution: a systematic literature review. In: *Ionita, A.D., Litoiu, M., Lewis, G. (Eds.), Migrating Legacy Applications: Challenges in Service Oriented Architecture and Cloud Computing Environments*. Citeseer.
- Khusidman, V. & Ulrich, W. 2007. Architecture-driven modernization: transforming the enterprise. DRAFT V. 5. OMG (2007).
- Kitchenham, B., Linkman, S., Law, D., 1997. DESMET: a methodology for evaluating software engineering methods and tools. *Comput. Control Eng. J.* 8, 120–126.
- Kitchenham, B., Pearl Brereton, O., Budgen, D., Turner, M., Bailey, J., Linkman, S., 2009. Systematic literature reviews in software engineering – a systematic literature review. *Inf. software technol.* 51, 7–15.
- Kitchenham, B., Pfleeger, S.L., Pickard, L.M., Jones, P.W., Hoaglin, D.C., El Emam, K., Rosenberg, J., 2002. Preliminary guidelines for empirical research in software engineering. *IEEE Trans. Software Eng.* 28, 721–734.
- Kwon, Y.-W., Tilevich, E., 2014. Cloud refactoring: automated transitioning to cloud-based services. *Autom. Software Eng.* 21, 345–372.
- La, H.J., Kim, S.D., 2009. A systematic process for developing high quality saas cloud services. *Cloud Computing*. Springer.
- Lane, S., Richardson, I., 2011. Process models for service-based applications: a systematic literature review. *Inf. Software Technol.* 53, 424–439.
- Laszewski, T., Nauduri, P., 2011. *Migrating to the Cloud: Oracle Client/Server Modernization*. Elsevier.
- Louridas, P., 2010. Up in the air: moving your applications to the cloud. *IEEE Software* 27 (4), 6–11.
- Mahmood, Z., 2013. *Cloud Computing Methods and Practical Approaches*. Springer-Verlag, London.
- Medina, V., Garcia, J.M., 2014. A survey of migration mechanisms of virtual machines. *ACM Comput. Surv.* 46, 1–33.
- Mohagheghi, P., Berre, A., Henry, A., Barbier, F., Sadovykh, A., 2010. REMICS-REuse and migration of legacy applications to interoperable cloud services. In: *Nitto, E., Yahyapour, R. (Eds.), Towards a Service-Based Internet*. Springer Berlin Heidelberg.
- Nathuji, R., Kansal, A., Chaffarkhah, A., 2010. Q-clouds: managing performance interference effects for qos-aware clouds. In: *Proceedings of the 5th European Conference on Computer Systems. ACM*, pp. 237–250.
- Pahl, C., Xiong, H., Walshe, R., 2013. A comparison of on-premise to cloud migration approaches. *Service-Oriented and Cloud Computing*. Springer.
- Ramsin, R., Paige, R.F., 2008. Process-centered review of object oriented software development methodologies. *ACM Comput. Surv. (CSUR)* 40, 3.
- Razavian, M., 2013. *Knowledge-Driven Migration to Services*. Vrije Universiteit.
- Razavian, M., Lago, P., 2015. A systematic literature review on SOA migration. *J. Software: Evol. Process* 27, 337–372.
- Ried, S., Kisker, H., 2011. *Sizing the Cloud—A BT Futures Report*. Forrester Research, Inc., Cambridge, MA.
- Rimal, B.P., Eunmi, C., Lumb, I.A., 2009. Taxonomy and survey of cloud computing systems. In: *INC, IMS and IDC, 2009. NCM '09. Fifth International Joint Conference on*, 25–27 Aug. 2009, pp. 44–51.
- Ristenpart, T., Tromer, E., Shacham, H., Savage, S., 2009. Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds. In: *Proceedings of the 16th ACM Conference on Computer and Communications Security. ACM*, pp. 199–212.
- Singh, S., Chana, I., 2012. Cloud based development issues: a methodical analysis. *Int. J. Cloud Comput. Serv. Sci.* 2, 73–84.
- Sneed, H.M., 1995. Planning the reengineering of legacy systems. *Software, IEEE* 12, 24–34.
- Sneed, H.M., 2006. Integrating legacy software into a service oriented architecture. In: *Software Maintenance and Reengineering, 2006. CSMR 2006. Proceedings of the 10th European Conference on*, 22–24 March 2006, pp. 11–14.
- Sommerville, I., Ransom, J., 2005. An empirical study of industrial requirements engineering process assessment and improvement. *ACM Tran. Software Eng. Methodol.* 14, 85–117.
- Stafford 2013. Cloud developer job description getting more complicated all the time. in <http://searchcloudapplications.techtarget.com/opinion/Cloud-developer-job-description-getting-more-complicated-all-the-time> (last access 9/07/2015).
- Stone, A., 2001. *Keeping Legacy Software Alive*. Bloomberg Businessweek.
- Strauch, S., Andrikopoulos, V., Karastoyanova, D., Leymann, F., 2014. Migrating enterprise applications to the cloud: methodology and evaluation. *Int. J. Big Data Intell.* 1, 127–140.
- Sturm, A., Shehory, O., 2004. A framework for evaluating agent-oriented methodologies. In: *Agent-Oriented Information Systems*. Springer, pp. 94–109.
- Taher, Y., Nguyen, D.K., Lelli, F., Van Den HEUVEL, W.-J., Papazoglou, M., 2012. On engineering cloud applications-state of the art, shortcomings analysis, and approach. *Scalable Comput.* 13, 215–231.
- Toosi, A.N., Calheiros, R.N., Buyya, R., 2014. Interconnected cloud computing environments: challenges, taxonomy, and survey. *ACM Comput. Surv.* 47, 7.

- Tran, Q.-N.N. & Low, G.C. 2005. Comparison of ten agent-oriented methodologies. *Agent-Oriented Methodologies*, Idea Group Publishing, pp. 341–367.
- Vaquero, L.M., Rodero-Merino, L., Buyya, R., 2011. Dynamically scaling applications in the cloud. *ACM SIGCOMM Comput. Commun. Rev.* 41, 45–52.
- Warren, I., Ransom, J., 2002. Renaissance: a method to support software system evolution. In: *Computer Software and Applications Conference, 2002. COMPSAC 2002. Proceedings. 26th Annual International, 2002*, pp. 415–420.
- Wood, B., Pethia, R., Gold, L.R. & Firth, R. 1988. A guide to the assessment of software development methods. DTIC Document.
- Yl, W., Blake, M.B., 2010. Service-oriented computing and cloud computing: challenges and opportunities. *Internet Comput. IEEE* 14, 72–75.
- Zimmermann, O., Miksovich, C., Küster, J.M., 2012. Reference architecture, metamodel, and modeling principles for architectural knowledge management in information technology services. *J. Syst. Software* 85, 2014–2033.

Mahdi Fahmideh Gholami is a PhD candidate at the University of New South Wales. He holds a B.Sc. and an M.Sc. in the software engineering. Mahdi's general research interests are design science research, conceptual modeling, method engineering, cloud computing, and service-oriented architecture. Prior starting his PhD, Mahdi has served as a system analyst and programmer in national government IT projects for several years.

Farhad Daneshgar received his PhD in Information Systems from the University of Technology, Sydney Australia. He is a Senior Lecturer at the UNSW Business School, University of New South Wales, Sydney, Australia, and is an adjunct Professor at Bangkok University. Farhad is the creator of the Awareness Modeling Language, and has published extensively in the areas of Knowledge Management and Enterprise Systems, and was awarded twice for his Outstanding Research Article at the University of New South Wales. Farhad is a member of editorial board in five academic journals.

Graham Low received the BE and PhD degrees from The University of Queensland. He is an emeritus professor of information systems in the School of Information Systems, Technology and Management at the University of New South Wales. His research program focuses on the implementation and adoption of new technologies. This can take the form of new/modified approaches/techniques for information systems development such as methodological approaches to agent-oriented information systems design and management of the information systems design and implementation process.

Ghassan Beydoun received a degree in Computer Science and a Ph.D. degree in Knowledge Systems from the University of New South Wales. He is currently an Associate Professor at the School of Computing and Information Technology at the University of Wollongong. He is also Director of Software Design Science Research Centre. He has authored more than 100 papers for international journals and conferences. He investigates the best uses of ontologies in developing methodologies for distributed systems. His other research interests include multi-agent systems applications, ontologies and their applications, knowledge acquisition and disaster management.