# Capitalizing on Empirical Knowledge during Cloud Computing Adoption

Mahdi Fahmideh, Ghassan Beydoun

Faculty of Engineering and Information Technology, University of Technology Sydney
Mahdi.Fahmdeh@uts.edu.au, Ghassan.Beydoun@uts.edu.au (corresponding author)

**Abstract.** Like many other IT innovations, adopting cloud computing technology in IT-based organisations may not be a hazard-free endeavour and demands a proper understanding of requirements and risks involved prior to taking any action. The number of accounts on using of cloud services to empower existing legacy systems is ever increasing. The time is ripe to undertake an analysis to gain a realistic view of what migrating legacy systems to cloud may offer, an understanding of exceptional situations causing goal failure in such a transition, and insights on countermeasures. This body of knowledge, although is useful, is dispersed over the current literature. It is hard for busy practitioners to digest, synthesize, and harness this body of knowledge into practice when integrating legacy systems with cloud services. We address this issue by making an innovative synergy between the approaches *evidence-based software engineering* and *goal-oriented modelling*. Through using evidence-based software engineering approach, we develop a repository of commonly occurred obstacles and platform agnostic resolution tactics that is later utilized during a systematic goal-obstacle elaboration process regarding migrating legacy systems to cloud platforms. The application of the framework is also demonstrated.

**Keywords**: Cloud Computing Adoption, Legacy systems, Evidence-based software engineering, Goal-oriented requirement engineering, Legacy system reengineering

## 1. Introduction

Cloud computing makes a fundamental shift in delivering services to clients. A perennial concern of IT managers embarking on migrating legacy systems to cloud platforms is to ensure that benefits of cloud services are attainable in their organisation setting (Zardari, Bahsoon et al. 2014), (Khajeh-Hosseini, Sommerville et al. 2011). Despite all hypes over cloud computing, still some organisations are reluctant to move their legacy systems to the cloud due to uncertainties originating from reasons such as security, facing unexpected issues that are out of the control of service provider and consumer, failure experience of other organisations, interoperability and vendor lock-in problem, cultural shift, and many others (Chow, Golle et al. 2009; Pepitone 2011; Linthicum 2012; Tsidulko 2016).

Since the emergence of cloud computing technology in 2007, there has been an ever increasing account, published by both academia and industrial communities, on utilizing cloud services to improve working legacy systems in different organisational and project settings. Both of these sources provide a test bed that can be used for informed decisions in moving and maintaining legacy systems in cloud platforms. But given the spread of the literature, a systematic support is not yet available to take advantage of this body of knowledge. This requires synthesis and analysis to make it more explicit, reusable, and accessible.

Despite repeated calls in previous researches (Giovanoli 2012; Zimmermann, Wegmann et al. 2015) for capturing cloud adoption knowledge to improve decision making and risk elimination, existing work towards this remains intangible. This article alleviates this gap via deploying a combination of *evidence-based software engineering* (Kitchenham, Pearl Brereton et al. 2009) and *goal-oriented modelling* (Yu 1997). Based on the premise that failure to adequately identify and tackle such obstacles in advance may cause a cloud-enabled system which is costly to change if they happen in later stages, we introduce a knowledge-based decision support framework for the eliciting, modelling, and reasoning about requirements in migrating legacy systems to cloud platforms.

The framework includes collections of commonly occurring obstacles hindering satisfying quality goals and corresponding countermeasures to handle these obstacles. These collections are stored in a repository. They are identified through an extensive literature review of published academic studies, empirical or experience reports explaining situations in which a technical or non-technical obstacle may occur against cloud migration goals.

The framework adopts a goal-oriented modelling approach to offer a visualized process for systematically analysing goals and obstacles in migrating legacy systems to cloud platforms. The evidence-based repository is utilized in a goal-oriented elaboration process. We believe the proposed framework helps a system architect in better exploration of potential risks and improving reliability of decision outcomes in migrating and maintaining legacy systems in cloud platforms. We illustrate the applicability of the framework in two scenarios of cloud migration.

The rest of this article is organised as follows. Section 2 provides a motivating scenario of this study and reviews related work in the literature. Section 3 presents the research methodology followed to develop and evaluate the proposed framework. Section 4 delineates the development of the framework components. Section 5 illustrates the application of the framework in a scenario of moving an open-source Web-based legacy system to a cloud platform. Section 6 summarises the article with a discussion of limitations and future directions of the research.

## 2. Motivating scenario and related work

Uncertainty about a particular IT innovation reflects a lack of knowledge about the technology involved. This acts as a barrier and adds risks to the technology adoption. In essence, individuals and social system involved would reject the innovation and the technology (Rogers 2003). From a technical point of view, potential risks include security, incompatibilities, and reduced performance. The current study is inspired by the following scenario adapted from (Khajeh-Hosseini, Greenwood et al. 2010; El-Gazzar, Hustad et al. 2016): Assume an organization provides a supply chain service across the country via several Web-based integrated applications including customer relationship management, order processing, payment, inventory management, shipping, and expense reimbursement. Top level management intends to promote the organization's competitiveness via expanding their services throughout the world. Nevertheless, the organization could not afford procurement and maintenance of new infrastructure to support timely processing of orders and large transactions when the number of customers increased. The top level management is attracted to the cloud services as it is said to provide powerful infrastructures along with a wide-range of services. A system architect is appointed to offer an overall architectural solution to host some of these legacy systems to the cloud as a co-location and to achieve higher throughput. However, she is unsettled with many questions, for example:

(i) By moving these systems to the cloud, will higher system performance be attainable in all situations?
(ii) What risks are likely to occur to obstruct reducing infrastructure cost and system security in the cloud? and
(iii) How such risks can be negated or at least reduced in advance instead of experiencing them in later stages?

In assisting system architects to tackle above questions, several solutions have been proposed. Drawing upon some relevant studies to compare decision making frameworks (e.g. (Babar, Liming et al. 2004) and following some guidelines for legacy system cloud enablement (Fahmideh, Daneshgar et al. 2016)), eight analysis criteria were set: migration type, lifecycle focus, evaluation granularity, process support, stakeholder involvement, modelling language, experience repository, and tool support. In the following, existing literature is discussed in the view of the criteria to characterise existing frameworks and to show how the proposed framework in this study is situated in the context of existing literature and where it supersedes or synthesizes previous work.

**(i) migration type.** Existing studies can be examined regarding the type of migration they aim to evaluate. Based on the common service delivery models IaaS, SaaS, and PaaS, several variants can be considered through which legacy systems can utilize cloud services. In (Fahmideh, Daneshgar et al. 2016) these migration variants are defined as follows. In *Type I* the business logic layer of a legacy system, which offers discrete and reusable functionality, is deployed in the cloud infrastructure through IaaS model such as Amazon EC2 but the data layer is kept in on-premise network. (Anstett, Leymann et al. 2009) presents some factors (e.g. operating system, platform middleware) that a system architect should take into account when deploying business process engines in IaaS. In *Type II*, legacy system components are replaced with a fully tested cloud services using SaaS model. (Godse and Mulik 2009) and (Wu, Lan et al. 2011) are two example frameworks presenting approaches for selecting the most appropriate SaaS product for organizational needs. In *Type III*, the database of a legacy system is deployed on a cloud data store provider such as Amazon Simple Storage Service (S3), Amazon Elastic Block Store, Dropbox, or Zip Cloud whilst business logic components are kept in on-premise network. In *Type IV*, the database of a legacy is modified and converted to a cloud database solution such as Amazon SimpleDB, Google App Engine data store, or Google Cloud SQL. Finally, in *Type V* the whole legacy system stack is encapsulated in virtual machines and then run in the cloud infrastructure. The framework proposed in (Khajeh-Hosseini, Greenwood et al. 2012) supports decision makers in identifying concerns to examine the costs of deploying IT systems on the cloud.

**(ii) lifecycle focus.** Decision making frameworks can be classified regarding migration phases, meaning that when a framework is appropriate to use. Fahmideh et al. define three phases namely plan, enable, and maintain (Fahmideh 2016b). The frameworks related to the plan phase are concerned with the feasibility assessment of adopting cloud services in an organisation and filtering out migration types meet organisational requirements. Some studies such as (El-Gazzar, Hustad et al. 2016), (Low, Chen et al. 2011) and (Wu 2011) mainly investigate important factors such as security, cost, and organizational readiness that should be taken into account when adopting cloud services. Other approaches proposed in (De Assunçao, Di Costanzo et al. 2009), (Deelman, Singh et al. 2008), (Kondo, Javadi et al. 2009), (Walker, Brisken et al. 2010), and (Khajeh-Hosseini, Greenwood et al. 2012) focus on analysing the feasibility of cloud adoption from the cost saving point of view and suggest if a deployment option is cost effective. For these studies concerns related to re-architecting legacy systems to cloud such as data security, interoperability of legacy components and cloud services, or system performance in the cloud are not covered. In the enable phase, cloud services that meet computational requirements of legacy systems and the most suitable legacy components that can benefit from cloud services are selected following with defining an optimum deployment of these components on cloud servers. Multi-criteria based decision making techniques are applied for shortlisting and ranking of candidate cloud services: e.g. Analytic Hierarchy Process (AHP) (Garg, Versteeg et al. 2013), (Godse and Mulik 2009), Analytic Network Process (ANP) (Menzel, Schönherr et al. 2013) along with many other studies. There are situations in which post-migration assessment needs to be performed. The decision making frameworks related to this phase may suggest which legacy components should be de-migrated to local environment or empowered with new cloud services based on evolution needs and changes in operational environment conditions (Scandurra, Mongiello et al. 2016).

**(iii) evaluation granularity.** The unit of analysis in decision making frameworks might be at the different levels such as organizational, legacy system, or component. A system architect may proceed through this funnel view to evaluate suitability and filter out migration variants do not meet their requirements. Some frameworks such as (Nikkhouy 2013), (Christoforou and Andreou 2013), and (Low, Chen et al. 2011) assess whether an organisation is ready to benefit from cloud services. On the other hand, the frameworks (Tak, Urgaonkar et al. 2011), (Juan-Verdejo and Baars 2013), (Menzel and Ranjan 2012), (Khajeh- Hosseini, Greenwood et al. 2012), (Fittkau, Frey et al. 2012), (Saripalli and Pingali 2011), and (Calheiros, Ranjan et al. 2011) examine which legacy systems are adequate for moving to the cloud using migration types V. Furthermore, (Leymann, Fehling et al. 2011) suggests an approach to find legacy system components suitable for being cloud-enabled based on criteria such as latency, data transfer, and component dependencies.

**(iv) evaluation approach.** Decision making frameworks use a wide range of techniques to satisfy desired goals. This criterion is useful to know the level of information required by a framework and which techniques are required for an evaluation exercise. To name a few, some frameworks are pure metric based (e.g. (De Assunçao, Di Costanzo et al. 2009), (Deelman, Singh et al. 2008), and (Kondo, Javadi et al. 2009)), a few use goal-reasoning (e.g. (Zardari, Bahsoon et al. 2014) and

(Scandurra, Mongiello et al. 2016)), and some combine different techniques (e.g. (Menzel and Ranjan 2012), (Fittkau, Frey et al. 2012), (Saripalli and Pingali 2011)), or use optimisation technique (e.g. (Leymann, Fehling et al. 2011)).

**(v) process support.** Decision making frameworks may define a precise definition of steps and their sequence along with input and output products, guidelines, controls, and heuristics for an accurate assessment which ultimately help a system architect to accomplish decision making goals. Except for some frameworks presented in Table 1, the majority of frameworks are reviewed in this section provide at least a coarse-grained description of evaluation process, tough they differ in the detail of a process providing. However, some frameworks are related to the planning phase of the cloud adoption do not have an explicit process support.

**(vi) stakeholder involvement.** As with many scenarios of decision making in the software engineering, cloud computing adoption may involve different stakeholders such as cloud service provider, consumer, broker, developers, project manager, and end-user whose interests attempt to influence risks and benefits. Active participation of stakeholders which considers elicitation of stakeholders' goals and their prioritization and conflicts during decision making process is essential for the quality of evaluation process. While existing frameworks recognize the importance of incorporating key stakeholders, they vary in type of stakeholders involving.

**(vii) modelling language.** Evaluation frameworks can be compared in terms of their attention to employ a notation to represent elements, semantic interpretation, and outcome of each decision steps. Using a modelling language can facilitate communications and understandability of decision making process for its stakeholders and the automation of an evaluation process. For example, the framework by (Christoforou and Andreou 2013) uses Influence Diagrams, a directed acyclic graph with nodes showing decision variables and how they influence each other. Nodes representations along with their dependencies model decision making questions and provide final decision nodes. In another example, (Zardari, Bahsoon et al. 2014) uses goal-oriented modelling to represent risks encountered and mitigating strategies in using cloud services.

**(viii) experience repository.** The notion of reuse is a commonly accepted means for increasing productivity in the software engineering field (A. Aurum 2003). Like many software development activities, decision making might be a knowledge-intensive process that can be a costly exercise if each evaluation starts from scratch in an ad-hoc manner. The effort of a decision making can be reduced if knowledge from activities in previous cloud computing adoption scenarios is be maintained and incorporated in further decision making scenarios. In addressing this need, CLiCk (Cloud Life Cycle) provides a repository containing historical information on QoS of different service platforms to improve the accuracy of service selection (Giovanoli 2012). Reusing and sharing recurring decision logs in the course of re-architecting legacy systems to cloud platforms is suggested by (Zimmermann, Wegmann et al. 2015). In another work, (Menzel, Schönherr et al. 2013) provides a catalogue of criteria to be reused for creating customized evaluation methods addressing criteria and aiding to evaluate alternative service providers and find the one suits requirements of cloud service consumers.

**(ix) tool support.** A decision making process might be involved with time consuming and expensive tasks such as collecting, documenting, and maintaining relevant domain data. In particular, consumers at the early stage of transition to the cloud may face a higher number of risks in utilizing cloud services which should be carefully evaluated (Lacity, Khan et al. 2009). Decision-making tools can capture, for example, alternative cloud services and their service level agreements offering by different providers, costs and risk factors relevant to decision scope, automate as many as decision steps, and come up with evaluation outcomes. The importance of tool support is recognized by some frameworks such as (Khajeh- Hosseini, Greenwood et al. 2012) and (Menzel and Ranjan 2012).

**Concluding remark.** Table 1 shows the summarization of characteristics of existing frameworks in the view of analysis criteria. Our proposed framework in the current study distinguishes itself from the existing one in the view of analysis criterion *experience repository*. Compared to the existing frameworks reviewed above, our framework provides an evidential knowledge repository of reusable cloud-specific obstacles and corresponding resolution tactics along with a visualization mechanism for systematically analysing risks in migrating legacy systems to cloud platforms. Perhaps, the only notable close work to our framework is by (Giovanoli 2012) which provides a repository containing information on the different service providers for the provider- and service classification. However, it does not cover several areas of obstacles and resolution tactics (e.g. incompatibilities between legacy systems and cloud platforms). In fact, none of the existing frameworks utilises cloud adoption knowledge during goal reasoning targeting legacy system migration to cloud environments to address probable risks may occur and undertake precaution countermeasures. They rather rely on knowledge of system architect which might be imprecise and incomplete. By utilizing an evidential knowledge provided by the proposed framework, system architects can get an informed insight of attainability of goals via cloud-enablement of legacy systems. In addition, our proposed framework differs from these studies by providing a detailed analysis process of the cloud-specific goal-obstacle supported by the evidential knowledge repository.

Table 1. Literature comparison addressing the evaluating of cloud computing adoption

| Study | Aim | Migration Type | Lifecycle Focus | Evaluation granularity | Evaluation approach | Process support | Stakeholder involvement | Modelling language | Experience repository | Tool Support |
|---|---|---|---|---|---|---|---|---|---|---|
| (Anstett, Leymann et al. 2009) | Identifying factors such as operating system, platform middleware and legacy system to be considered in deploying BPEL on IaaS. | Type V | Plan phase | Legacy system | Not specified | Not specified | Not specified | Not specified | Not considered | Not available |
| (Khajeh- Hosseini, Greenwood et al. 2012) | Providing a decision making support for identifying concerns in using IaaS. | Type V | Plan phase | Legacy systems | Cost modelling | Yes | Yes | Deployment model of system | Not considered | Yes |
| (El-Gazzar, Hustad et al. 2016) | Identifying inhibitors and organisational drivers are involved in a decision making for cloud computing adoption. | All | Plan phase | Organisation | Not specified | Not specified | Not specified | Not specified | Not considered | Not available |
| (Low, Chen et al. 2011) | Exploring factors affecting organisations in adopting cloud computing. | All | Plan phase | Organisation | Not specified | Not specified | Not specified | Not specified | Not considered | Not available |
| (Wu, Lan et al. 2011) | Exploring factors influencing successful SaaS adoption. | Type II | Plan phase | Organisation | Decision making trial and evaluation laboratory | Not specified | Not specified | Cause-effect diagram | Not considered | Not available |
| (De Assunçao, Di Costanzo et al. 2009) | Evaluating the optimality of scheduling strategies used by an organisation to reduce response time in using IaaS. | Type V | Enable phase | Organisation | Performance metrics | Not specified | Not specified | Not specified | Not considered | Not available |
| (Deelman, Singh et al. 2008) | Analysing the cost-performance trade-off between difference executions and resource provisioning plans by legacy systems. | Type III | Enable phase | Legacy systems | Performance metrics | Not specified | Not specified | Not specified | Not considered | Not available |
| (Kondo, Javadi et al. 2009) | Comparing cost and performance of legacy systems in using IaaS. | Type V | Enable phase | Legacy systems | Performance metrics | Not specified | Not specified | Not specified | Not considered | Not available |
| (Walker, Brisken et al. 2010) | Reasoning about the cost of leasing infrastructure from cloud storage. | Type III | Enable phase | Organisation | Net present value | Not specified | Not specified | Not specified | Not considered | Not available |
| (Garg, Versteeg et al. 2011) | Measuring, comparing, and prioritizing cloud services based on users' requirements. | Type V | Enable phase | Organisation/ Legacy systems | QoS metrics | Yes | Implicitly supported | Not specified | Not considered | Not available |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| (Godse and Mulik 2009) | Analysing and selecting appropriate SaaS products. | Type II | Enable phase | Organisation | AHP | Embedded in framework description | Implicitly supported | Not specified | Not considered | Not available |
| (Menzel, Schönherr et al. 2013) | Examining if IaaS meets organisation needs by evaluating and ranking alternatives using a set of criteria catalogue. | All | Enable phase | Organisation/ Legacy systems | ANP | Yes | Implicitly supported | Not specified | Not considered | Yes |
| (Scandurra, Mongiello et al. 2016) | Redeploying e-commerce cloud applications on different servers at run-time based on evolving requirements, sudden changes in the operational environment conditions, and application traffic. | | Maintain phase | Legacy systems | Goal reasoning | Embedded in framework description | Not specified | Graph modelling | Not considered | Not available |
| (Nikkhouy 2013) | Exploring potential benefits and risks in migrating legacy systems to cloud services. | All | Plan phase | Organisation | Change analysis | Yes | Implicitly supported | Cause and effect diagram | Not considered | Not available |
| (Christoforou and Andreou 2013) | Assessing the feasibility of the cloud adoption in organizations regarding factors such as security, legal issues, availability, cost, ROI, compliance, performance, scalability, and data access/import-export. | All | Plan phase | Organisation | Analysing influencing factors | Embedded in framework description | Yes | Influence diagrams modelling | Not considered | Not available |
| (Tak, Urgaonkar et al. 2011) | Exploring factors such as workload intensity, growth rate, storage capacity and software licensing costs affecting the cost of deployment options in the cloud. | V | Plan phase | Legacy systems | Using benchmarks representing of different scenarios | Not specified | Not specified | NPV models | Not considered | Not available |
| (Juan-Verdejo and Baars 2013) | Identifying suitable components of legacy systems for deploying on IaaS with respect to interdependencies among components and factors such as data transfer volumes, performance, sensitivity of cloud-based data repositories, and exposure to public networks. | V | Enable phase | Legacy systems | Combination of scenario based & AHP | Embedded in framework description | Yes | Legacy system architecture model | Not considered | Not available |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| (Menzel and Ranjan 2012) | Identifying a compatible combination of software images (e.g., Web server image) in mapping web applications to virtualized cloud services while expected QoS of applications are satisfied. | V | Enable phase | Legacy systems | Combination of optimization and AHP | Embedded in framework description | Yes | Simulation models | Not considered | Not available |
| (Fittkau, Frey et al. 2012) | Evaluation of competing cloud deployment options and finding the most suitable mapping of virtual machines to cloud services regarding cost and system performance. | V | Enable phase | Legacy systems | Combination of optimization and scenario-based | Yes | Yes | Simulation models | Not considered | Not available |
| (Saripalli and Pingali 2011) | Ranking legacy system workloads for migrating to cloud environments based on attributes such as latency, bandwidth, and cost. | All | Enable phase | Legacy systems | Combination of multi-attribute decision making and wide-band Delphi | Embedded in framework description | Yes | Decision matrix | Not considered | Not available |
| (Calheiros, Ranjan et al. 2011) | Determining the best deployment options of components of legacy systems on cloud servers whilst QoS are satisfied. | V | Enable phase | Legacy systems | Scenario-based | Embedded in framework description | Yes | Simulation models | Not considered | Yes |
| (Leymann, Fehling et al. 2011) | Rearrangement of the legacy application deployment topology on cloud servers regarding dependencies among its components and requirements such as latency, transfer, and data privacy are addressed. | V | Enable phase | Legacy systems | Optimisation algorithm (e.g. simulated annealing) | Yes | Not specified | Metamodeling, application templates | Not considered | Yes |
| (Giovanoli 2012) | Assessing and selecting the most suitable cloud services via guidelines provided in a database containing information of different cloud service providers. | All | Enable phase | Legacy systems | Not specified | Not specified | Not specified | Not specified | Information repository of cloud service providers | Yes |
| (Zardari, Bahsoon et al. 2014) | Prioritising obstacles related to cloud service adoption and resolution tactics. | All | Plan phase | Legacy systems | Goal reasoning and AHP | Yes | Yes | Goal models | Not considered | No |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| This work | Analysing goal-obstacle in migrating legacy systems to cloud platforms along with resolution tactics which utilizes an evidence-based repository during the steps of the goal-oriented elaboration process. | All | Plan and enable phases | Legacy systems | Goal reasoning and evidence-based approach | Yes | Yes | Goal models | Yes | No |

## 3. Research methodology

As this research aims to prescribe an IT artefact, i.e. the framework, design science research (Henver, March et al. 2004) suits this inquiry. An essential feature of this genre of research is to produce a viable artefact that addresses a relevant solution to an unsolved problem. Based on guidelines in conducting the design science research, the following three phases were followed:

*Phase 1- Problem identification*. This phase has been already described in the sections 1 and. That is, there is a lack of systematic knowledge reuse to improve reliability of requirement analysis results and decision outcomes during migrating legacy systems to cloud platforms. Our research objective can be stated as “development of a systematic approach for goal-obstacle analysis related to migrating legacy systems to cloud platforms framework which utilizes empirical evidence of obstacles and resolutions on adopting cloud services”.

*Phase 2- Design and develop*. Through this, the proposed framework is developed. The framework constitutes two core components:

(i) an empirical knowledge repository of recurring goals, obstacles, and countermeasures reusable in re-architecting legacy systems to use cloud services,

(ii) a procedure including steps to be taken to identify cloud specific obstacles, assessing their risk (i.e. likelihood and severity), and tackling them by generating new goals.

In the design science research, different approaches and kernel theories from inside or outside of software engineering discipline that inform the artefact creation can be brought to bear (Gregor and Jones 2007). For the development of the first component, i.e. the knowledge repository, we apply an evidence-based software engineering (EBSE) paradigm (Kitchenham, Pearl Brereton et al. 2009) which provides a common body of knowledge based on practitioners' experience in using different tools and techniques of software engineering. A common technique to operationalize EBSE is Systematic Literature Review (SLR) (Kitchenham, Pearl Brereton et al. 2009) where its goal is to gather findings from different empirical studies to draw plausible conclusions (Kitchenham, Pearl Brereton et al. 2009). In this research, the framework’s repository is developed out of a SLR of published empirical studies of different scenarios of enabling legacy systems to use cloud services.

For the second component, i.e. procedure, we use a generic goal-oriented modelling framework called KAOS (Keep All Objects Satisfied) providing a support for elaborating, structuring and analysing software requirements, including both functional and non-functional requirements (Van Lamsweerde 2009). KAOS supports different levels of expression and reasoning that vary from semi-formal to formal analysis goal models depending on the reasoning precision sought (Dardenne, Van Lamsweerde et al. 1993; Van Lamsweerde and Letier 2004). In KAOS, goals are iteratively refined through top-down approach (by asking *how* questions to refine goals into sub-goals) as well as bottom-up approach (by asking *why* questions to identify parent goals). The refinement proceeds until all goals reach clear and assignable responsibilities to agents who can realize the goals. We use the KAOS’s framework in conducting goal-obstacle analysis of migrating legacy systems to cloud computing adoption to empower legacy systems. However, we believe general concepts in KAOS such as goal, obstacle, and resolution tactic are not sufficiently applicable for refining them into testable cloud migration requirements due to lack of a precise definition. For example, in KAOS the concept of obstacle refer to “an exceptional condition that prevents the goal from being satisfied” (van Lamsweerde and Letier 2000). We have specialized these concepts with empirical knowledge provided in the framework repository. For instance, the generic notion of *Obstacle* in KAOS is specialized with cloud-specific obstacles as discussed in Section 4.1.

*Phase 3 - Evaluate*. This phase appraises the efficacy of the framework resulting from Phase 2 through case studies of moving an open-source Web-based legacy system to Pivotal Cloud Foundry and a digital document processing legacy system to Microsoft Azure cloud platform.

An important feature of the design science research is the cyclical/iterative process of design and evaluation phases towards refinement and improvement of the design artefact (Henver, March et al. 2004; Peffers, Tuunanen et al. 2008). Through cycles, the designed artefact is situated in a problem space and is iteratively refined to achieve an expected efficacy. This research was conducted in two consecutive cycles where each cycle refined the framework that produced from the predecessor cycle.

The first cycle resulted in the initial version of the framework. It was validated by domain experts in the systematic literature review, goal modelling, and cloud computing.

In the second cycle, the repository component of the framework, in particular resolution tactics, were validated via comparing them with existing migration methods to verify that tactics are sufficiently complete (Fahmideh 2016b). Next, they were validated through a public survey with a population of 104 shortlisted and randomly selected domain experts from different industry sectors (Fahmideh, Daneshgar et al. 2017). This survey confirmed the relevance and importance of the resolution tactics in the process of legacy system migration to cloud platforms. The second component of the framework was validated via two case studies as presented in Section 5.

## 4. Development of the framework

As shown in Figure 1, the framework comprises (i) a repository holding empirical knowledge about collections of obstacles and resolution tactics, and (ii) a goal-oriented procedure relying on the repository in order to elaborate goal-obstacle analysis for a specific cloud adoption scenario. The structure of the following two subsections is as follow. First, in Section 4.1, the result of the literature review to develop the framework repository is described. This includes establishing a literature review protocol, conducting review, and identifying, synthesising, and organising the collections of obstacles and resolution tactics. Next, in Section 4.2, a systematic procedure for goal-obstacle analysis is presented utilizing the information in the repository.

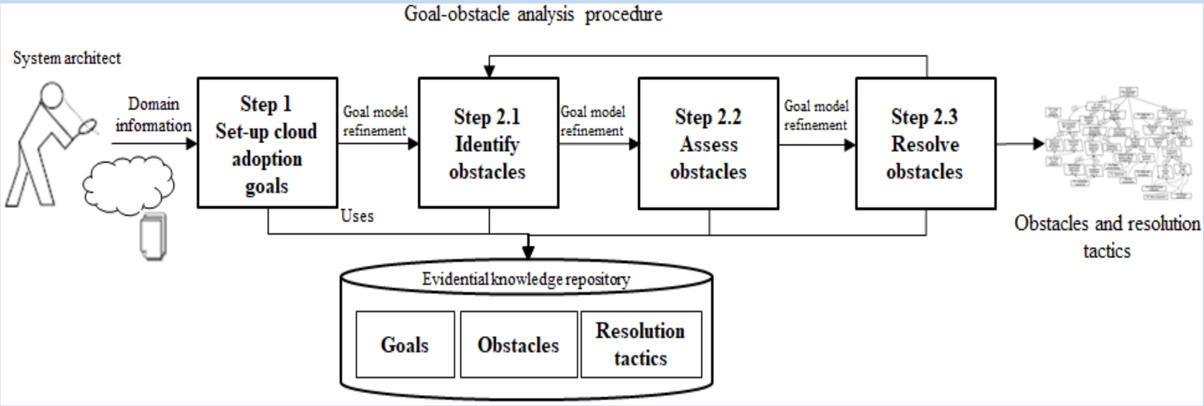

Goal-obstacle analysis procedure
System architect
Domain information
Step 1 Set-up cloud adoption goals
Goal model refinement
Step 2.1 Identify obstacles
Goal model refinement
Step 2.2 Assess obstacles
Goal model refinement
Step 2.3 Resolve obstacles
Obstacles and resolution tactics
Uses
Evidential knowledge repository
Goals
Obstacles
Resolution tactics

Figure 1. Structure of the proposed framework

### 4. 1 Repository detailed design

Three tasks were performed to develop the collections in the repository component of the framework:

(i) *Derivation of system quality goals expecting to be satisfied by migrating legacy systems to cloud environments.*
(ii) *Derivation of recurrence obstacles against achieving quality goals*
(iii) *Derivation of resolution tactics in handling these obstacles*

As shown in Figure 2, a SLR was undertaken to conduct tasks (ii) and (iii). We did not see a need to conduct SLR for task (i) as we used a fixed set of system quality goals which is commonly agreed and deemed important in the software engineering and the cloud computing literature. The objective of the SLR was to answer the following inquiries: (i) *what obstacles may occur against system quality goal when legacy system are moving to or are in operation in cloud platforms* and (ii) *what resolution tactics are suggested to address these obstacles?* The same SLR was conducted for tasks (ii) and (iii).

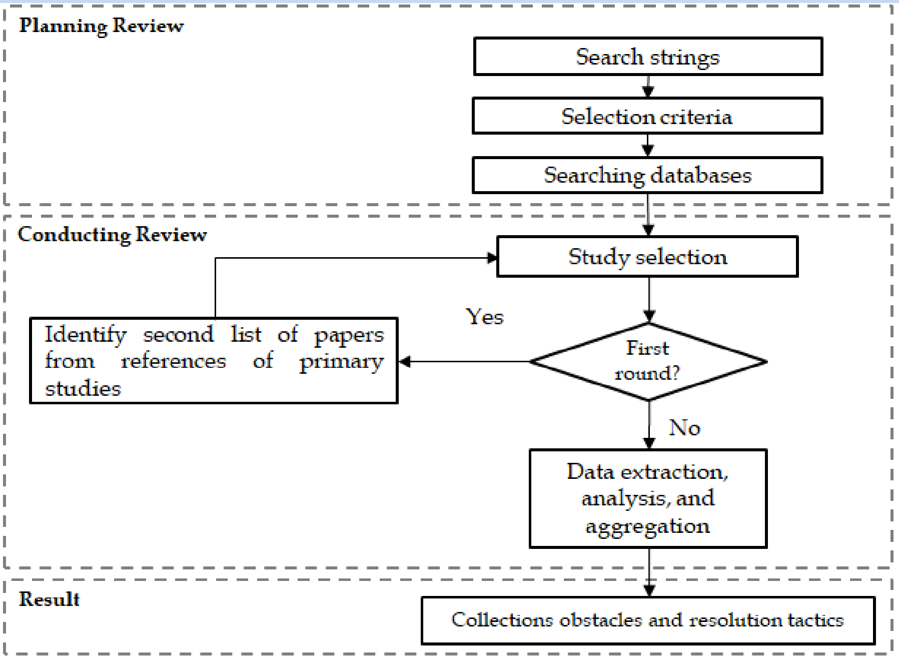

Planning Review
Search strings
Selection criteria
Searching databases
Conducting Review
Study selection
Yes
Identify second list of papers from references of primary studies
First round?
No
Data extraction, analysis, and aggregation
Result
Collections obstacles and resolution tactics

Figure 2. SLR conducted for developing the framework repository

### *Planning Review*

The objective of this phase was to tackle any researcher bias (Kitchenham, Pearl Brereton et al. 2009) through defining search strings, selection criteria, and searching databases.

**Search strings.** The search strings were defined based on the guidelines recommended in (Dieste and Padua 2007). These included: (i) defining main terms by breaking down the research questions, (ii) identifying alternative synonyms for main terms, (iii) checking the search strings in any relevant papers that retrieved, (iv) incorporating alternative synonyms using the logical operator *OR*, and (v) using the logical operator *AND* to link the main terms. The terms *Cloud*, *Cloud Computing*, *Legacy*, *Migration*, and *IaaS*, *PaaS*, and *SaaS* were set as the main keywords from which different search strings defined and combined using the logical operators OR and AND. Table 2 shows some examples of search strings.

| Table 2. list of related search strings (SS) |
|---|
| SS1: “Migration” OR “Cloud adoption” OR “Cloud migration” OR “Migration to cloud” OR “Legacy to cloud migration” OR “Legacy migration to cloud” AND [SS2 OR SS3 OR SS4 OR SS5 OR SS6] |
| SS2: “IaaS risks” OR “IaaS challenges” OR “IaaS challenges” OR “IaaS adoption” OR “IaaS benefits” |
| SS3: “PaaS risks” OR “PaaS challenges” OR “PaaS issues” OR “PaaS adoption” OR “PaaS benefits” |
| SS4: “SaaS risks” OR “SaaS challenges” OR “SaaS issues” OR “SaaS adoption” OR “SaaS benefits” |
| SS5: “Monolith application” OR “Legacy code” OR “Legacy system” OR “Existing system” OR “Legacy component” OR “Legacy software” OR “Legacy application” “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” |
| SS6: “Motivation” OR “Benefits” OR “Advantage” OR “Disadvantage” OR “Goal” OR “Objective” |

**Selection criteria.** The following criteria were set to select studies identified from the literature for purpose of the repository development. (i) the study is related to migrating legacy systems or developing systems for cloud platforms with a proper description of the context, clear objectives or research goals (ii) described situations, i.e. obstacles, that may cause goal failure and if any resolution tactics, (iii) provided a proper validations through case study, example, interview, etc., (iv) published from 2007 onwards in software engineering and information systems journals/conference proceedings; and (v) described in English language.

**Searching databases.** The following databases were searched: IEEE Explore, ACM Digital Library, SpringerLink, ScienceDirect, Wiley InterScience, ISI Web of Knowledge, Google scholar. In some cases, we found that some internet blogs and trade journal articles, named multi-vocal literature

(Ogawa and Malen 1991), provided empirical accounts of issues surrounding legacy system migration to cloud platform. This kind of sources was not overlooked to search.

### *Conducting Review*

**Study selection.** The databases numerated in the previous step were searched using the search strings. The title, abstraction, and in the most cases the content of each study was screened regarding the inclusion criteria. It is important to mention that conducting the review was not a linear and mechanical process; rather it was a hermeneutic, iterative, and informed by careful reading each study and understanding its context. Forward and backward searches were performed so that the studies cited in the references and related work sections of the study were fed into the review process as new resources. The review phase, although can never be ended, resulted in identifying 112 studies as shown in Appendix A.

**Data extraction, analysis, and aggregation.** Segments of each study that stated any obstacles or resolution tactics were extracted along with the reference to the study. Some leading questions that are used during the development of the collections were the following: (i) does the study report any technical or social obstacles that may cause cloud adoption goals fail? If so, what is the obstacle? (ii) how can the obstacle influence the successful adoption of cloud services? and (iii) Is there any resolution tactics suggested by study to overcome the obstacles? All collections obtained through this step presented in Appendix B. A synopsis is provided here in what follows:

> ***(i) Goals collection*** includes ten ready-made common software system quality goals that cloud services can positively contribute to the efficiency of working legacy systems. This includes *Availability*, *Scalability*, *Security*, *Performance*, *Customizability*, *Interoperability*, *Portability*, *Testability*, *Consistency*, and *Reduced IT cost*. These goals are used as a facilitator for initializing and refining goal models as described Section 4.2.
>
> ***(ii) Obstacles collection*** has information about 67 common probable situations causing quality goal failure and thus hampers legacy systems benefit from cloud services. The collection of obstacles, which can be technical or social, their associations to quality goals and variants of cloud adoption are presented in Appendix B.
>
> ***(iii) Resolution tactics collection*** contains 45 platform-agnostic solutions applicable for handling obstacles. Resolution tactics are a result of applying abstraction and synthesisation on existing ad-hoc implementation techniques to utilize cloud service available in the literature. They are used during the goal-obstacle analysis to explore alternative ways to resolve identified obstacles. Note that this research tended to keep resolution tactics at the abstract level. Thus their operationalization details are left to system developers or manager as to existing supportive techniques or tools available in the cloud computing marketplace. Appendix B lists all the tactics.

**Results.** Table 3 shows an excerpt of the information stored in the repository. A goal, obstacle, and resolution tactic is respectively denoted by an identifier-number G, O, and T. For example, the quality goal for a system is that it should be *interoperable* (G6) across different platforms. According to the cloud computing literature [S2], [S3], [S4], [S5], [S35], [S36], and [S37], cloud services can typically be integrated into a system through either brokering or cloud interfaces. Nevertheless, there are some potential obstacles which obstruct this quality goal. These obstacles, for example, as reported in [S23], [S24], [S12], [S38], [S25], [S26], [S27], [S39], and [S40] are *Incompatible pluggable cloud services (O19)*, *Incomplete APIs (O20)*, *Incompatible datatypes (O21)*, *Operating system incompatibility (O22)*, and *Machine-image incompatibility (O23)*. In addressing these potential obstacles, two generic tactics *Refactor legacy source code (T2)* and *Develop adaptor/wrapper (T3)* are suggested in [S65], [S66], [S67], [S75], [S76]. Maintaining consistency, the architect may slightly change the original names of these goals, obstacles, and resolution tactics for simplifying modelling.

Table 3. An excerpt of probable obstacles obstructing the quality goal *system interoperability* along with some alternative resolution tactics and reference to the empirical studies

| Quality goal | Definition | Source |
|---|---|---|
| G6 | **Interoperability.** Various cloud services can be illimitably incorporated to and integrated with the system as required. | Genera literature on cloud computing (e.g. |

| | | |
|---|---|---|
| | | [S2], [S3], [S4],[S5], [S35], [S36], [S37] |
| **Obstacle** | **Definition** | **Source** |
| O19 | **Incompatible pluggable cloud services.** At runtime, system might be plugged to a cloud service which is incompatible with the other cloud services. | [S23] |
| O20 | **Incomplete APIs.** Cloud service provider lacks providing a rich set of APIs. | [S24] |
| O21 | **Incompatible data types.** Data types used in legacy and cloud service are incompatible. | [S12], [S38] |
| O22 | **Operating system incompatibility.** System components are distributed and moved among cloud servers with different operating systems which might be incompatible for managing, representing, and formatting virtual machines. | [S25], [S26], [S27] |
| O23 | **Machine-image incompatibility.** Virtual machines are moving between different cloud platforms but each platform has different underlying implementation for virtual machines. | [S39], [S40] |
| **Resolution tactic** | **Definition** | **Source** |
| T5 | **Refactor legacy source code.** Adapt the source code for being compatible and able to interact with the selected cloud platform programming language and APIs. | [S65], [S66], [S67] |
| T3 | **Develop adaptor/wrapper.** Add adaptors for resolving mismatches, occurring at runtime system execution, between legacy system components and cloud services. | [S75], [S76] |

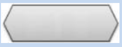
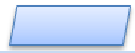
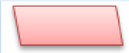
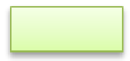
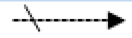
## 4.2 Establishing a procedure for goal-obstacle analysis

As mentioned earlier, this research uses KAOS modelling concepts to elicit, model, and reason about goals for migrating legacy systems to cloud platforms in a precise and systematic way. However, as KAOS modelling concepts are generic, we specialized these concepts by introducing 67 obstacles and 45 tactics. Table 4 presents KAOS modelling concepts used for representing cloud related goals, obstacles, and tactics.

Our KAOS-based procedure includes the following two steps: (i) *Set-up cloud migration goals* to set up and visualize high-level quality goals targeted for moving legacy systems to the cloud, (ii) *Analyse obstacles* comprising sub-steps for identifying obstacles causing goal failure, assessing their risk, and defining resolution tactics to modify existing goals, or generating new ones to prevent, to reduce, or to mitigate the obstacles. The output of the procedure provides for a system architect a consolidated requirement model in which cloud migration goals are elaborated with potential obstacles to be tackled, assumptions, alternative resolution tactics to eliminate, alleviate, or avoid obstacles. This model is later incorporated into the process of migrating or developing systems for cloud environments.

To illustrate the inner working of the procedure, an example scenario of moving the database of a legacy system to the cloud service Amazon Simple Storage (S3), i.e. migration type V is described. S3 is a public, secure, and highly scalable data storage providing data storing and retrieving any amount of data (AmazonS3).

Table 4. notations used for goal modelling

| Modelling element | Definition | Graphical notation |
|---|---|---|
| Migration type | An option through which legacy systems can benefit from cloud services to improve their working performance (See section 2). |  |
| Goal | A quality goal that is expected to be satisfied by utilizing cloud services. |  |
| Obstacle | A technical or a none-technical exceptional situation/condition preventing the goal satisfaction. |  |
| Resolution tactic | A generic solution to resolve an obstacle by introducing new goals, assumptions, or by modifying existing goals. |  |
| AND/OR | A mechanism to refine goals model to a set of fine-grain |  |

|  |  |  |
|---|---|---|
| Decomposition | goals/obstacles. |  |
| Contribution | Positive contribution a migration type to a quality goal. | —+→ |

### Step 1 Set-up cloud migration goals

The framework provides a collection of quality goals commonly intended in moving existing legacy systems to the cloud that a system architect and other stakeholders can use to initiate a goal model. In under study example, three goals are set for moving the legacy system database to S3 platform (Figure 3). This includes *Achieve [Reduced IT cost]*, *Achieve [Improved performance]*, and *Achieve [Improved availability]*. Goals can be refined into fine granular ones for more accurate analysis using AND/OR decomposition. An AND-refinement means that satisfying top goal depends on satisfying all sub-goals. OR-refinement presents a set of alternative goals in which satisfying top goal can be realized by satisfying one of alternative goals (Cailliau and van Lamsweerde 2013). Using AND-refinement mechanism, the goal *Achieve [Improved performance]* is a combination of sub-goals *Achieve [Reduced data uploading time]* and *Achieve [Reduced query processing time]* meaning that the satisfaction of *Achieve [Improved performance]* depends on the satisfaction of both these sub-goals. Figure 3 the dotted arrows show this fact that goals, obstacles, and resolution tactics are extracted from the repository.

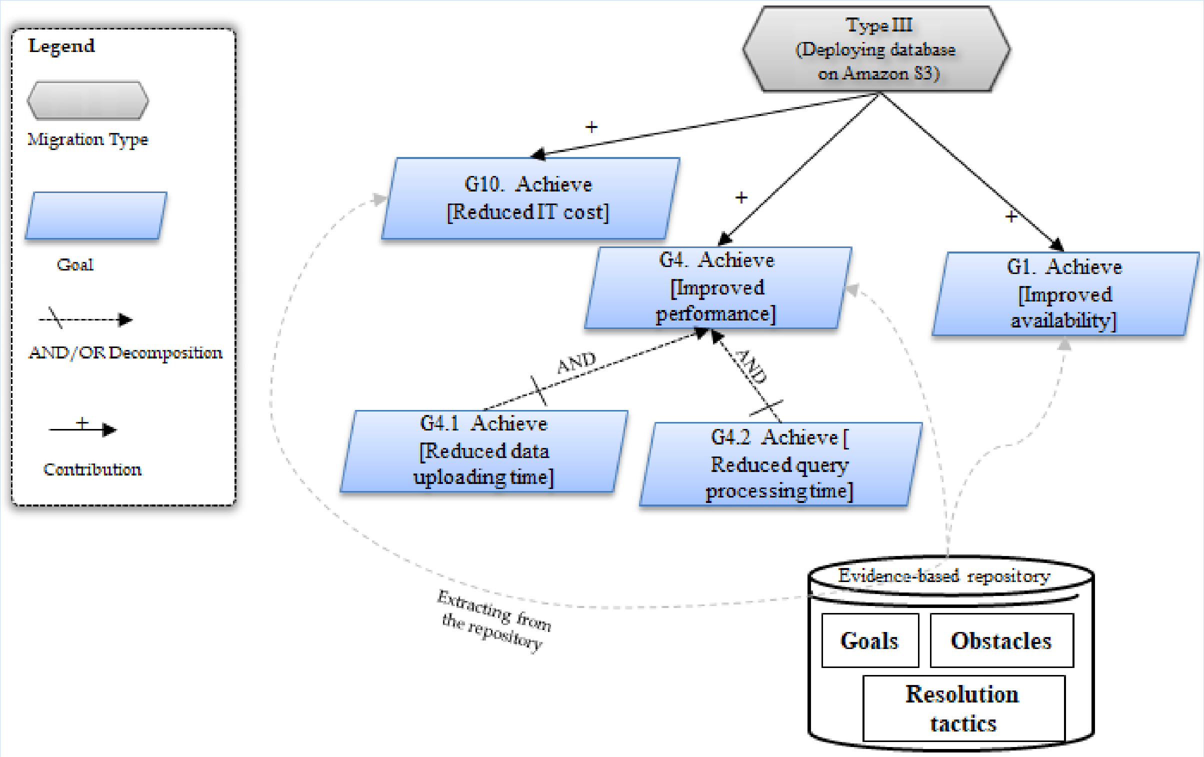


Figure 3. Goals in moving the legacy system database to Amazon S3

### Step 2 Analyse obstacles

Generally, goals are viewed idealistic and overlook unexpected behaviours of real environment may cause their failures (van Lamsweerde and Letier 2000; Letier 2001). Taking a pessimistic view to a goal model, such situations, referred to them as obstacle, should be systematically detected, assessed, and handled in the early stage of migration and if needed goals should be modified (Letier 2001). Obstacles are a dual notion to goals meaning that as goals capture desired conditions, obstacles capture undesirable conditions (Letier 2001). The framework defines an iterative identify-assess-resolve cycle as follow:

(i) Systematic identification of obstacles that may impede the satisfaction of goals (Step 2.1);
(ii) Assess risk of obstacles in terms of likelihood and criticality (Step 2.2); and
(iii) Resolve obstacles by modifying existing goals or generating new ones so as to prevent reduce or mitigate the obstacles (Step 2.3).

### *Step 2.1 Identify obstacles*

To refine a goal model to obstacles, the system architect can identify obstacles in two ways:

*(i)* *Evidential* where the probable obstacles are identified from the repository. For each quality goal, the system architect reviews the collection of obstacles and shortlists probable ones. This is mainly based on in on information provided by developers, user experience, and statistics about legacy systems, and available accounts about cloud services.

*(ii)* *Domain-based* where the obstacle is domain specific and in fact a refinement of an existing obstacle in the repository. Domain-specific obstacles are means to refine the goal model to new sub-obstacles.

Similar to goal elements, AND/OR operators are used to refine obstacles to sub-obstacles. AND-refinements includes a combination of sub-obstacles causing the parent obstacle whilst, OR-refinements result in a set of alternative sub-obstacles may occur separately (Letier 2001). Figure 4 shows an example of a generated goal model as the output of this step. The goal *Achieve [Reduced data uploading time]* is obstructed by the obstacles *Performance variability of Amazon S3 (O27)* and *Geographical distance (O28)*. The root obstacle *Performance variability of Amazon S3 (O27)*, which is suggested by the repository, is refined into two domain-specific obstacles *High uploading time for blobs datatype (100k entries) (O27_1)* and *Low throughput to write buckets (O27_2)*. Moreover, the goal *Achieve [Improved availability]* is obstructed by the obstacles *Service transient fault (O3)* and *Cloud outage (O1)*. The obstacle *Cloud outage (O1)*, by itself, is refined into three sub-obstacles *Local network disruption (O1_1)*, *I/O issues of servers (O1_2)*, and *S3 data centre outage (O1_3)* that are domain-specific refinements of the obstacle *Cloud outage (O1)* suggested by the repository. The domain specific obstacle *S3 data centre outage (O1_3)* is also refined into two obstacles *Local electrical storm (O1_3_1)* and *S3 power outage (O1_3_2)*.

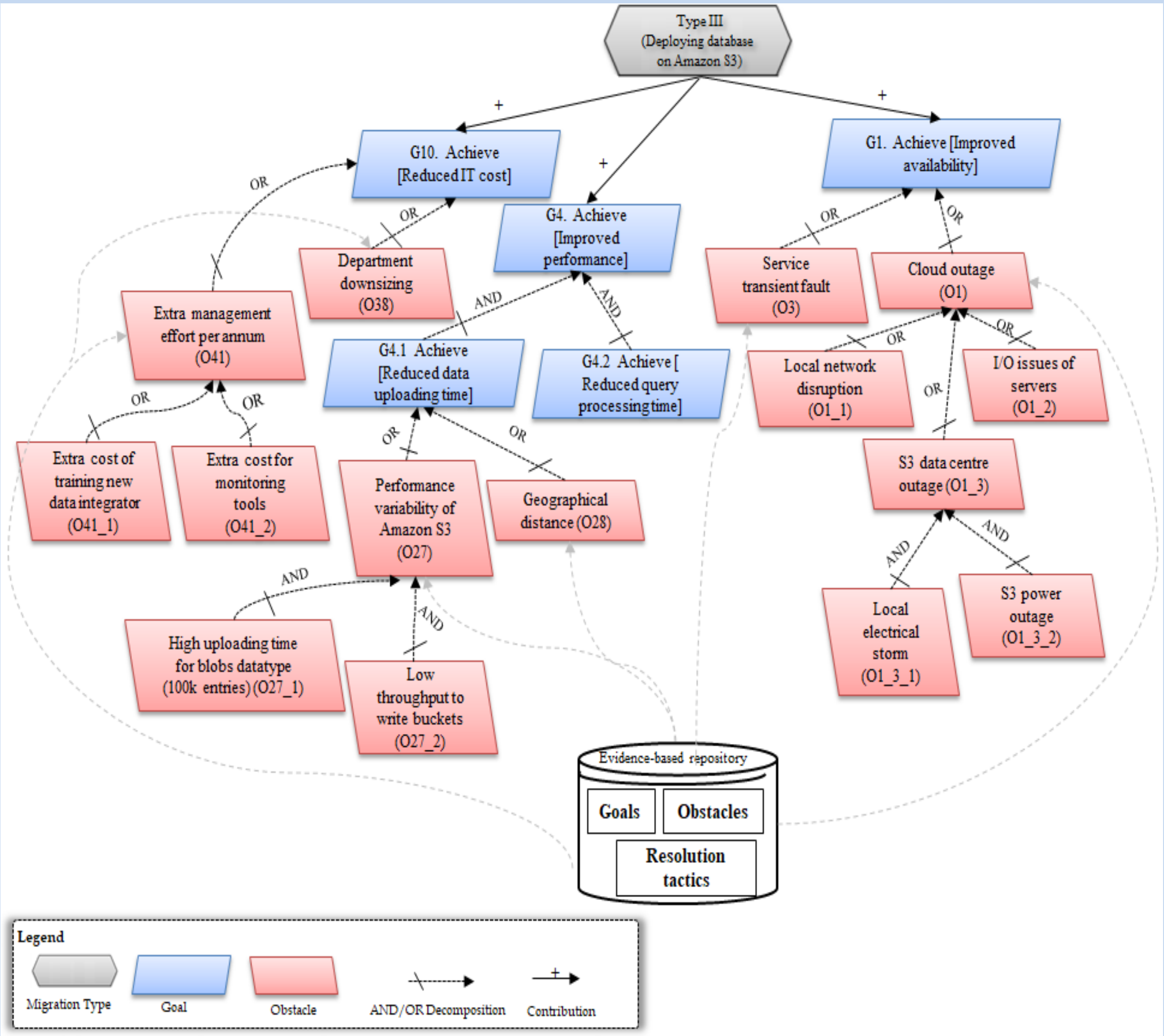

Type III (Deploying database on Amazon S3)
+
+
+
G10. Achieve [Reduced IT cost]
G4. Achieve [Improved performance]
G1. Achieve [Improved availability]
OR
OR
Department downsizing (O38)
Extra management effort per annum (O41)
AND
AND
G4.1 Achieve [Reduced data uploading time]
G4.2 Achieve [ Reduced query processing time]
OR
OR
Extra cost of training new data integrator (O41_1)
Extra cost for monitoring tools (O41_2)
OR
OR
Performance variability of Amazon S3 (O27)
Geographical distance (O28)
AND
AND
High uploading time for blobs datatype (100k entries) (O27_1)
Low throughput to write buckets (O27_2)
OR
OR
Service transient fault (O3)
Cloud outage (O1)
OR
OR
OR
Local network disruption (O1_1)
I/O issues of servers (O1_2)
S3 data centre outage (O1_3)
AND
AND
Local electrical storm (O1_3_1)
S3 power outage (O1_3_2)
Evidence-based repository
Goals
Obstacles
Resolution tactics
Legend
Migration Type
Goal
Obstacle
AND/OR Decomposition
Contribution

Figure 4. Obstacles against achieving quality goals *Achieve [Reduced IT cost]*, *Achieve [Improved performance]*, and *Achieve [Improved availability]* retrieved from the repository

### *Step 2.2 Assess Obstacles*

Analysing the risk or criticality of obstacles identified in Step 2.1 is important to get an understanding of requirements for making a legacy system cloud enabled. The framework uses the standard qualitative technique Risk Analysis Matrix (RAM) devised by the acquisition reengineering team at the Air Force Electronic System Centre (Franklin 1996). The qualitative expression of obstacle risks in RAM is suitable when precise numerical methods are not difficult to find or not required. In this technique the likelihood of an obstacle is judged by qualitative scales from *Almost Certain*, *Likely*, *Possible*, *Unlikely*, and *Rare* and the consequence of the occurrence of an obstacle is represented by *Insignificant*, *Minor*, *Moderate*, *Major*, and *Catastrophic*. The risk of an obstacle is the direct product of its probability of occurrence and the severity (i.e. Risk = Likelihood × Consequences). Taking these qualitative scales to measure the likelihood of obstacle occurrence and its consequence into account, a risk matrix can be created that highlights the risk zone as shown in Table 5. An organization may define zones like generally unacceptable, acceptable, and low-risk. For example, the risk of an obstacle might be perceived as moderate (M) but it is still tolerable whilst an obstacle with H (High) and E (Extreme) should be handled carefully.

Note that, the product of likelihood and consequence of obstacles in this example table relies on domain information sources such as the specification of cloud service providers, statistics from legacy systems, developers, end-users' experience, and an overall impact of risks on goals. The system architect may use a voting procedure involving all stakeholders to assign correct qualitative values to the occurrence likelihood and their consequences. Hence, the values in the table below should be computed regarding the domain information in a goal-obstacle analysis scenario.

Table 5. risk matrix for obstacles

| | Consequence severity | | | | |
|---|---|---|---|---|---|
| Likelihood | Insignificant | Minor | Moderate | Major | Catastrophic |
| Almost Certain | H | H | E | E | V |
| Likely | M | H | H | E | V |
| Possible | L | M | H | E | E |
| Unlikely | L | L | M | H | E |
| Rare | L | L | M | H | H |

V: Very extreme risk, E: Extreme risk; H: High risk; M: Moderate risk; L: Low risk

### *Step 2.3 Resolve Goal Obstacles*

Obstacles whose risks are recognized serious enough, (e.g. very extreme, extreme, and high risk) must be tackled (van Lamsweerde and Letier 2000; Letier and Van Lamsweerde 2004). The framework relies on the knowledge repository that provides a catalogue of resolution tactics to address obstacles identified in the previous step. Resolution tactics are platform agnostic but varying among seven categories: namely *Goal/Service/Migration type Substitution*, *Obstacle prevention*, *Obstacle reduction*, *Goal weakening*, *Goal restoration*, *Goal mitigation*, and *Do nothing* (Appendix B). Resolution tactics are platform agnostic to give system developers freedom to have a broad ranges of techniques to operationalize them. Their full definitions are presented in Appendix B. In this superficial example, all the obstacles are assumed sever and thus the goal model is refined by resolution tactics used to handle the obstacles (Figure 5). For instance, to reduce the occurrence likelihood of obstacle *Geographical distance (O28)*, the system architect chooses the resolution tactics *Refine network topology (T24)* from the repository. Another example is the reducing the risk of obstacles *High uploading time for blobs (100k entries) (O27_1)* and *Low throughput to write buckets (O27_2)* through incorporating the resolution tactic *Use multiple cloud servers (T27)* in the new architecture of the legacy system.

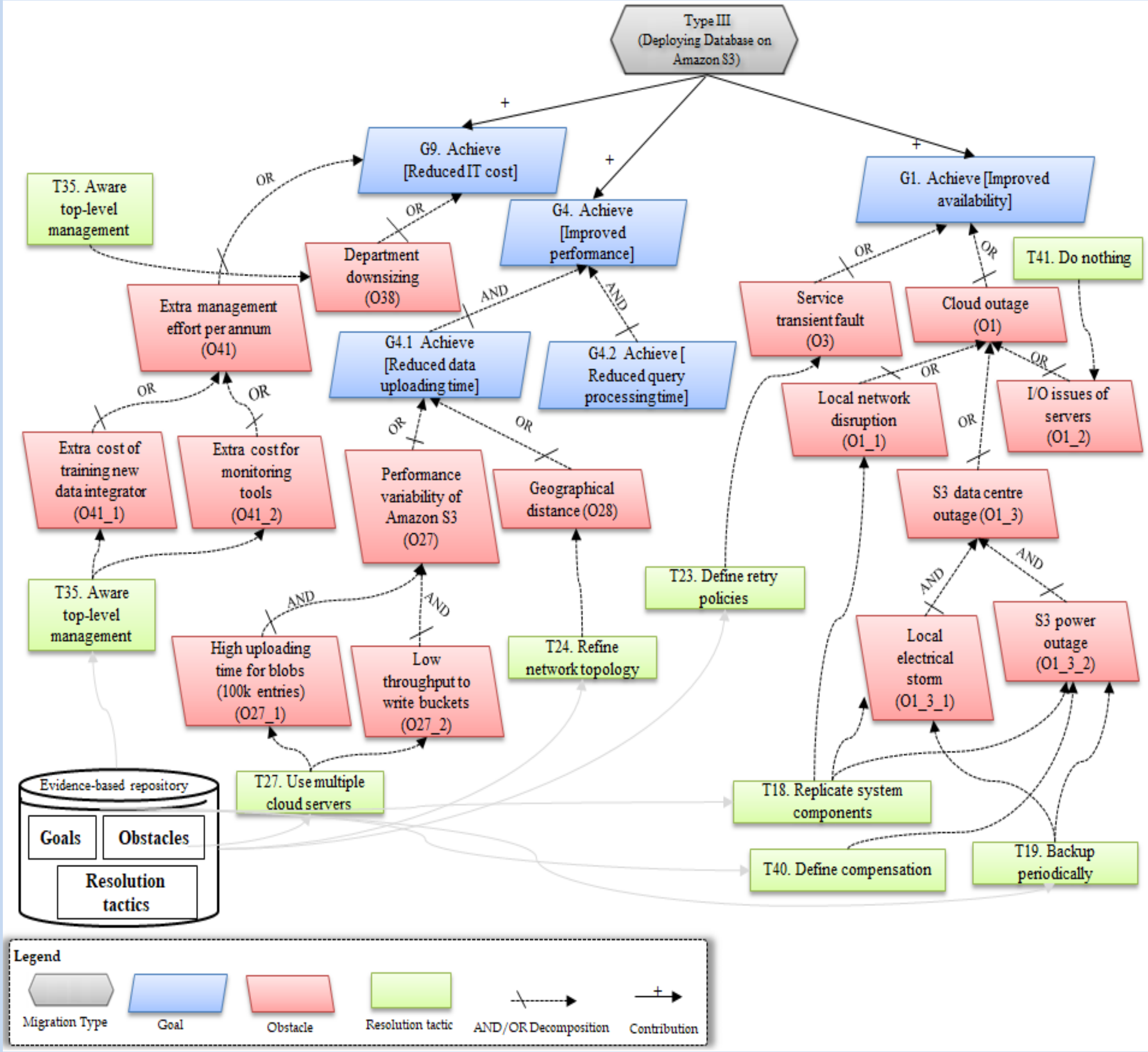

Type III (Deploying Database on Amazon S3)
+
+
+
G9. Achieve [Reduced IT cost]
G4. Achieve [Improved performance]
G1. Achieve [Improved availability]
T35. Aware top-level management
OR
OR
Department downsizing (O38)
Extra management effort per annum (O41)
AND
AND
G4.1 Achieve [Reduced data uploading time]
G4.2 Achieve [ Reduced query processing time]
OR
Service transient fault (O3)
Cloud outage (O1)
OR
T41. Do nothing
OR
OR
OR
Extra cost of training new data integrator (O41_1)
Extra cost for monitoring tools (O41_2)
OR
OR
Performance variability of Amazon S3 (O27)
Geographical distance (O28)
Local network disruption (O1_1)
OR
I/O issues of servers (O1_2)
OR
S3 data centre outage (O1_3)
T35. Aware top-level management
T23. Define retry policies
AND
AND
AND
AND
High uploading time for blobs (100k entries) (O27_1)
Low throughput to write buckets (O27_2)
T24. Refine network topology
Local electrical storm (O1_3_1)
S3 power outage (O1_3_2)
Evidence-based repository
Goals
Obstacles
Resolution tactics
T27. Use multiple cloud servers
T18. Replicate system components
T40. Define compensation
T19. Backup periodically
Legend
Migration Type
Goal
Obstacle
Resolution tactic
AND/OR Decomposition
Contribution
+

Figure 5. Resolution tactics to tackle obstacles

## 5. Application of the framework in practice

This section presents two case studies as a benchmark for the validation of the framework. The first case was a scenario of moving an open-source Web-based system providing real-time stock quotes for users to a private cloud platform. The second scenario was about moving a Web-based system for processing digital documents to a public cloud. The system architect used domain information related to the migration scenario as the main source to selected and shortlisted the pre-constructed collection of obstacles and resolution tactics. In both scenarios, the risk matrix values presented in Table 5 were used to assess obstacle risk.

### 5.1 Case study 1

*SpringTrader* (Gordon 2015) is an open-source Web-based system that enables users to create account and browse stock portfolio, lookup stock quotes, and buy and sell stock shares. It has been developed using J2EE framework and maintained by many contributors over time. *SpringTrader* architecture includes (i) Web-based layer enable users for browsing stock quotes and portfolios, lookup stock quotes, and ordering stock trade orders and (ii) a backend that fulfils orders. The communication between the web-based frontend and the backend is a-synchronous where the front-end delivering orders to a message queue and the back-end processing them.

Moving *SpringTrader* to Pivotal Cloud Foundry which is an open source platform for developing and deploying full stack software systems in the cloud would enable users to utilize this powerful cloud platform to access real-time stock market data in a more interactive way with the system as well as the individual scaling up/down of system components. The system architect was interested in analysing architectural requirements in enabling *SpringTrader* to operate in this platform. The documentation of this project is available at (Gordon 2015).

### *Step 1 Set-up cloud migration goals*

A goal model was created with the three initial goals *Achieve [Increased scalability]*, *Achieve [Keeping system interoperable]*, and *Achieve [Keeping system available]* with the following specifications:

> **Goal** Achieve [Increased scalability]
> **Definition** [Moving the *SpringTrader* to the cloud should make it scalable in the sense that the system should be able to service massive end users' requests during workload]
>
> **Goal** Achieve [Keeping system interoperable]
> **Definition** [The *SpringTrader* should be integratable with and be able to call cloud services]
>
> **Goal** Achieve [Keeping system availability]
> **Definition** [Moving the *SpringTrader* to the cloud should not affect the system availability to end users]

### *Step 2 Analyse obstacles*

**Step 2.1 Identify obstacles**. Reviewing the architecture model of *SpringTrader* revealed that the tight dependencies among system component impede their individual scalability and portability across multiple instances of servers. This was an instance of the obstacle *Tight dependencies (O51)* against the goal *Achieve [Increased scalability]*. For new platform, it was planned to use cloud database solutions such as MySQL and MongoDB for the *SpringTrader*. However, they were incompatible with the SQL database of *SpringTrader*. This was indeed an obstacle to the goal *Achieve [Keeping system interoperable]*, an instantiation of the root obstacle *Incompatibility of legacy data storage and cloud (O49)* defined in the repository. This obstacle, by itself, was refined into two obstacles *Incompatible data operations (O50)* and *Incompatible data types (O21)*. Another obstacle to the goal *Achieve [Keeping system interoperable]* was that *SpringTrader* had been implemented using Java Development Kit 6 and Spring 3 that accordingly are not compatible with their equivalent (i.e. Java Development Kit 8 and Spring 4) in the Cloud Foundry platform. Integrating the *SpringTrader* with the Quote Web-Service, a service using the public Yahoo Finance APIs to provide real-time market data, would raise the risk of service unavailability as this service is hosted on the Cloud Foundry servers and geographically out of the local network of *SpringTrader*. This domain information confirmed that the obstacle *Service transient fault (O3)* would likely to occur against the goal *Achieve [Keeping system availability]*. The goal model was updated with new four obstacles shown in Figure 6.

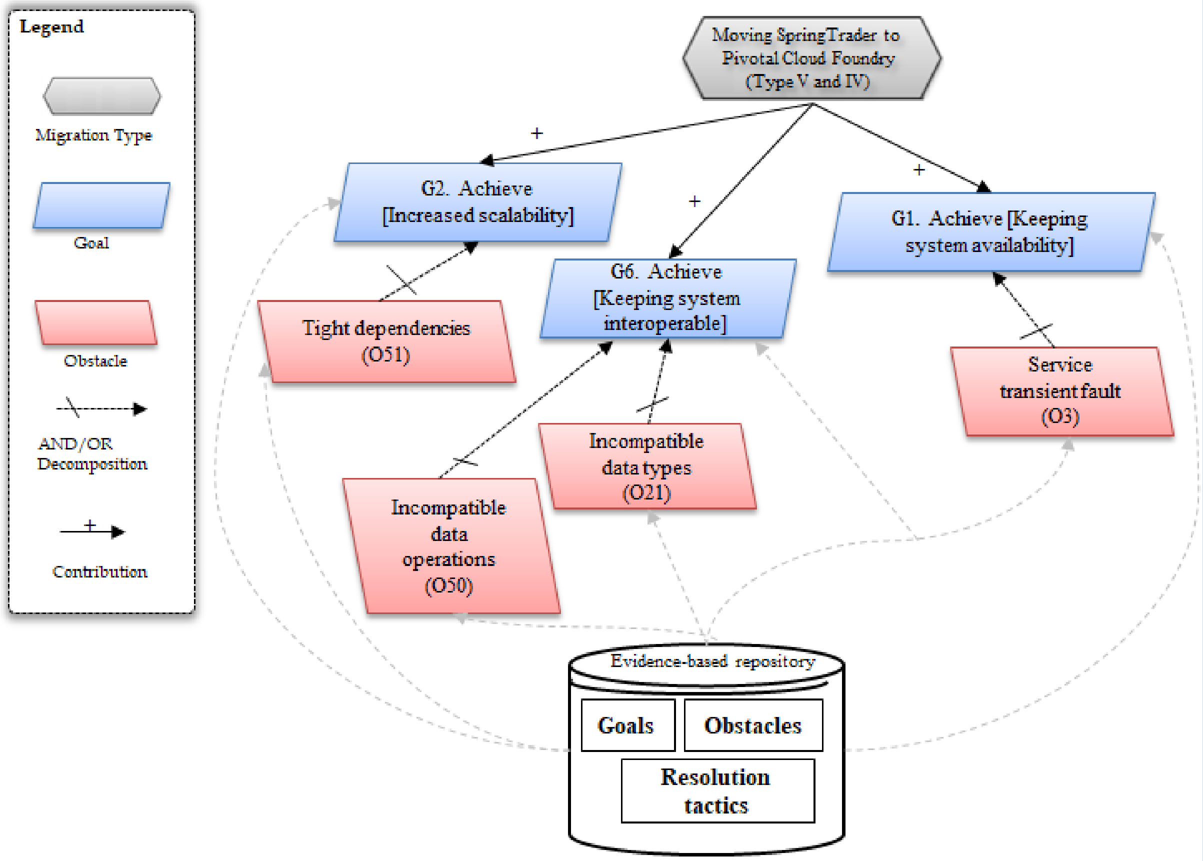

Legend
Migration Type
Goal
Obstacle
AND/OR Decomposition
Contribution
Moving SpringTrader to Pivotal Cloud Foundry (Type V and IV)
+
G2. Achieve [Increased scalability]
G6. Achieve [Keeping system interoperable]
G1. Achieve [Keeping system availability]
Tight dependencies (O51)
Service transient fault (O3)
Incompatible data types (O21)
Incompatible data operations (O50)
Evidence-based repository
Goals
Obstacles
Resolution tactics

Figure 6. Goals *Achieve [Increased scalability]*, *Achieve [Keeping system interoperable]*, and *Achieve [Keeping system availability]* refined to four obstacles informed by the framework repository

**Step 2.2 Assess obstacles.** The occurrence probability and the consequence of the obstacles identified from step 2.1 were assessed according to the domain information such as developers and end users' opinions and statistics about *SpringTrader*. Table 6 shows the risk matrix of obstacles. The goal *Achieve [Increased scalability]* was refined to the obstacle *Tight dependencies (O51)* as shown in Figure 7. Based on the architecture model of legacy system, the occurrence likelihood of this obstacle is *Almost Certain*. This was also true for the obstacles *Incompatible data operations (O50)* and *Incompatible data types (O21)* since *SpringTrader* database was incompatible with the Pivotal Cloud Foundry platform.

The occurrence likelihood of obstacle *Service transient fault (O3)* was found *Possible* as it could be situations in which *SpringTrader* components could not successfully call Quote Web-Service in the first attempt due to transient faults in making network connection to Quote Web-Service hosted in servers in Pivotal Cloud Foundry platform. Although it was a violation from the goal *Achieve [Keeping system available]*, its consequence was believed *Minor*. Therefore, the risk of this obstacle was set *Low*.

Table 6. Risk matrix for the obstacles identified from Step 2.1

| Obstacle against quality goal | Likelihood | Consequence | Risk |
|---|---|---|---|
| *Tight dependencies (O51)*. *SpringTrader* components have tight dependencies to meta-libraries that are sometimes dependencies that are incompatible with JDK 8. This cause the component of the system cannot be scalable and portable across multiple instances of servers. | Almost Certain | Major | E |
| *Incompatible data operations (O50)*. Various SQL statements in *SpringTrader* related to manipulating records are not syntactically and semantically compatible with corresponding MongoDB statements and MySQL provided by Pivotal Cloud Foundry platform. | Almost Certain | Major | E |
| *Incompatible data types (O21)*. In some cases data types (e.g. length and format) used in *SpringTrader* database are not compatible with corresponding ones in MySQL and MongoDB. | Almost Certain | Major | E |

| | | | |
|---|---|---|---|
| *Service transient fault (O3)*. Quote Web-Service might be temporarily unavailable for reasons such as network traffic load or server workload. | Possible | Minor | L |

**Step 2.3 Resolve goal obstacles.** The system architect explored the repository to find the catalogue of resolution tactics handling obstacles for incorporation into new architecture of *SpringTrader* reengineered to operate in Pivotal Cloud Foundry platform. The system architect selected the resolution tactic *Decouple system components (T7)* from the category *Obstacle prevention* (see Appendix B) to remove obstacle *Tight dependencies (O51)*. The tactic used to operationalise was implementing a mediator and synchronisation mechanisms to manage interaction between the system components each deployed in different servers of Pivotal Cloud Foundry platform. For the obstacles *Incompatible data operations (O50)* and *Incompatible data types (O21)* the architect used the tactics *Adapt data (T12)* and *Develop adaptor/wrapper (T6)*, respectively. The former is to convert data types of *SpringTrader* into the data type of database solutions, i.e. MySQL and MongoDB, offered by the Pivotal Cloud Foundry platform whilst the latter is to add adaptors/wrappers that are responsible for runtime conversion of *SpringTrader* operations into the Pivotal Cloud Foundry.

To reduce the probability occurrence of the obstacle *Service transient fault (O3)*, the adopted resolution tactic was *Define retry policies (T23)* which is subsumed under the group *Goal Restoration*. That is, implementing a retry policy in the architecture of *SpringTrader* to specify the delay before executing the next attempt for connecting to the Pivotal Cloud Foundry server when transient faults occur due to network congestion. In addition, the system architect chose the tactic *Replicate system components (T18)* from the group *Obstacle Prevention* group. The tactic is to partition, replicate, and distribute components and data (replicas) of *SpringTrader* over multiple servers of Pivotal Cloud Foundry.

Resolution tactics defined in the framework repository are generic recurrent solutions that can be operationalized using different implementation techniques or tools available in the cloud computing marketplace. For example, in this scenario the resolution tactic *Develop adaptor/wrapper (T6)*, addressing incompatibilities between a legacy system database and a cloud database solution, is operationalized using the notion of bounded context (Thönes 2015) in the sense that the transition of data is packed and unpacked during the executing of transactions. As another example, to realize the tactic *Decouple system components (T7)*, developers used *micro-service architecture design* (Dragoni, Giallorenzo et al. 2016) along with a service discovery mechanism to enable *SpringTrader* to locate micro services by name at a known catalogue endpoint and look them up dynamically at runtime. Figure 7 shows the resolution tactics selected in this scenario.

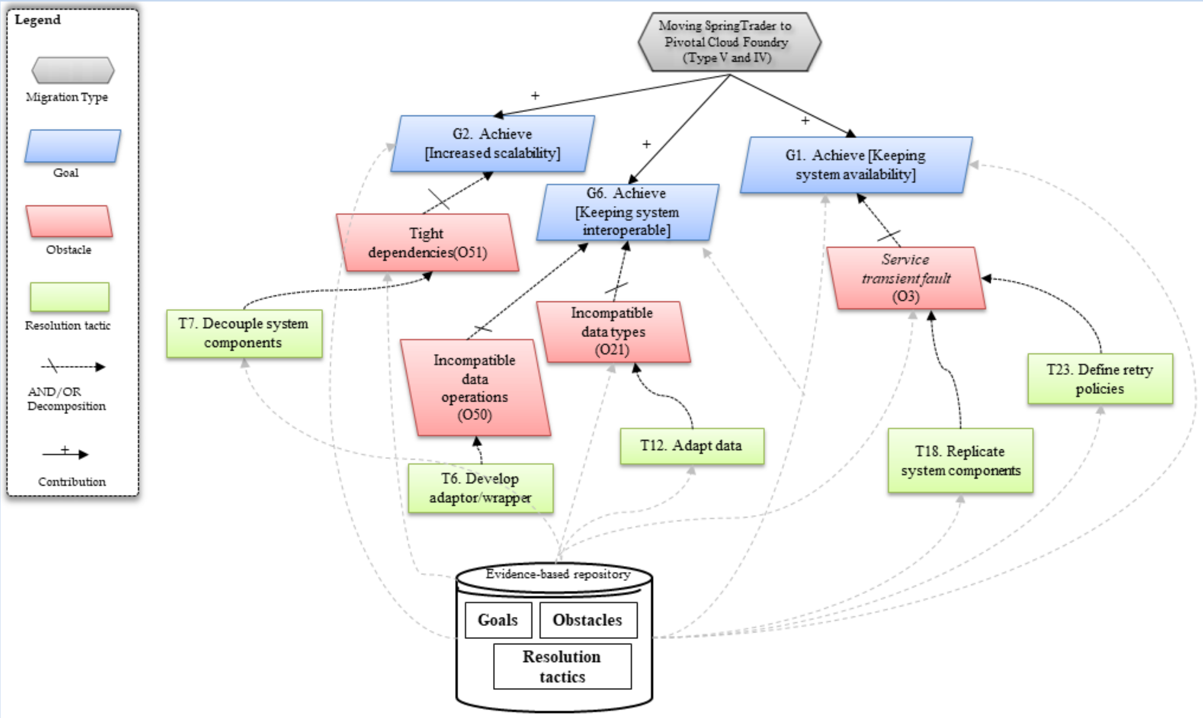

Legend
Migration Type
Goal
Obstacle
Resolution tactic
AND/OR Decomposition
Contribution
Moving SpringTrader to Pivotal Cloud Foundry (Type V and IV)
+
+
+
G2. Achieve [Increased scalability]
G6. Achieve [Keeping system interoperable]
G1. Achieve [Keeping system availability]
Tight dependencies(O51)
Service transient fault (O3)
T7. Decouple system components
Incompatible data types (O21)
Incompatible data operations (O50)
T23. Define retry policies
T12. Adapt data
T18. Replicate system components
T6. Develop adaptor/wrapper
Evidence-based repository
Goals
Obstacles
Resolution tactics

Figure 7.Resolutions tactics for handling obstacles

In Table 7, the first and second columns, respectively, show the resolution tactics and their operationalisation techniques used to handle obstacles presented in third column.

Table 7. Resolution tactics to handle obstacles in migrating *SpringTrader* to Pivotal Cloud Foundry

| Resolution tactic | Operationalisation | Relation to obstacle |
|---|---|---|
| Decouple system components (T7) | Decouple the *SpringTrader* components from each other by using mediator enabling a- synchronised interaction among loosely coupled components in deployed on distributed architecture of Pivotal Cloud Foundry. | Tight dependencies (O51) |
| Develop adaptor/wrapper(T6) | Develop adaptor component in *SpringTrader* to emulate operations are supported in MySQL and MongoDB and map mismatches between datatypes in *SpringTrader* and Pivotal Cloud Foundry. | Incompatible data operations (O50) |
| Adapt data (T12) | Implement a mapping table to convert incompatible data types in *SpringTrader* and MySQL and MongoDB. | Incompatible data types (O21) |
| Replicate system components (T18) | Partition, replicate, and distribute the components of *SpringTrader* on multiple servers of Pivotal Cloud Foundry. | Service transient fault (O3) |
| Define retry policies (T23) | Implement retry policies in the source code of *SpringTrader* to specify the delay before executing the next attempt when Pivotal Cloud Foundry does not respond. | |

### 5.2 Case study 2

The second case is adapted from the scenario presented in (Rabetski 2012; Rabetski and Schneider 2013). *InformIT* is a small independent software vendor in Sweden providing a Web-based digital document processing (DDP) legacy system offers publishing services to medium and large companies who have adequate infrastructure to perform these resource-demanding services. While small companies were interested in taking the advantages of DDP's services, they could not afford the financial commitment to procure new infrastructure, charging per user and installation to use DDP. *InformIT* believed that DDP's services could be also used by small companies without a need for upgrading infrastructure if they would deployed and run on the cloud via migration type V. The history of goal-obstacle analysis conducted by system architecture regarding reengineering DDP described in the following.

### *Step 1 Setting-up cloud adoption goals*

The cloud enablement scenario should not exceed 90 days. This is represented via the goal *Achieve [Reduced cloud adoption cost]* and its specification is:

> **Goal** Achieve [Reduced cloud adoption cost]
> **Definition** [According to *InformIT* policy, the latest completion time for any new technology adoption for small companies should not exceed more than 90 days. In this scenario, moving the DDP to the cloud should be fulfilled with minimum development effort].

In addition to the above, goals *Achieve [Improved performance]*, *Achieve [Improved testability]*, and *Achieve [Improved portability]* were set to be satisfied by moving the DDP to a cloud platform. For example, the goal *Achieve [Improved performance]* is defined:

> **Goal** Achieve [Improved performance]
> **Category** Performance Goal
> **Definition** [acceptable system throughput for rendering a digital document with any size should be no more than 4.9 seconds].

### *Step 2 Analyse obstacles*

**Step 2.1 Identify obstacles.** Scanning the framework repository in view of domain information refined the top goals towards root obstacles and subsequently leaf ones. The result shown in Figure 8. For example, there were two probable obstacles *Learning curve (O33)* and *Incompatibility of legacy and cloud service (O48)* against the satisfaction of the system goal *Achieve [Reduced cloud adoption cost]*. Moreover, experience of developers confirmed that the goal *Achieve [Improved performance]* might be obstructed by the performance variability of cloud servers once DDP is in operation on these servers. This is shown by the obstacle *Performance variability of cloud server (O27)* in the goal model (Figure 8).

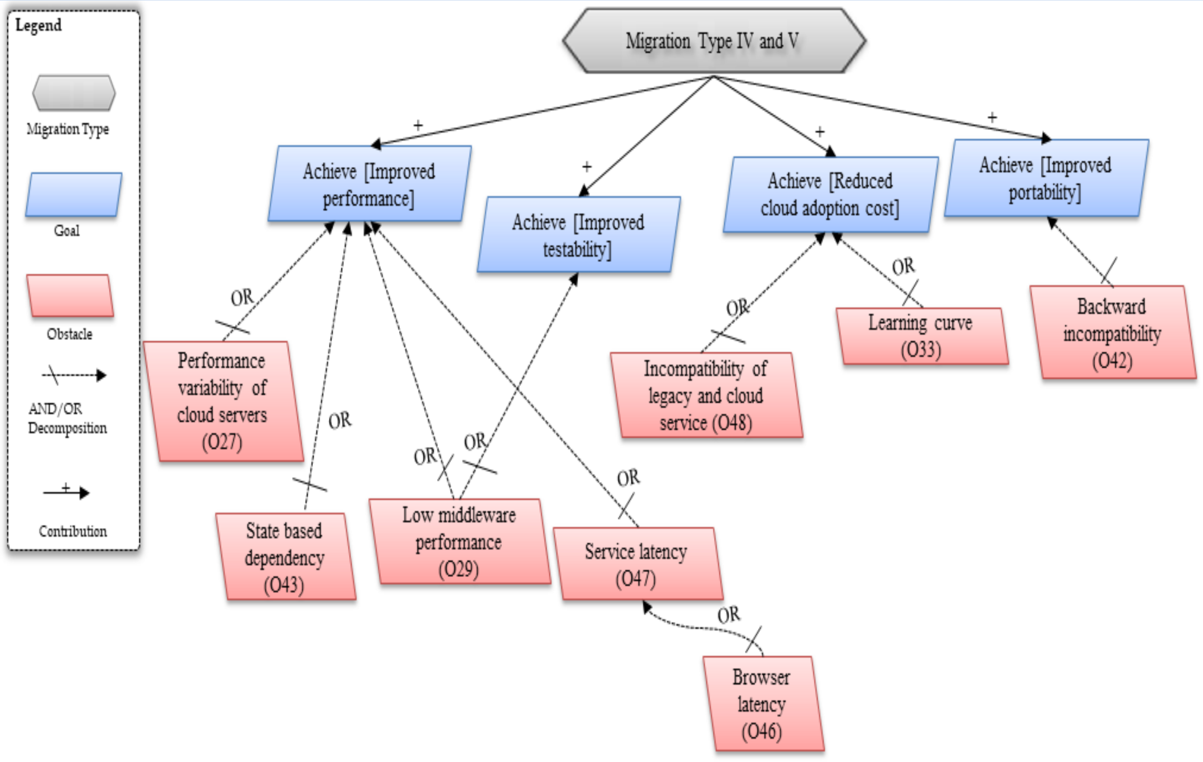

Legend
Migration Type
Goal
Obstacle
AND/OR Decomposition
Contribution
Migration Type IV and V
+
+
+
+
Achieve [Improved performance]
Achieve [Improved testability]
Achieve [Reduced cloud adoption cost]
Achieve [Improved portability]
OR
OR
OR
OR
OR
OR
OR
OR
OR
Performance variability of cloud servers (O27)
State based dependency (O43)
Low middleware performance (O29)
Service latency (O47)
Browser latency (O46)
Incompatibility of legacy and cloud service (O48)
Learning curve (O33)
Backward incompatibility (O42)

Figure 8 Refined the goals *Achieve [Improved performance]*, *Achieve [Improved testability]*, *Achieve [Reduced cloud adoption cost]*, *Achieve [Improved portability]* to leaf obstacles

**Step 2.2 Assess obstacles**. Technical documents of DDP and an early investigation of public cloud platforms revealed the ocurrence of obstacles *Learning curve (O33)* and *Incompatibility of legacy and cloud service (O48)* was *Almost Certain* with a *Major* consequence on the satisfaction of the goal

*Achieve [Reduced cloud adoption cost]*, indicating an *Extreme* risk. All leaf obstacles were assigned a risk value based on the likelihood of their occurrence and consequence as shown in Table 8.

Table 8. Risk matrix for the obstacles identified from Step 2.1

| Obstacle | Likelihood | Consequence | Risk |
|---|---|---|---|
| Performance variability of cloud servers (O27) | Likely | Major | E |
| State based dependency (O29) | Likely | Moderate | H |
| Low middleware performance (O29) | Likely | Moderate | H |
| Browser latency (O46) | Likely | Moderate | H |
| Incompatibility of legacy and cloud service (O48) | Almost Certain | Major | E |
| Learning curve (O33) | Almost Certain | Major | E |
| Backward incompatibility (O42) | Likely | Moderate | H |

### Step 2.3 Resolve goal obstacles.

The system architect tried to tackle obstacles *Learning curve (O33)* and *Incompatibility of legacy and cloud service (O48)* by using the resolution tactics *Substitute cloud service (T3)* and *Goal weakening (T36)*. *Substitute cloud service (T3)* is to resolve an obstacle by selecting/changing a cloud service/provider in a way that the new selected cloud service can still contribute to quality goals. As DDP had been developed with Microsoft family technologies and developers had programming experience of, choosing Microsoft Azure cloud platform was taken precedence over other popular cloud platforms such as Amazon Web Service and Google App Engine. This choice could also contribute to the goal *Achieve [Reduced cloud adoption cost]* by decreasing the likelihood occurrence of incompatibilities between DDP and Microsoft Azure cloud platform from initial value *Almost Certain* to *Possible*.

In some cases that an obstructed goal is found to be very idealistic, its definition can be changed to make its constraints relaxing in a way that the obstruction occurrence become tolerable. In this regard, the tactic *Degrade goal (T36)* was used by for the goal *Achieve [Reduced cloud adoption cost]* by extending the project deadline from 90 to 120 days. Figure 9 shows the produced goal model so far.

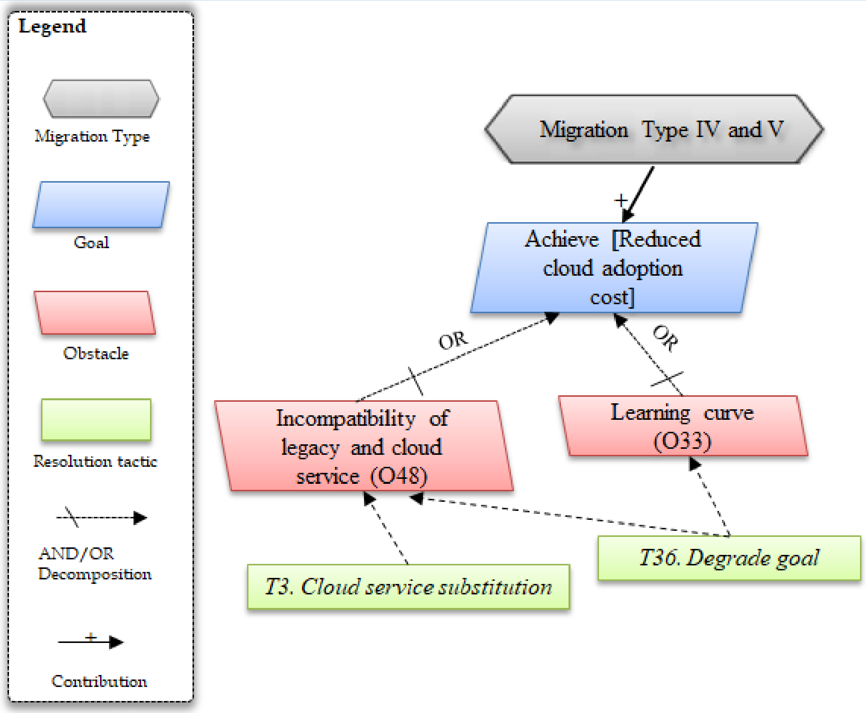


Figure 9 Obstacles to the goal *Achieve [Reduced cloud adoption cost]* and applied resolution tactics *Substitute cloud service (T3)* and *Degrade goal (T36)*

The obstacle resolution is an iterative process in the sense that once a tactic is chosen, it may raise new obstacles that should be resolved accordingly by reiterating steps 2.1 to 2.3 and refining the goal model. In the current scenario, despite applying the resolution tactic *Substitute cloud service (T3)* to reduce the obstacle *Incompatibility of legacy and cloud service (O48)*, the domain information about DDP and cloud platform Microsoft Azure documentation confirms that the violation from the goal *Achieve [Reduced cloud adoption cost]* is still possible because DDP's APIs are not compatible with their counterparts in the Microsoft Azure cloud platform. The obstacle *Incompatibility of legacy and cloud service (O48)* is refined into two obstacles *Incompatible APIs (O44)* (i.e. between DDP and Microsoft Azure platform) and *Incompatibility of legacy data storage and cloud (O49)*. The parent obstacle *Incompatibility of legacy data storage and cloud (O49)* is also refined into two leaf domain specific obstacles (Figure 10). The definition of the leaf obstacles against the goal *Achieve [Reduced adoption cost]* is as follows:

**Obstacle** *Incompatible APIs (O44)*
**Definition** [DDP uses API's offered by .NET 2.0 and Visual Studio 2005 which may not be compatible with Microsoft Azure platforms].

**Obstacle** *Incompatible datatypes (O21)*
**Definition** [Datatypes in DDP are based on SQL Server Database .NET 2.0 platform which might not be compatible with Microsoft Azure database solution].

**Obstacle** *Incompatible data operations (O50)*
**Definition** [Data operations in DDP supported by SQL Server Database .NET 2.0 platform might not be compatible with Microsoft Azure database solution].

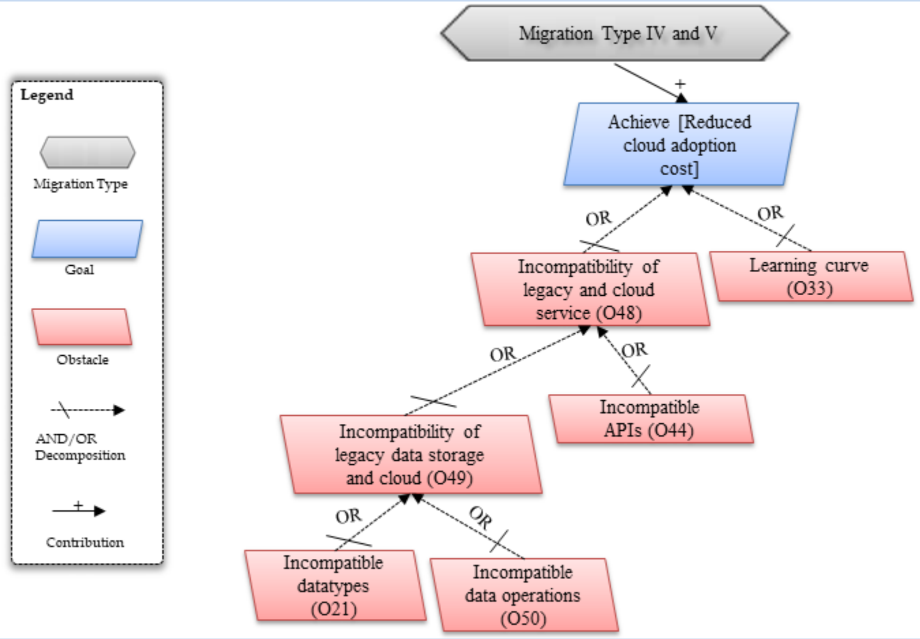

Migration Type IV and V
+
Legend
Migration Type
Goal
Obstacle
AND/OR Decomposition
Contribution
Achieve [Reduced cloud adoption cost]
OR
OR
Incompatibility of legacy and cloud service (O48)
Learning curve (O33)
OR
OR
Incompatible APIs (O44)
Incompatibility of legacy data storage and cloud (O49)
OR
OR
Incompatible datatypes (O21)
Incompatible data operations (O50)

Figure 10 Refinement of goal *Achieve [Reduced IT cost]* to obstacles

In addition to abovementioned obstacles, applying tactic *Substitute cloud service (T3)* generated new obstacles specific to Microsoft Azure Platform. That is, the root obstacle *Microsoft Azure middleware latency (O29)* was refined into three leaf obstacles, *Microsoft Azure database middleware latency (O29_1)*, *Microsoft Azure message middleware latency (O29_2)*, and *Microsoft Azure transaction middleware latency (O29_3)*. In addition, the obstacle *Service latency (O47)* was refined into two obstacles *On-premise hardware latency (O47_1)* and *Distance from Microsoft Azure servers (O28)*. Figure 11 shows goal model refined after using the resolution tactic *Substitute cloud service (T3)* and based on evidential information provided from the repository. For simplicity, the system architect has changed the original names of some obstacles identified from the repository but obstacle codes left unchanged. For example, in Figure 11, the obstacle *Backward incompatibility (O42)* was changed to *Switch between regular file system API to Microsoft Azure Storage API (O42)*.

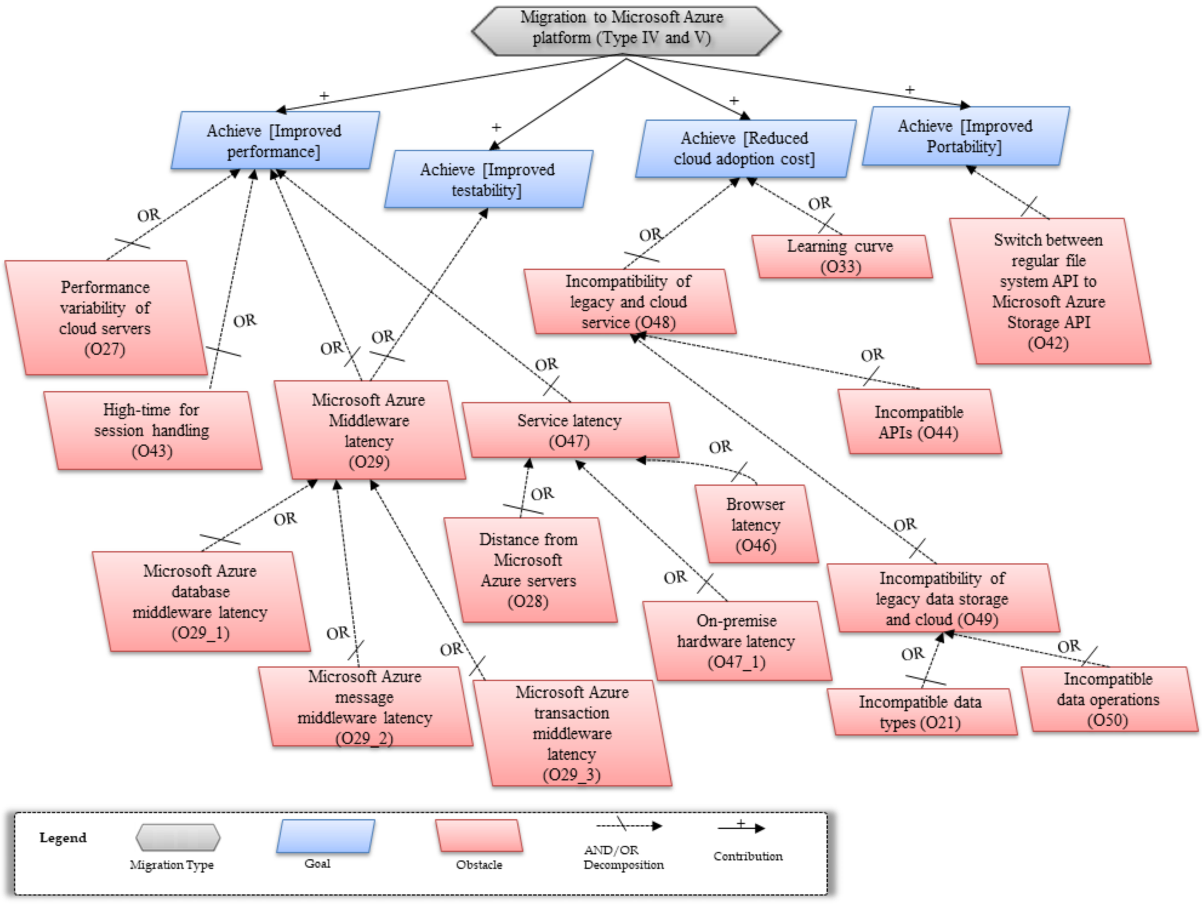

Migration to Microsoft Azure platform (Type IV and V)
+
+
+
+
Achieve [Improved performance]
Achieve [Improved testability]
Achieve [Reduced cloud adoption cost]
Achieve [Improved Portability]
OR
Performance variability of cloud servers (O27)
OR
High-time for session handling (O43)
OR
OR
Microsoft Azure Middleware latency (O29)
OR
Service latency (O47)
OR
Incompatibility of legacy and cloud service (O48)
OR
Learning curve (O33)
Switch between regular file system API to Microsoft Azure Storage API (O42)
OR
Incompatible APIs (O44)
OR
Browser latency (O46)
OR
Distance from Microsoft Azure servers (O28)
OR
Microsoft Azure database middleware latency (O29_1)
OR
Microsoft Azure message middleware latency (O29_2)
OR
Microsoft Azure transaction middleware latency (O29_3)
OR
On-premise hardware latency (O47_1)
OR
Incompatibility of legacy data storage and cloud (O49)
OR
Incompatible data types (O21)
OR
Incompatible data operations (O50)
Legend
Migration Type
Goal
Obstacle
AND/OR Decomposition
+
Contribution

Figure 11 Refined goals models after identifying obstacles

To handle new obstacles generated as a result of applying the tactic *Substitute cloud service (T3)*, the system architect selected resolution tactics from the category *Obstacle prevention* (Appendix B) as described in the following. For the obstacles *Incompatible data operations (O50)* and *Incompatible datatypes (O21)*, the system architect used *Develop adaptor (T6)* and *Adapt data (T12)*, respectively. The former is to implement a wrapper component which hides the incompatibilities (e.g. queries and stored procedures) between the data layer of DDP and Microsoft Azure SQL whilst the later tactic is to convert SQL data types used in DDP to Microsoft Azure SQL database. To resolve the obstacle *Incompatible APIs (O44)*, the tactic *Develop adaptor (T6)* was used. For the obstacle *High-time for session handling (O43)*, the tactics *Make system stateless (T29)* was used. Also, in handling the obstacle *Switch between regular file system API to Microsoft Azure Storage API (O42)*, the architect selected the tactic *Decouple system components (T7)* from the repository.

To reduce the probability occurrence of the root obstacle *Microsoft Azure middleware latency (O29)*, the adopted resolution tactics was *Refine network topology (T24)* which belongs to *Obstacle reduction* group. This tactic was operationalized through selecting Microsoft Azure servers close to *InformIT*'s network located in North Europe. For the obstacle *Browser latency (O46)* the tactic *Update patches (T21)* is used regularly.

Furthermore, in addressing the obstacle *Performance variability of Microsoft Azure servers (O27)*, the system architect applied *Degrade goal (T36)*, a tactic from *Goal weakening* group, to modify the definition of satisfaction level for the root goal *Achieve [Improved performance]*. The tactic refined this goal to a more liberal one via allowing the expected processing time of documents by DDP to be varied up to 2 hours in a peak time for documents with size more than 40 megabyte. The suggested tactic was hard to get acceptance by DDP users; however, the purpose of considering this tactic was to probe possible solutions to tackle the obstacle. Hence, the second tactic was *Acquire more cloud resources (T26)* operationalized by adding 3 more virtual servers.

The occurrence likelihood of the obstacle *On-premise hardware latency (O47_1)* was found *Possible* with a consequence as *Insignificant* (i.e. Possible * Insignificant = Low risk) and thus left unresolved using the tactic *Do-Nothing*. Figure 12 shows the resultant goal model and incorporation of resolution tactics into the system architecture. Table 9 summarises all identified obstacles and selected resolution tactics.

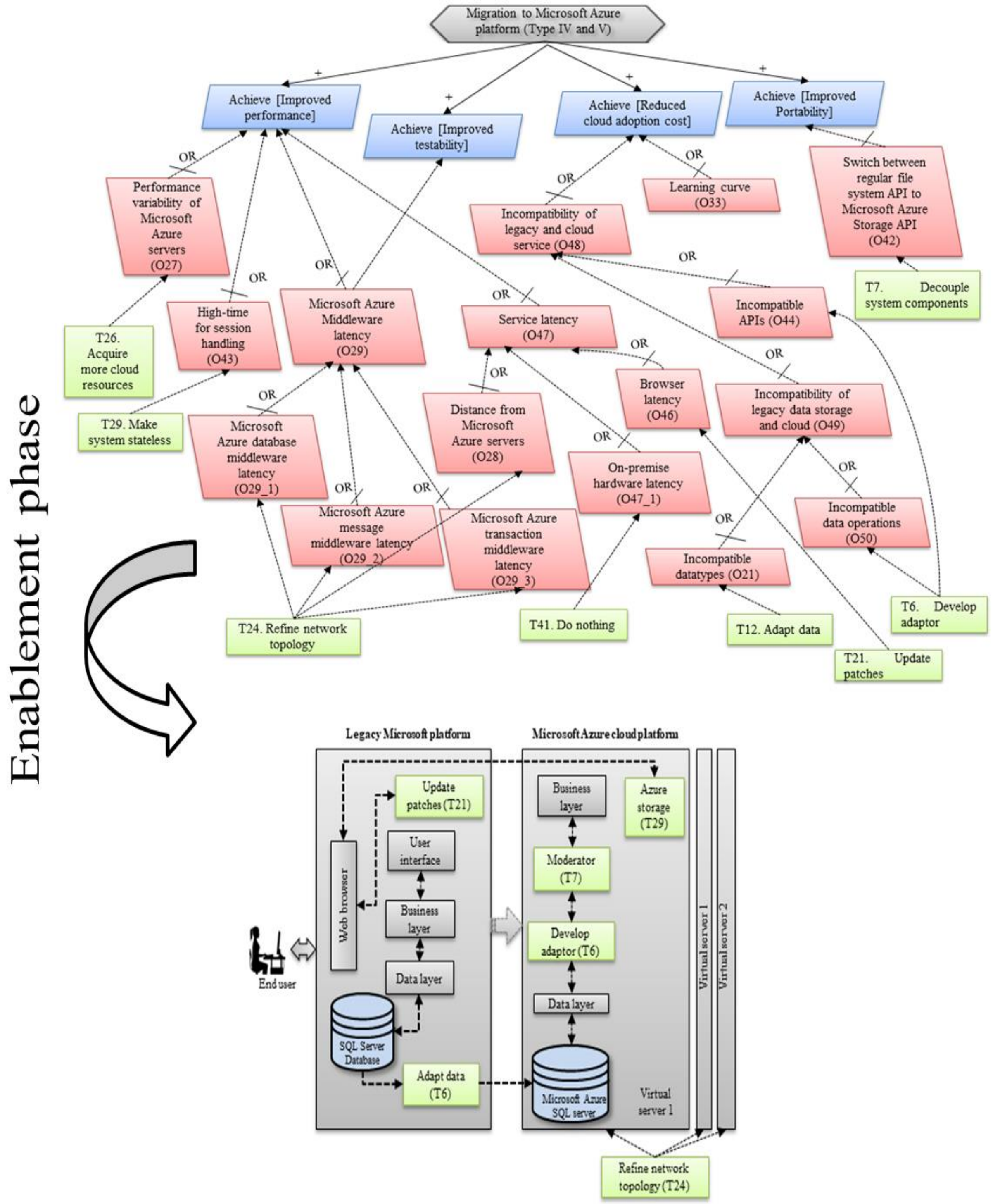

Migration to Microsoft Azure platform (Type IV and V)
+
Achieve [Improved performance]
Achieve [Improved testability]
Achieve [Reduced cloud adoption cost]
Achieve [Improved Portability]
OR
Performance variability of Microsoft Azure servers (O27)
Learning curve (O33)
Switch between regular file system API to Microsoft Azure Storage API (O42)
Incompatibility of legacy and cloud service (O48)
High-time for session handling (O43)
Microsoft Azure Middleware latency (O29)
Service latency (O47)
Incompatible APIs (O44)
T7. Decouple system components
T26. Acquire more cloud resources
Browser latency (O46)
Incompatibility of legacy data storage and cloud (O49)
T29. Make system stateless
Microsoft Azure database middleware latency (O29_1)
Distance from Microsoft Azure servers (O28)
On-premise hardware latency (O47_1)
Incompatible data operations (O50)
Microsoft Azure message middleware latency (O29_2)
Microsoft Azure transaction middleware latency (O29_3)
Incompatible datatypes (O21)
T6. Develop adaptor
T24. Refine network topology
T41. Do nothing
T12. Adapt data
T21. Update patches
Enablement phase
Legacy Microsoft platform
Microsoft Azure cloud platform
Update patches (T21)
Azure storage (T29)
Business layer
User interface
Moderator (T7)
Web browser
Develop adaptor (T6)
Data layer
End user
SQL Server Database
Adapt data (T6)
Microsoft Azure SQL server
Virtual server 1
Virtual server 1
Virtual server 2
Refine network topology (T24)

Figure 9. Resolutions tactics for handling obstacles (design) are incorporated into a new architecture of DDP during the enablement phase of migration process

Table 3 Repository support of obstacles and corresponding resolution tactics to address them

| Goal | Obstacle | | | | Resolution tactic (Specialized for this case study) |
|---|---|---|---|---|---|
| | Obstacle | Likelihood | Consequence | Risk | |
| Achieve [Reduced adoption cost] | O33 | Almost Certain | Major | E | *Substitute cloud service (T3).* Among three major cloud platforms Amazon Web Services, Google App Engine, and Microsoft Azure, the architect choses Microsoft Azure because the legacy system has been developed using Microsoft family technology and system developers has consistent experience with it. This reduces the cost of learning cloud technology and also potential effort in addressing incompatibilities between legacy system and the cloud. |
| | O44 | Likely | Moderate | H | *Develop adaptors (T6).* A wrapper component is developed to resolve API mismatches between legacy system and Microsoft Azure. It hides specific Microsoft Azure characteristics that cause conflicts with the legacy system. |
| | O21 | Almost Certain | Moderate | E | *Adapt data (T12).* The legacy system data types are converted to Microsoft Azure cloud database solution. |
| | O50 | Almost Certain | Moderate | E | *Develop adaptor (T6).* As the Microsoft Azure does not support some kind of stored procedures, an emulator is implemented and deployed on Microsoft Azure server which performs missing functionalities that are not supported by this platform. |
| Achieve [Improved Portability] | O42 | Likely | Moderate | H | *Decouple system components (T7).* Legacy system components are decoupled so that dependency among them is minimized and they can work independently and interact in a-synchronised way. A mediator component is implemented to manage interaction between the loosely coupled components deployed on Microsoft Azure servers. |
| Achieve [Improved testability] | O29_2 | Likely | Moderate | H | *Refine network topology (T24).* Define the geographical location of virtual machines close to North Europe to minimize latency of Azure middleware. |
| | O29_1 | Likely | Moderate | H | |
| | O29_3 | Likely | Moderate | H | |
| Achieve [Improved performance] | O27 | Likely | Major | E | *Acquire more cloud resources (T26).* Rent three virtual machines to address slow CPU clock rates. Use physical disk shipping to reduce effects of network latency/transfer rates. Use third party monitoring tools to independently verify the system performance. |
| | O29_2 | Likely | Moderate | H | |
| | O29_1 | Likely | Moderate | H | |
| | O29_3 | Likely | Moderate | H | |
| | O29 | Likely | Moderate | H | *Make system stateless (T29).* Legacy system components should be modified in a way that they do not depend on internal state. Rather, such states should be stored in an external storage or requested from an external component. |
| | O28 | Likely | Moderate | H | *Refine network topology (T24).* Modify the current deployment and distribution model of legacy system components on the basis of transaction delay, proximity, and geographical distribution. In this case, system components are deployed in Microsoft Azure servers located in North Europe close to Sweden to reduce latency. |
| | O46 | Likely | Moderate | H | *Update patches (T21).* Update cloud service consumer browsers regularly. |
| | O47_1 | Possible | Insignificant | L | *Do nothing (T41).* This obstacle is not perceived critical. |

## 6. Conclusion, limitations and future work

Large legacy systems that are in operation and stored critical data over years may have not been developed with unique concerns related to cloud environments such as security, elasticity, multi-tenancy, and interoperability. This article has been on this premise that moving legacy systems to cloud platforms may involve uncertain risks that should be systematically explored and mitigated in early stages where there is more flexibility to handle risks instead of experiencing them at later stages which might be costly once the legacy system working in the cloud.

Reusing the empirical knowledge of cloud service adoption in different scenarios is a promising approach in better exploration of these risks and reliability of decision outcomes. In this regard, our proposed framework harnesses a synergy between *evidence-based software engineering* and *goal-oriented modelling* approaches. The framework comprises an evidence-based repository of goals, obstacles, and corresponding resolution tactics that are utilized to improve the reliability and accuracy of decision outcomes in migrating legacy systems to cloud platforms. The output produced by the framework is a model of cloud adoption goals associated with potential obstacles, either from evidential data of the repository or the system domain along with subsequent resolution tactics to tackle obstacles. This obstacles and resolution tactics in this model are incorporated by developers into the enable phase during reengineering the architecture of legacy systems and making appropriate trade-offs on basis of, for example, cost, security, or performance. The application of the framework was illustrated through two case studies of moving two different Web-based systems, one providing real-time stock quotes and the second offering digital document processing services, to Pivotal Cloud Foundry and Microsoft Azure platforms. The proposed framework is the first attempt turning the existing body of knowledge of moving legacy systems to the cloud into a concise, accessible, and a reusable source. This has not been a feature of past research in the cloud computing field.

We do not claim that the repository of the framework is complete and it is unlikely to cover all possible issues. There might be some short-cuts to satisfy goals, or some hidden factors that hinder certain goal achievement. However, the proposed framework helps system architects to conduct better-informed decisions in similar situations based on the evidential data populated in the repository. Some deficiencies regarding the completeness of the repository are clear areas for further research. There is an unequal availability of empirical studies in the literature in support of the collections in the repository. In one hand, as shown in the Appendix B and suggested by several results, the resolution tactic *Develop adaptor/wrapper (T6)* can be used in addressing several obstacles namely *Incompatible pluggable cloud services (O19)*, *Incomplete APIs (O20)*, *Incompatible data types (O21)*, *Operating system incompatibility (O22)*, *Machine-image incompatibility (O23)*, *Virtual machine contextualization incompatibility (O24)*, *API incompatibility across multiple cloud (O25)*, and *Proprietary APIs (O36)*. On the other hand, there is only one resolution tactic to address the obstacle *Extra testing effort (O32)* which is *Prioritize tests (T30)*. Hence, further research is required to add more empirical findings to the repository as more studies appear in the cloud computing literature.

Our future work includes adding a probabilistic layer for goal specification and obstacle assessment in view of their estimation and required degrees of satisfaction grounded on system domain. The criticality of obstacle consequences will be computed by propagation probabilities from leaf obstacles towards high-level goals through the goal refinement model. To this aim, we will extend the procedure steps of the framework by annotating obstacle and goal elements with the probability of their occurrence (Cailliau and van Lamsweerde 2013). In addition, the framework repository in its current state is stored in textual template and does not provide a systematic mechanism for regularly updating the repository with new empirical data as identified in the literature. The goal-obstacle procedure utilizing the repository is also manual. We plan to provide a tool support for using the framework that facilitates using the framework when working with large scale goal models.

## Reference

A. Aurum, R. J., C. Wohlin, and M. Handzic (2003). Managing Software Engineering Knowledge.

AmazonS3 "Amazon Web Services S3 - Simple Cloud Storage Service." https://aws.amazon.com/s3/?sc_channel=PS&sc_campaign=acquisition_AU&sc_publisher=google&sc_medium=s3_b&sc_content=s3_e&sc_detail=s3.amazonaws.com&sc_category=s3&sc_segment=11

8649900484&sc_matchtype=e&sc_country=AU&s_kwcid=AL!4422!3!118649900484!e!!g!!s3.amazonaws.com&ef_id=Uvy8OgAABEYZb5Xa:20161022040726:s (Last accessed October 2016).

Anstett, T., F. Leymann, et al. (2009). Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes. Services - I, 2009 World Conference on.

Babar, M. A., Z. Liming, et al. (2004). A framework for classifying and comparing software architecture evaluation methods. Software Engineering Conference, 2004. Proceedings. 2004 Australian.

Cailliau, A. and A. van Lamsweerde (2013). "Assessing requirements-related risks through probabilistic goals and obstacles." Requirements Engineering **18**(2): 129-146.

Calheiros, R. N., R. Ranjan, et al. (2011). "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms." Software: Practice and Experience **41**(1): 23-50.

Chow, R., P. Golle, et al. (2009). Controlling data in the cloud: outsourcing computation without outsourcing control. Proceedings of the 2009 ACM workshop on Cloud computing security, ACM.

Christoforou, A. and A. S. Andreou (2013). A Cloud Adoption Decision Support Model Using Influence Diagrams. Artificial Intelligence Applications and Innovations, Springer**:** 151-160.

Dardenne, A., A. Van Lamsweerde, et al. (1993). "Goal-directed requirements acquisition." Science of computer programming **20**(1): 3-50.

De Assunçao, M. D., A. Di Costanzo, et al. (2009). Evaluating the cost-benefit of using cloud computing to extend the capacity of clusters. Proceedings of the 18th ACM international symposium on High performance distributed computing, ACM.

Deelman, E., G. Singh, et al. (2008). The cost of doing science on the cloud: the montage example. Proceedings of the 2008 ACM/IEEE conference on Supercomputing, IEEE Press.

Dieste, O. and O. Padua (2007). Developing search strategies for detecting relevant experiments for systematic reviews. Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on, IEEE.

Dragoni, N., S. Giallorenzo, et al. (2016). "Microservices: yesterday, today, and tomorrow." arXiv preprint arXiv:1606.04036.

El-Gazzar, R., E. Hustad, et al. (2016). "Understanding cloud computing adoption issues: A Delphi study approach." Journal of Systems and Software **118**: 64-84.

Fahmideh, M., F. Daneshgar, et al. (2017). "Key challenges during legacy software system migration to cloud computing platforms—an empirical study."

Fahmideh, M., F. Daneshgar, et al. (2016). "Cloud migration process—A survey, evaluation framework, and open challenges." Journal of Systems and Software **120**: 31-69.

Fahmideh, M., Low Graham, Ghassan Beydoun (2016b). "Conceptualising Cloud Migration Process." Twenty-Fourth European Conference on Information Systems (ECIS), İstanbul,Turkey, 2016 **1**: 0.

Fittkau, F., S. Frey, et al. (2012). CDOSim: Simulating cloud deployment options for software migration support. Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2012 IEEE 6th International Workshop on the.

Franklin, C. (1996). "Lt. Gen (USAF) Commander ESC, January 1996, Memorandum for ESC Program Managers, ESC/CC." Risk Management, Department of the Air Force, Headquarters ESC (AFMC) Hanscom Air Force Base, MA.

Garg, S. K., S. Versteeg, et al. (2011). SMICloud: A Framework for Comparing and Ranking Cloud Services. Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on.

Garg, S. K., S. Versteeg, et al. (2013). "A framework for ranking of cloud computing services." Future Generation Computer Systems **29**(4): 1012-1023.

Giovanoli, G. (2012). Building a Knowledge Base for Guiding Users through the Cloud Life Cycle

Godse, M. and S. Mulik (2009). An approach for selecting software-as-a-service (SaaS) product. 2009 IEEE International Conference on Cloud Computing, IEEE.

Godse, M. and S. Mulik (2009). An approach for selecting software-as-a-service (SaaS) product. Cloud Computing, 2009. CLOUD'09. IEEE International Conference on, IEEE.

Gordon, J. (2015). Case Study: Refactoring A Monolith Into A Cloud Native App, https://www.dropbox.com/s/t0sfl6cn6uounhy/Migration%20scenario-SpringTrader.docx?dl=0.

Gregor, S. and D. Jones (2007). "The anatomy of a design theory." Journal of the Association for Information Systems **8**(5): 312-335.

Henver, A., S. T. March, et al. (2004). "Design science in information systems research." MIS quarterly **28**(1): 75-105.

Juan-Verdejo, A. and H. Baars (2013). Decision support for partially moving applications to the cloud: the example of business intelligence. Proceedings of the 2013 international workshop on Hot topics in cloud services. Prague, Czech Republic, ACM**:** 35-42.

Khajeh-Hosseini, A., D. Greenwood, et al. (2010). Cloud migration: A case study of migrating an enterprise it system to iaas. Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, IEEE.

Khajeh-Hosseini, A., I. Sommerville, et al. (2011). Decision support tools for cloud migration in the enterprise. Cloud Computing (CLOUD), 2011 IEEE International Conference on, IEEE.

Khajeh-Hosseini, A., D. Greenwood, et al. (2012). "The cloud adoption toolkit: supporting cloud adoption decisions in the enterprise." Software: Practice and Experience **42**(4): 447-465.

Kitchenham, B., O. Pearl Brereton, et al. (2009). "Systematic literature reviews in software engineering – A systematic literature review." Information and software technology **51**(1): 7-15.

Kondo, D., B. Javadi, et al. (2009). Cost-benefit analysis of cloud computing versus desktop grids. Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on, IEEE.

Lacity, M. C., S. A. Khan, et al. (2009). "A review of the IT outsourcing literature: Insights for practice." The Journal of Strategic Information Systems **18**(3): 130-146.

Letier, E. (2001). Reasoning about agents in goal-oriented requirements engineering, PhD thesis, Université catholique de Louvain.

Letier, E. and A. Van Lamsweerde (2004). Reasoning about partial goal satisfaction for requirements and design engineering. ACM SIGSOFT Software Engineering Notes, ACM.

Leymann, F., C. Fehling, et al. (2011). "Moving applications to the cloud: An approach based on application model enrichment." International Journal of Cooperative Information Systems **20**(03): 307-356.

Linthicum, D. (2012). "Why Cloud Computing Projects Fail?" Available at: http://www.slideshare.net/Linthicum/why-cloud-computing-projects-fail, last access October 2016.

Low, C., Y. Chen, et al. (2011). "Understanding the determinants of cloud computing adoption." Industrial management & data systems **111**(7): 1006-1023.

Menzel, M. and R. Ranjan (2012). CloudGenius: decision support for web server cloud migration. Proceedings of the 21st international conference on World Wide Web, ACM.

Menzel, M., M. Schönherr, et al. (2013). "(MC2)2: criteria, requirements and a software prototype for Cloud infrastructure decisions." Software: Practice and Experience **43**(11): 1283-1297.

Nikkhouy, E. (2013). "Decision Making About Migrating To The Cloud Model." Cloud-Based Software Engineering: 8.

Ogawa, R. T. and B. Malen (1991). "Towards rigor in reviews of multivocal literatures: Applying the exploratory case study method." Review of educational research **61**(3): 265-286.

Peffers, K., T. Tuunanen, et al. (2008). "A design science research methodology for information systems research." Journal of management information systems **24**(3): 45-77.

Pepitone, J. (2011). "Amazon EC2 outage downs Reddit, Quora." Retrieved May **17**: 2011.

Rabetski, P. (2012). "Migration of an on-premise application to the cloud."

Rabetski, P. and G. Schneider (2013). Migration of an On-Premise Application to the Cloud: Experience Report. Service-Oriented and Cloud Computing, Springer**:** 227-241.

Rogers, E. (2003). M., Diffusion of Innovations, Free Press, New York.

Saripalli, P. and G. Pingali (2011). MADMAC: Multiple Attribute Decision Methodology for Adoption of Clouds. Cloud Computing (CLOUD), 2011 IEEE International Conference on.

Scandurra, P., M. Mongiello, et al. (2016). Towards a goal-oriented approach to adaptable re-deployment of cloud-based applications. Proceedings of the 6th international conference on cloud computing and services science.

Tak, B. C., B. Urgaonkar, et al. (2011). To move or not to move: the economics of cloud computing. Proceedings of the 3rd USENIX conference on Hot topics in cloud computing. Portland, OR, USENIX Association**:** 5-5.

Thönes, J. (2015). "Microservices." IEEE software **32**(1): 116-116.

Tsidulko, J. (2016). "The 10 Biggest Cloud Outages Of 2016." Available at http://www.crn.com/slide-shows/cloud/300081477/the-10-biggest-cloud-outages-of-2016-so-far.htm.

Van Lamsweerde, A. (2009). "Requirements engineering: from system goals to UML models to software specifications."

van Lamsweerde, A. and E. Letier (2000). "Handling obstacles in goal-oriented requirements engineering." Software Engineering, IEEE Transactions on **26**(10): 978-1005.

Van Lamsweerde, A. and E. Letier (2004). From object orientation to goal orientation: A paradigm shift for requirements engineering. Radical Innovations of Software and Systems Engineering in the Future, Springer**:** 325-340.

Walker, E., W. Brisken, et al. (2010). "To lease or not to lease from storage clouds." Computer **43**(4): 44-50.

Wu, W.-W. (2011). "Mining significant factors affecting the adoption of SaaS using the rough set approach." Journal of Systems and Software **84**(3): 435-441.

Wu, W.-W., L. W. Lan, et al. (2011). "Exploring decisive factors affecting an organization's SaaS adoption: A case study." International Journal of Information Management **31**(6): 556-563.

Yu, E. S. K. (1997). Towards modelling and reasoning support for early-phase requirements engineering. Requirements Engineering, 1997., Proceedings of the Third IEEE International Symposium on.

Zardari, S., R. Bahsoon, et al. (2014). "Cloud Adoption: Prioritizing Obstacles and Obstacles Resolution Tactics Using AHP."

Zimmermann, O., L. Wegmann, et al. (2015). Architectural decision guidance across projects-problem space modeling, decision backlog management and cloud computing knowledge. Software Architecture (WICSA), 2015 12th Working IEEE/IFIP Conference on, IEEE.

# Appendix A (Studies used to create the knowledge repository of the framework)

| Identifier | Study |
|---|---|
| [S1] | Torbacki, W. (2008). *SaaS–direction of technology development in ERP/MRP systems*. Archives of Materials Science 58: 58. |
| [S2] | Fox, A., et al. (2009). *Above the clouds: A Berkeley view of cloud computing*. Dept. Electrical Eng. and Comput. Sciences, University of California, Berkeley,Rep. UCB/EECS 28. |
| [S3] | Habib, S. M., et al. (2010). *Cloud computing landscape and research challenges regarding trust and reputation*. Ubiquitous Intelligence & Computing and 7th International Conference on Autonomic & Trusted Computing (UIC/ATC), 2010 7th International Conference on, IEEE. |
| [S4] | Wood, T., et al. (2010). *Disaster recovery as a cloud service: Economic benefits & deployment challenges*. 2nd USENIX Workshop on Hot Topics in Cloud Computing. |
| [S5] | Marston, S., et al. (2011). *Cloud computing—The business perspective*. Decision support systems 51(1): 176-189. |
| [S6] | Anstett, T., et al. (2009). *Towards BPEL in the Cloud: Exploiting Different Delivery Models for the Execution of Business Processes*. Services - I, 2009 World Conference on. |
| [S7] | Herbert, L. and J. Erickson (2009). *The ROI Of Software-As-A-Service*. Forrester Research. |
| [S8] | Wada, H., et al. (2011). *Data Consistency Properties and the Trade-offs in Commercial Cloud Storage: the Consumers' Perspective*. CIDR. |
| [S9] | La, H. J., et al. (2009). *Technical challenges and solution space for developing SaaS and mash-up cloud services*. e-Business Engineering, 2009. ICEBE'09. IEEE International Conference on, IEEE. |
| [S10] | Duipmans, E. and L. F. Pires (2012). *Business Process Management in the cloud: Business Process as a Service (BPaaS)*. University of Twente. |
| [S11] | Widera P, K. N. (2011). *Protein models comparator: scalable bioinformatics computing on the Google App Engine platform*. Computing Research Repository: 8. |
| [S12] | Andrikopoulos, V., et al. (2013). *How to adapt applications for the Cloud environment*. Computing 95(6): 493-535. |
| [S13] | Brebner, P. C. (2012). *Is your cloud elastic enough?: performance modelling the elasticity of infrastructure as a service (IaaS) cloud applications*. Proceedings of the third joint WOSP/SIPEW international conference on Performance Engineering, ACM. |
| [S14] | Li, A., et al. (2010). *CloudCmp: comparing public cloud providers*. Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, ACM. |
| [S15] | Mansfield-Devine, S. (2008). *Cloud Security: Danger in the clouds*. Netw. Secur. 2008(12): 9-11. |
| [S16] | Jensen, M., et al. (2009). *On technical security issues in cloud computing*. Cloud Computing, 2009. CLOUD'09. IEEE International Conference on, IEEE. |
| [S17] | Hay, B., et al. (2011). *Storm clouds rising: security challenges for IaaS cloud computing*. System Sciences (HICSS), 2011 44th Hawaii International Conference on, IEEE. |
| [S18] | Vogels, W. (2009). *Eventually consistent*. Communications of the ACM 52(1): 40-44. |
| [S19] | Ristenpart, T., et al. (2009). *Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds*. Proceedings of the 16th ACM conference on Computer and communications security, ACM. |
| [S20] | Gupta, R. (2012). *Above the Clouds: A View of Cloud Computing*. Asian Journal of Research in Social Sciences and Humanities 2(6): 84-110. |
| [S21] | Zissis, D. and D. Lekkas (2012). *Addressing cloud computing security issues*. Future Generation Computer Systems 28(3): 583-592. |
| [S22] | Hubbard, D. and M. Sutton (2010). *Top Threats to Cloud Computing V1. 0*. Cloud Security Alliance. |
| [S23] | Hussain, O. K., et al. (2012). *A framework for user feedback based cloud service monitoring*. Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on, IEEE. |
| [S24] | Alhamad, M., et al. (2010). *Conceptual SLA framework for cloud computing*. Digital Ecosystems and Technologies (DEST), 2010 4th IEEE International Conference on, IEEE. |
| [S25] | Harmer, T., et al. (2009). *Provider-Independent Use of the Cloud*. Euro-Par 2009 Parallel Processing. H. Sips, D. Epema etal., Springer Berlin Heidelberg. 5704: 454-465. |
| [S26] | K. Keahey, M. T., A. Matsunaga, and J. Fortes (2009). *Sky Computing*. IEEE Internet Computing, Palo Alto vol. 13: 315-340. |
| [S27] | Loutas, N., et al. (2010). *Towards a Reference Architecture for Semantically Interoperable Clouds*. Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on. |
| [S28] | Martinez Garro, J. and P. Bazan (2013). *Constructing hybrid architectures and dynamic services in Cloud BPM*. Science and Information Conference (SAI), 2013. |
| [S29] | Zissis, D. and D. Lekkas (2012). *Addressing cloud computing security issues*. Future Generation Computer Systems 28(3): 583-592. |
| [S30] | Vecchiola, C., et al. (2012). *Deadline-driven provisioning of resources for scientific applications in hybrid clouds with Aneka*. Future Generation Computer Systems 28(1): 58-65. |
| [S31] | Kossmann, D. and T. Kraska (2010). *Data Management in the Cloud: Promises, State-of-the-art, and Open Questions*. Datenbank-Spektrum 10(3): 121-129. |
| [S32] | Iosup, A., et al. (2011). *On the performance variability of production cloud services*. Cluster, Cloud and Grid |

| | |
|---|---|
| | Computing (CCGrid), 2011 11th IEEE/ACM International Symposium on, IEEE. |
| [S33] | Ma, D. (2007). *The Business Model of Software-As-A-Service*. Services Computing, 2007. SCC 2007. IEEE International Conference on, IEEE. |
| [S34] | Buyya, R., et al. (2008). *Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities*. High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on, Ieee. |
| [S35] | Rayport, J. F. and A. Heyward (2009). *Envisioning the cloud: the next computing paradigm*. Int. J. Database Manage. Syst.(IJDMS) 1(1). |
| [S36] | Vogels, W. (2009). *CTO roundtable: cloud computing*. |
| [S37] | Gao, J., et al. (2011). *Cloud testing-issues, challenges, needs and practice*. Software Engineering: An International Journal 1(1): 9-23. |
| [S38] | Strauch, S., Vasilios Andrikopoulos, Uwe Breitenbücher, Santiago Gómez Sáez,Oliver Kopp, Frank Leymann (2013). *Using Patterns to Move the Application Data Layer to the Cloud*. Proceedings of the 5th International Conference on Pervasive Patterns and Applications, PATTERNS 2013, 27 May – June 1 2013, Valencia, Spain, Xpert Publishing Services (XPS). |
| [S39] | Armstrong, D., et al. (2011). *Towards a contextualization solution for cloud platform services*. Cloud Computing Technology and Science (CloudCom), 2011 IEEE Third International Conference on, IEEE. |
| [S40] | Sriram, I. and A. Khajeh-Hosseini (2010). *Research agenda in cloud technologies*. arXiv preprint arXiv:1001.3259. |
| [S41] | Batarseh, F. A., et al. (2013). *Context-assisted test cases reduction for cloud validation*. International and Interdisciplinary Conference on Modeling and Using Context, Springer. |
| [S42] | Parveen, T., Tilley, S. (2010). *When to Migrate Software Testing to the Cloud?* Software Testing, Verification, and Validation Workshops (ICSTW), 2010 Third International Conference on. |
| [S43] | Tran, V., et al. (2011). *Application migration to cloud: a taxonomy of critical factors*. Proc. of 2nd International Workshop on Software Engineering for Cloud Computing, ACM. |
| [S44] | Khajeh-Hosseini, A., et al. (2010). *Cloud migration: A case study of migrating an enterprise it system to iaas*. Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on, IEEE. |
| [S45] | Catteddu, D. (2010). *Cloud Computing: Benefits, Risks and Recommendations for Information Security*. Web Application Security. C. Serrão, V. Aguilera Díaz and F. Cerullo, Springer Berlin Heidelberg. 72: 17-17. |
| [S46] | Khajeh-Hosseini, A., et al. (2010). *Research challenges for enterprise cloud computing*. arXiv preprint arXiv:1001.3257. |
| [S47] | Torbacki, W. (2008). *SaaS–direction of technology development in ERP/MRP systems*. Archives of Materials Science 58: 58. |
| [S48] | M. Xin, N. L. (2008). *Software-as-a-service model: elaborating client-side adoption factors*. Proceedings of the Twenty-ninth International Conference on Information |
| [S49] | Kerr, J. and K. Teng (2012). Cloud computing: legal and privacy issues. Journal of Legal Issues and Cases in Business 1: 1. |
| [S50] | Leavitt, N. (2009). *Is Cloud Computing Really Ready for Prime Time?* Computer 42(1): 15-20. |
| [S51] | Satzger, B., et al. (2013). *Winds of change: from vendor lock-in to the meta cloud*. Internet Computing, IEEE 17(1): 69-73. |
| [S52] | Silva, G. C., et al. (2013). *A systematic review of cloud lock-in solutions*. Cloud Computing Technology and Science (CloudCom), 2013 5th International Conference on, IEEE. |
| [S53] | Dillon, T., et al. (2010). *Cloud computing: Issues and challenges*. Advanced Information Networking and Applications (AINA), 2010 24th IEEE International Conference on, Ieee. |
| [S54] | So, K. (2011). *Cloud computing security issues and challenges*. International Journal of Computer Networks 3(5). |
| [S55] | Dalheimer, M. and F.-J. Pfreundt (2009). *Genlm: license management for grid and cloud computing environments*. Cluster Computing and the Grid, 2009. CCGRID'09. 9th IEEE/ACM International Symposium on, IEEE. |
| [S56] | Morgan, L. and K. Conboy (2013). *Factors affecting the adoption of cloud computing: an exploratory study*. |
| [S57] | Joint, A., et al. (2009). *Hey, you, get off of that cloud?* Computer Law & Security Review 25(3): 270-274. |
| [S58] | Hajjat, M., et al. (2010). *Cloudward bound: planning for beneficial migration of enterprise applications to the cloud*. Proceedings of the ACM SIGCOMM 2010 conference. New Delhi, India, ACM: 243-254. |
| [S59] | Bezemer, C.-P. and A. Zaidman (2010). *Multi-tenant SaaS applications: maintenance dream or nightmare?* Proceedings of the Joint ERCIM Workshop on Software Evolution (EVOL) and International Workshop on Principles of Software Evolution (IWPSE), ACM. |
| [S60] | Chong, F., et al. (2006). *Multi-tenant data architecture*. MSDN Library, Microsoft Corporation |
| [S61] | Krebs, R., et al. (2012). *Architectural Concerns in Multi-tenant SaaS Applications*. CLOSER. |
| [S62] | Mietzner, R., et al. (2009). *Variability modeling to support customization and deployment of multi-tenant-aware Software as a Service applications*. Proceedings of the 2009 ICSE Workshop on Principles of Engineering Service Oriented Systems, IEEE Computer Society: 18-25. |
| [S63] | Gonidis, F., et al. (2012). *Addressing the challenge of application portability in cloud platforms*. 7th South-East European Doctoral Student Conference. |
| [S64] | Petcu, D. (2011). *Portability and interoperability between clouds: challenges and case study*. European Conference on a Service-Based Internet, Springer. |

| | |
|---|---|
| [S65] | Chauhan, M. A. and M. A. Babar (2012). *Towards Process Support for Migrating Applications to Cloud Computing*. Cloud and Service Computing (CSC), 2012 International Conference on. |
| [S66] | Strauch, S., et al. (2014). *Migrating eScience Applications to the Cloud: Methodology and Evaluation*. |
| [S67] | Strauch, S., V.A., D. Karastoyanova, F. Leymann, (2014), *Migrating Enterprise Applications to the Cloud: Methodology and Evaluation*, International Journal of Big Data Intelligence. |
| [S68] | ShaoJie, T., Yuan†, J., Li., X, 2016, AMAZING: An Optimal Bidding Strategy for Amazon EC2 Cloud Spot Instance, available at http://www.cs.iit.edu/~xli/paper/Conf/EC-CLOUD2012.pdf |
| [S69] | Computing, C. (2010). *Toward a multi-tenancy authorization system for cloud services*. |
| [S70] | Karampaglis, Z., et al. (2012). *Secure Migration of Legacy Applications to the Web*. Migration(1/18). |
| [S71] | Pahl, C., et al. (2013). *A Comparison of On-Premise to Cloud Migration Approaches*. Service-Oriented and Cloud Computing, Springer: 212-226. |
| [S72] | Council, C. S. C. (2013). *Migration applications to public Cloud Services: roadmap for success*. |
| [S73] | Menychtas, A., et al. (2013). *ARTIST Methodology and Framework: A novel approach for the migration of legacy software on the Cloud*. Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), 2013 15th International Symposium on, IEEE. |
| [S74] | Varia, J. (2010). *Migrating your existing applications to the aws cloud*: *A Phase-driven Approach to Cloud Migration*. |
| [S75] | Betts, D. (2012). *Moving Apps to the Cloud on Microsoft* |
| [S76] | Binz, T., et al. (2011). *CMotion: A framework for migration of applications into and between clouds*. Service-Oriented Computing and Applications (SOCA), 2011 IEEE International Conference on. |
| [S77] | Rabetski, P. and G. Schneider (2013). *Migration of an On-Premise Application to the Cloud: Experience Report*. Service-Oriented and Cloud Computing, Springer: 227-241. |
| [S78] | Bahga, A. and V. K. Madisetti (2013). *Rapid Prototyping of Multitier Cloud-Based Services and Systems*. Computer 46(11): 76-83. |
| [S79] | Chauhan, M. A. and M. A. Babar (2011). *Migrating Service-Oriented System to Cloud Computing: An Experience Report*. Cloud Computing (CLOUD), 2011 IEEE International Conference on. |
| [S80] | Guo, C. J., et al. (2007). *A framework for native multi-tenancy application development and management*. 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. |
| [S81] | Huru, H. A. (2009). *MILAS: ModernIzing Legtacy Applications towards Service Oriented Architecture (SOA) and Software as a Service (SaaS)*. |
| [S82] | Zagarese, Q., et al. (2012). *Enabling advanced loading strategies for data intensive web services*. Web Services (ICWS), 2012 IEEE 19th International Conference on, IEEE. |
| [S83] | Miranda, J., et al. (2013). *Assisting Cloud Service Migration Using Software Adaptation Techniques*. Proceedings of the 2013 IEEE Sixth International Conference on Cloud Computing, IEEE Computer Society. |
| [S84] | Laszewski, T. and P. Nauduri (2011). *Migrating to the Cloud: Oracle Client/Server Modernization*, Elsevier. |
| [S85] | Bessani, A., et al. (2011). *DepSky: dependable and secure storage in a cloud-of-clouds*. Proceedings of the sixth conference on Computer systems. Austria, ACM: 31-46. |
| [S86] | Nussbaumer, N. and X. Liu (2013). *Cloud Migration for SMEs in a Service Oriented Approach. Computer Software and Applications*, Conference Workshops (COMPSACW), 2013 IEEE 37th Annual. |
| [S87] | Banerjee, J. (2012). *Moving to the cloud: Workload migration techniques and approaches*. High Performance Computing (HiPC), 2012 19th International Conference on, IEEE. |
| [S88] | Bezemer, C.-P., et al. (2010). *Enabling multi-tenancy: An industrial experience report*. Software Maintenance (ICSM), 2010 IEEE International Conference on, IEEE. |
| [S89] | Kwok, T., et al. (2008). *A software as a service with multi-tenancy support for an electronic contract management application*. Services Computing, 2008. SCC'08. IEEE International Conference on, IEEE. |
| [S90] | Zhu, Y. (2014). *A Platform for Changing Legacy Application to Multi-tenant Model*. International Journal of Multimedia and Ubiquitous Engineering 9(8): 407-418. |
| [S91] | Pahl, C. and H. Xiong (2013). *Migration to PaaS clouds-Migration process and architectural concerns*. Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the, IEEE. |
| [S92] | Durkee, D. (2010). *Why cloud computing will never be free*. Queue 8(4): 20. |
| [S93] | Barker, S. K. and P. Shenoy (2010). *Empirical evaluation of latency-sensitive application performance in the cloud*. Proceedings of the first annual ACM SIGMM conference on Multimedia systems, ACM. |
| [S94] | Khajeh-Hosseini, A., et al. (2011). *Decision support tools for cloud migration in the enterprise*. Cloud Computing (CLOUD), 2011 IEEE International Conference on, IEEE. |
| [S95] | Batarseh, F. A., et al. (2013). *Context-assisted test cases reduction for cloud validation*. International and Interdisciplinary Conference on Modeling and Using Context, Springer. |
| [S96] | Krebs, R., et al. (2012). *Architectural Concerns in Multi-tenant SaaS Applications*. CLOSER 12: 426-431. |
| [S97] | Strauch, S., et al. (2012). *ESB MT: Enabling Multi-Tenancy in Enterprise Service Buses*. Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, IEEE. |
| [S98] | S.Strauch, V. A., T.Bachmann, D.Karastoyanova, S.Passow, K.Vukojevic-Haupt (2013). *Decision Support for the Migration of the Application Database Layer to the Cloud*. Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2013, 2-5 December 2013, Bristol, UK: 639--646. |

| | |
|---|---|
| [S99] | Qaisi, L.M., and Aljarah, I. (2016). *A Twitter Sentiment Analysis for Cloud Providers: A Case Study of Azure Vs. Aws*, Computer Science and Information Technology (CSIT), 2016 7th International Conference on: IEEE, pp. 1-6. |
| [S100] | Dignan, L. (2016). *Public Cloud Computing Vendors: A Look at Strengths, Weaknesses, Big Picture*, available at http://www.zdnet.com/article/public-cloud-computing-vendors-a-look-at-strengths-weaknesses-big-picture/. |
| [S101] | Tsai, P. (2016). *Aws Vs. Azure: It Pros Weigh the Pros and Cons*, available at: https://www.cloudcomputing-news.net/news/2016/sep/06/aws-vs-azure-it-pros-weigh-pros-and-cons. |
| [S102] | Serrano, N., Gallardo, G., and Hernantes, J. (2015). *Infrastructure as a Service and Cloud Technologies*, IEEE software (32:2), pp. 30-36. |
| [S103] | Modi, C., Patel, D., Borisaniya, B., Patel, A., and Rajarajan, M. (2013). *A Survey on Security Issues and Solutions at Different Layers of Cloud Computing*, The journal of supercomputing), pp. 1-32. |
| [S104] | Somorovsky, J., Heiderich, M., Jensen, M., Schwenk, J., Gruschka, N., and Lo Iacono, L. (2011). *All Your Clouds Are Belong to Us: Security Analysis of Cloud Management Interfaces*, Proceedings of the 3rd ACM workshop on Cloud computing security workshop: ACM, pp. 3-14. |
| [S105] | Tajadod, G., Batten, L., and Govinda, K. (2012). *Microsoft and Amazon: A Comparison of Approaches to Cloud Security*, Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on: IEEE, pp. 539-544. |
| [S106] | Roloff, Eduardo, et al., (2012), *Evaluating high performance computing on the windows azure platform*, Cloud Computing (CLOUD), IEEE 5th International Conference on. IEEE. |
| [S107] | Google documentation, (2017). *Regions and Zones*, available at https://cloud.google.com/compute/docs/regions-zones/regions-zones. |
| [S108] | Smeds, J., Nybom, K., and Porres, I, (2015). *Devops: A Definition and Perceived Adoption Impediments*, in Agile Processes in Software Engineering and Extreme Programming: 16th International Conference, Xp 2015, Helsinki, Finland, May 25-29, 2015, Proceedings, C. Lassenius, T. Dingsøyr and M. Paasivaara (eds.). Cham: Springer International Publishing, pp. 166-177. |
| [S109] | Riungu-Kalliosaari, L., (2016). *DevOps Adoption Benefits and Challenges in Practice: A Case Study*, Product-Focused Software Process Improvement: 17th International Conference, PROFES 2016, Trondheim, Norway, November 22-24, 2016, Proceedings 17. Springer International Publishing. |
| [S110] | Pahl, Claus, (2015). *Containerization and the paas cloud*, IEEE Cloud Computing 2.3, p 24-31. |
| [S111] | Gunka, A, Stepan, S., (2013). *Moving an application to the cloud: an evolutionary approach*. Proceedings of the 2013 international workshop on Multi-cloud applications and federated clouds. ACM. |
| [S112] | Ardagna, Danilo, (2012), *Modaclouds: A model-driven approach for the design and execution of applications on multiple clouds.* Proceedings of the 4th international workshop on modeling in software engineering. IEEE Press. |

# Appendix B (Collections in the framework repository)

Catalogue of goals commonly contributed by cloud computing technology

| # | Quality goal | Explanation (from cloud service consumer perspective) | Study |
|---|---|---|---|
| G1 | Availability | Anywhere/anytime/any device (desktop, laptop, and mobile) access to resources (e.g. CPU, storage, virtual machines, and network bandwidth) which are redundant and guarantee more availability (24/7/365 and 99.99% availability) compared to run in-house infrastructure. | [S2], [S3], [S4], [S5], [S35], [S36], and [S37]. |
| G2 | Scalability | On the fly scaling up/ down resources and capability to provide varying resource demanding patterns. | |
| G3 | Security | Providing secure services protected from unauthorized access by other tenants. | |
| G4 | Performance | An excellent throughout speed and computations on cutting edge infrastructure. | |
| G5 | Customizability | Customisable and modifiable services upon requirements of consumers. | |
| G6 | Interoperability | Cloud services are integrable and incorporable with systems as required. | |
| G7 | Portability | Systems can move from one cloud to another cloud to get better offer (e.g. performance, price, and security) with minimum disruption. | |
| G8 | Testability | Providing a scalable infrastructure to perform test and evaluation of high-computational tasks. | |
| G9 | Consistency | Guarantee of data consistency and not resulting in an error state for the system once data are processing and changing in the cloud. | |
| G10 | Reduced IT cost | Lower expense for infrastructure procuring, data storages, software updates, maintenance, and staff. | |

Probable obstacles against goals in migrating legacy systems to cloud platforms

| # | Obstacle | Definition | Quality goals | | | | | | | | | | Migration type* | | | | | Study |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | Availability | Scalability | Security | Performance | Customizability | Interoperability | Portability | Testability | Consistency | Reduced IT cost | I | II | III | IV | V | |
| O1 | Cloud outage | A cloud provider may suffer from outages for reasons such as going out of business, being the subject of regulatory action, or the outage of contacts system. | * | | | | | | | | | | √ | √ | √ | √ | √ | [S2], [S44], [S58] |
| O2 | Service failure | Cloud service may be unavailable and callable by service consumer due to reasons such as network congestion, hardware failure, service middleware failure, or faults on various elements of service platforms. | * | | | | | | | | | | √ | √ | √ | √ | √ | [S2], [S9] |
| O3 | Service transient fault | Cloud service maybe temporarily unavailable due to network traffic load or restarting by administrators after a failure. | * | | | | | | | | | | √ | √ | √ | √ | √ | [S2], [S9] |
| O4 | Tenant interfere | Several tenants maybe in run on the same cloud and negatively affect the system data security. | | | * | | | | | | | | √ | √ | √ | √ | √ | [S59], [S60], [S88] |
| O5 | Un-customisable scalability | The scalability rules may not be flexible and merely controlled and managed by service provider. | | * | | | * | | | | | | - | √ | - | - | - | [S10], [S11] |
| O6 | Scaling latency | Cloud service may have delay in providing resource requested by service consumer due to reasons such as a server workload in the region, the rate of load acceleration, or quotas imposed by the cloud service provider. | | * | | | | | | | | | √ | √ | - | √ | √ | [S12], [S13], [S14] |
| O7 | Browser vulnerabilities | Cloud consumer who connects to cloud services by a Web browser might be attacked by a malicious tenant. | | | * | | | | | | | | √ | √ | √ | √ | √ | [S15], [S16] |
| O8 | Code disruption | System codes are performing in the cloud might be accessed and disrupted by other tenants are in operation in the same cloud. | | | * | | | | | | | | √ | - | - | - | √ | [S6], [S17] |
| O9 | Cloud attack | A malicious tenant can disrupt cloud service functionalities. | | | * | | | | | | | | √ | √ | √ | √ | √ | [S16], [S17], [S45], [S58] |
| O10 | Extra security cost | There might be an extra cost to address security if system components are deployed across different cloud server with complex relationships and security configuration which demands for provider-independent methods to establish a security and configuration context. Service consumer might be responsible for locking ports, patching the operating system, running an anti-virus software and enforcement of access control policies. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S6], [S26], [S28] |
| O11 | Lack of control on code execution location | Executing of a system in the cloud might not be fixed to a geographical location and rather the system may move from one physical server to another one during its lifetime. The decision on the execution location | | | * | | | | | | | | √ | - | - | - | √ | [S17], [S19] |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  |  | of the system is based on factors such as load balancing mechanism of cloud, network and server performance and availability, and even characteristics of the current consumer. |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| O12 | Lack of control on data location | Sensitive data may move to the outside the organization network or country. There is no assumption where the location of the data is. |  |  | * |  |  |  |  |  |  |  | - | √ | √ | √ | √ | [S12], [S20], [S21] |
| O13 | Data remanence | The residual representation of data after finishing system execution on the cloud server may cause unwilling disclosure of private data. |  |  | * |  |  |  |  |  |  |  | - | √ | √ | √ | √ | [S22] |
| O14 | Data interruption | Tenants or subcontractors of cloud providers may get access to system data and affect data confidentiality. |  |  | * |  |  |  |  |  |  |  | - | √ | √ | √ | √ | [S29] |
| O15 | Session hijacking | A malicious tenant may use a valid session key to get authorised access to use system using cloud service. |  |  | * |  |  |  |  |  |  |  | √ | √ | - | - | √ | [S29] |
| O16 | System source codes propriety | Cloud provider, its subcontractors, or tenant may get access to all system codes/algorithms which might be confidential. |  |  | * |  |  |  |  |  |  |  | √ | √ | - | - | √ | [S17] |
| O17 | Vendor lock-in | System owner is dissatisfied with cloud service but it cannot easily and inexpensively transfer its system and data to another platform or in-house. |  |  |  |  |  | * | * |  |  |  | √ | √ | √ | √ | √ | [S50], [S51], [S52] |
| O18 | Traversal vulnerability | A malicious tenant may damage resources that are used by other tenants. |  |  | * |  |  |  |  |  |  |  | √ | √ | √ | √ | √ | [S59], [S60], [S61], [S62] |
| O19 | Incompatible pluggable cloud services | At runtime, system might be plugged to a cloud service which is incompatible with the other cloud services. |  |  |  |  |  | * | * |  |  |  | √ | - | - | - | √ | [S23] |
| O20 | Incomplete APIs | Cloud service provider lacks providing a rich set of APIs. |  |  |  |  | * | * | * |  |  |  | √ | √ | - | √ | √ | [S24] |
| O21 | Incompatible data types | Data types used in legacy and cloud service are incompatible. |  |  |  |  |  | * | * |  |  |  | √ | √ | - | √ | √ | [S12], [S38] |
| O22 | Operating system incompatibility | System components are distributed and moved among cloud servers with different operating systems which might be incompatible for managing, representing, and formatting virtual machines. |  |  |  |  |  | * | * |  |  |  | √ | - | - | - | √ | [S25], [S26], [S27] |
| O23 | Machine-image incompatibility | Virtual machines are moving between different cloud platforms but each platform has different underlying implementation for virtual machines. |  |  |  |  |  | * | * |  |  |  | √ | - | - | - | √ | [S39], [S40] |
| Q24 | Virtual machine contextualization incompatibility | Virtual machines are moving between different platforms but each platform may use different methods for customizing the context of virtual machine such as setting the operating system's username and password. |  |  |  |  |  | * | * |  |  |  | √ | √ | - | - | √ | [S39],[S40] |
| O25 | API incompatibility across multiple cloud | Cloud service may offer APIs to implement systems or virtual machines which might be incompatible with each other services. |  |  |  |  |  | * | * |  |  |  | √ | - | - | - | √ | [S25], [S26], [S27], [S30], [S40] |
| O26 | Message passing | Message passing between system and cloud services or among system components deployed on cloud servers might be unsecure and accessed by malicious tenants. Also, message size might be large affecting system performance. |  |  | * | * |  |  |  |  |  |  | √ | √ | √ | √ | √ | [S12], [S45] |
| O27 | Performance variability of cloud service | Workload variability, virtualization overheads, or resource time-sharing of cloud server may have negative effect on the system performance operating in the cloud. |  |  |  | * |  |  |  |  |  |  | √ | √ | - | - | √ | [S12], [S31], [S32], [S93] |
| O28 | Geographical distance | High distance between system components that are distributed and |  | * |  | * |  |  |  |  |  |  | √ | √ | √ | √ | √ | [S12] |

| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | deployed on cloud servers may cause increased latency when accessing or manipulating the data. | | | | | | | | | | | | | | | | |
| O29 | Low middleware performance | A cloud service may have been built on several layers of middleware, from the guest operating system of the VM to the data-centre resource manager, which each middleware may impact on the system efficiency. | | * | | * | | | | * | | | √ | √ | - | √ | √ | [S32] |
| O30 | High cancellation fees | Cloud service provider may force a consumer to a long term commitment and consumers’ early exit may causes forfeit. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S33] |
| O31 | Inflexible pricing model | Cloud service provider may not offer a billing model based on the service usage and limit consumer to flat rates or usage thresholds. | | | | | * | | | | | | √ | √ | √ | √ | √ | [S34] |
| O32 | Extra testing effort | The test of system which may be deployed on multiple cloud servers may needs testing connectivity of local components and those deployed on cloud servers along with adding a new dimension of test such as elasticity, multi-tenancy, interoperability, and elasticity. | | | | | * | | | * | | | √ | √ | √ | √ | √ | [S37], [S41], [S42], [S95] |
| O33 | Learning curve | Learning a new programming style, concepts, APIs, tools, and understanding organisational impact of the cloud technology might be time consuming. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S43] |
| O34 | Loose of control over resources and updates | Loss of control over resource management and their update. | | | | | * | | | | | | √ | √ | √ | √ | √ | [S45], [S46] |
| O35 | Bargaining power of provider | Cloud provider may get bargaining power in the future for example by raising service fee prices or refusing to invest maintenance backward compatible interface. | | | | | * | | | | | | √ | √ | √ | √ | √ | [S48], [S49] |
| O36 | Proprietary APIs | Proprietary cloud APIs may impede integration of cloud services with legacy systems. | | | | | * | * | * | | | | √ | √ | √ | √ | √ | [S53], [S54] |
| O37 | Licensing issue | Software is charged per instance model but cloud server creates several instances in the case of workload occurrence which might be contradictory with software licensing. | | | | | | | | | | * | √ | √ | - | √ | √ | [S45], [S55] |
| O38 | Department downsizing | The maintenance team of legacy systems may become downsize as some of their responsibilities are outsourced to cloud providers. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S44], [S56] |
| O39 | Resistance to change | Users/staff may resist against moving to the cloud due to change in their positions and organisational structure. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S36], [S40] |
| O40 | Non-compliancy | Users or standard regulations don’t consent to move personal/organisational data to the cloud. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S57] |
| O41 | Extra management effort | Maintaining a system deployed in several clouds takes extra effort such as keeping relationships with cloud providers, change of providers, and monitoring. | | | | | | | | | | * | √ | √ | √ | √ | √ | [S44] |
| O42 | Backward incompatibility | System might not be easily switched between on-premise and cloud environments. | | | | | | | * | | | | √ | √ | √ | √ | √ | [S77] |
| O43 | State-based dependency | System may heavily depend on contextual data, storing on server or client, such as configuration changes to operate and remain consistent from one session to another one. | | * | | * | | | | | | | √ | √ | - | - | √ | [S71], [S91] |
| O44 | Incompatible APIs | Legacy system APIs and cloud’s APIs are incompatible. | | | | | | * | * | | | * | √ | √ | √ | √ | √ | [S12], [43] |

| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| O45 | Network latency | Connection speed between on premise and cloud is low due to latency in on-premise network or latency of internal cloud network. | | * | | * | | | | | | | √ | √ | √ | √ | √ | [S12] |
| O46 | Browser latency | The browser in the on-premise environment is working slowly. | | * | | * | | | | | | | √ | √ | √ | √ | √ | [S12] |
| O47 | Service latency | Latency in performing cloud service due to obstacles O7, O28, O46, and O45. | | * | | * | | | | | | | √ | √ | √ | √ | √ | [S12] |
| O48 | Incompatibility of legacy system and cloud service | Incompatibility between legacy system and cloud services due to obstacles O21, O22, O23, Q24, and O25. | | | | | | * | * | | | * | √ | √ | √ | √ | √ | [S12], [S43] |
| O49 | Incompatibility of legacy system data storage and cloud | Incompatibility of between legacy data storage and cloud database solution due to O21 and O50. | | | | | | * | * | | | * | - | - | - | √ | - | [S12], [S43] |
| O50 | Incompatible data operations | Operations such as stored procedure, views, and functions providing by cloud data store might not be compatible (either syntactically or semantically) with those defined in legacy system. | | | | | | * | * | | | * | - | - | - | √ | - | [S12], [S43] |
| O51 | Tight dependencies | Tight dependencies among legacy system components or dependency to underlying technologies, operating systems, programming language, or other legacy systems may obstruct individual scalability and portability of components across multiple clouds and on premise. | | * | | | | | * | * | | | √ | √ | √ | √ | √ | [S67], [S98], [S108] |
| O52 | Inconsistency of system components | Cloud data storage services may offer weaker consistency properties in the sense that it will be taken long time to have consistent data across all servers. | | | | | | | | | * | | - | - | √ | √ | √ | [S8], [S18] |
| O53 | Identity theft | An attacker may get a valid user's identity and access resources of legacy systems. | | | * | | | | | | | | √ | √ | √ | √ | √ | [S104] |
| O54 | Variable price of cloud resources | The price of using cloud resources may vary depending on cloud workload across the time, particularly in a pick period. Such price variation may not be suitable for legacy systems with heavy processing tasks. | | | | | | | | | | * | √ | √ | - | - | √ | [S112] |
| O55 | High cost of support (Specific to AWS) | AWS is a general provider which expects its services to be used and managed by its uses independently. If a problem occurs, AWS has an expensive technical support. | | | | | | | | | | * | √ | - | - | - | - | [S102] |
| O56 | Unreliable IT support (Specific to AWS) | The quality of IT support by AWS might be a risk. | | | | | | | | | | * | √ | - | √ | - | √ | [S100] |
| O57 | Varying support fee (Specific to AWS) | AWS support fees vary on a sliding scale tied to monthly in a way that support costs may grow quickly if system performs heavy tasks. | | | | | | | | | | * | √ | - | - | - | - | [S101] |
| O58 | Vulnerable security (Specific to AWS) | Amazon simple storage service (S3) may be accessible via SSL (secure sockets layer) encrypted end points, implying that it is the user's responsibility to encrypt data before storing into S3. | | | * | | | | | | | | - | - | √ | - | - | [S103] |
| O59 | Injection attack (Specific to AWS) | An attacker may hijack user accounts by creating, modifying, and deleting virtual machine images, and changing administrative passwords to control interfaces used to manage cloud computing resources (e.g. S3 or EC2). | | | * | | | | | | | | √ | - | √ | - | - | [S104] |
| O60 | Inflexible cost model | Azure computes the cost of recourses that were used per minute with | | | | | | | | | | * | √ | √ | √ | √ | √ | [S99] |

| | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | (Specific to Azure) | rounding up service usage to the nearest minute. In other words, if a user allocated a resource for one hour and a half, then payment is computed for the exact period of time whilst a provider like Amazon round up service consumption to nearest hour. | | | | | | | | | | | | | | | | |
| O61 | Inflexible configuration (Specific to Azure) | The provider may not provide high flexible hardware configurability for each virtual machine instance compared to Amazon offering high flexibility in virtual machine configuration. | | | | | * | | | | | | √ | - | - | - | - | [S104] |
| O62 | Operating system incompatibility (Specific to Azure) | Microsoft Azure mainly supports Windows-based servers. Porting legacy systems from other platforms (e.g. Linux) to Azure might require modifying the source code to be compatible with Windows APIs and them able to execute on Azure. | | | | | * | | | | | | √ | | | | | [S104], [S106] |
| O63 | Limited geographical zone (Specific to Google) | Google may not provide an extensive coverage of data centres to deploy legacy systems. | * | | | | | | | | | | √ | | | | | [S107] |
| O64 | Inflexible cost model (Specific to Rackspace) | Rackspace may offer limited pricing options and month-to-month subscriptions. | | | | | | | | | | * | √ | - | - | - | - | [S102] |
| O65 | Heterogeneous production environments | The complexity and differences between production environments (e.g. cloud platform, third-party clouds, and legacy systems) related to deployments and configurations can hinder the efficiency of test. | | | | | | | | * | | | √ | √ | √ | √ | √ | [S108], [S109] |
| O66 | Costly virtual machine | Virtual machine and its underlying infrastructure might be costly in terms of need for large disk storage, isolated binary and library files, memory management, and full gust operating system image. | | | | | | | | | | * | √ | √ | - | - | √ | [S110] |
| O67 | Incompatible execution environments of system | A legacy system which is encapsulated in a virtual machine may not be interoperable and portable across multiple cloud platforms. | | | | | | * | | | | | √ | √ | - | - | √ | [S110] |

* For the migration types see the criterion migration type in Section 2.

| Catalogue of resolution tactics for handling obstacle | | | | | |
|---|---|---|---|---|---|
| **#** | **Resolution tactic** | **Definition** | **Relation to obstacle** | **Source** | **Category** |
| T1 | Substitute goal | Identify an alternative goal which is still contributable by the chosen migration type or cloud services in a way that the obstructed goal and obstacle will not occur. | Applicable to resolve all obstacles | KAOS methodology | Goal/Service/Migration type Substitution |
| T2 | Substitute cloud migration type | Choose an alternative cloud adoption type which satisfies the obstructed goal is adopted in a way that the obstacle will no longer occur. The tactics has root in the fact that different cloud adoption types, besides their specific contributions to quality goals, might have common contributions towards migration goals. | Applicable to resolve all obstacles | Inspired from KAOS methodology | Goal/Service/Migration type Substitution, Obstacle reduction |
| T3 | Substitute cloud service | Resolve the obstacle by selecting/changing the cloud service/provider in a way that new the cloud service can contribute to quality goals. Define a set of suitability criteria that characterise desirable features of cloud providers. The criteria include provider profile (e.g. pricing model, constraints, offered QoS, electricity costs, power, and cooling costs), organisation migration characteristics (migration goals, available budget), and system requirements. Based on the criteria, identify and select suitable cloud providers. | Applicable to resolve all obstacles | Inspired from KAOS methodology and [S65], [S79] | Goal/Service/Migration type Substitution, Obstacle reduction |

| | | | | | |
|---|---|---|---|---|---|
| T4 | Analyse migration feasibility | Perform a feasibility analysis to evaluate the benefits and the consequences of moving legacies to the cloud and its impact on organisation structure, staff’s roles, and legacies. | O38, O39 | [S73], [S74] | Obstacle prevention |
| T5 | Refactor legacy source code | Adapt the source code for being compatible and able to interact with the selected cloud platform programming language and APIs. | O19, O20, O21, O22, O23, Q24, O25 | [S65], [S66], [S67] | Obstacle prevention |
| T6 | Develop adaptor/wrapper | Add adaptors for resolving mismatches, occurring at runtime system execution, between legacy system components and cloud services. | O19, O20, O21, O22, O23, Q24, O25, O36 | [S75], [S76] | Obstacle prevention |
| T7 | Decouple system components | Decouple the legacy system components from each other. Use mediator and synchronisation mechanisms to manage interaction between the loosely coupled components in the cloud environment. | O51 | [S12], [S77], [S78] | Obstacle prevention |
| T8 | Encrypt/decrypt message passing | Add support for the runtime encryption/decryption of message transition between components in the on premise network and cloud environment. | O26 | [S12], [S75], [S79] | Obstacle prevention |
| T9 | Obfuscate code | Protect unauthorised access to code blocks of components by other tenants that are running on the same cloud provider. Use encryption mechanisms in the sense that no other tenants will be able to access, read, or alter the code blocks with the components when running in the cloud. | O8, O16 | [S6] | Obstacle prevention |
| T10 | Isolate tenant | Enable multi-tenancy in the system. Based on multi-tenancy requirement (i) define tenant-based identification and hierarchical access control for tenants and (ii) separate tenant data using authorization and authentication mechanisms. | O4 | [S80], [S81], [S96], [S97] | Obstacle prevention |
| T11 | Tune message granularity | Define suitable granularity for messages, which are passing between components hosted on local network and the cloud, based on the degree of functionality that is offered to the service consumer and consumer's infrastructure capability to process the messages. A proper message granularity can be identified or predicted based on pieces of data actually used by system or using heuristic functions to understand the number of interaction between system components over the cloud network. | O26 | [S12], [S82] | Obstacle prevention |
| T12 | Adapt data | Convert legacy data types to the data type of target cloud database solution. Also, add an extension component to the legacy which includes a set of commands to be performed by the legacy or cloud. The emulator supports missed database functionalities of cloud database solution provider. | O50, O21 | [S12], [S38], [S71], [S83], [S84] | Obstacle prevention |
| T13 | Involve staff with cloud adoption process | Involve staff and stakeholders actively in the cloud adoption process and give them insight of benefits of the cloud and organisational change. | O38, O39 | [S46] | Obstacle reduction |
| T14 | Define an authorization | Add a component determining if a tenant has privilege to perform a given action over the database. | O4 | [S69] | Obstacle prevention |
| T15 | Encrypt data | Use data encryption mechanisms prior outsourcing or hosting system data to the cloud. | O14, O13, O4 | [S12], [S79], [S85] | Obstacle prevention |
| T16 | Filter unauthorised requests | Add support to filter unauthorized data access received from users at the edge of premise or cloud network as early as possible to avoid unauthorized network traffic. | O14, O4 | [S58] | Obstacle prevention |
| T17 | Adjust security policies | Add support for runtime security assessment of received queries for run on data. | O14, O4 | [S58] | Obstacle prevention |
| T18 | Replicate system components | Partition, replicate, and distribute system components and data (replicas) on multiple cloud servers. | O3, O6, O27, O45, O28 | [S58], [S78], [S86] | Obstacle reduction |

| | | | | | |
|---|---|---|---|---|---|
| T19 | Backup periodically | Implement a procedure to periodically perform data backup. | O4, O14, O15, O17 | [S71], [S72] | Obstacle prevention |
| T20 | Detect and filter intrusions | Filter unauthorised packets and malformed data traversed between system components in local network and the cloud environments. | O4, O8, O9 | [S58], [S70] | Obstacle prevention |
| T21 | Update patches | Perform regular patch update across system components in the cloud. | O7, O8, O9, O46 | [S74], [S87] | Obstacle reduction |
| T22 | Isolate tenant | Protect tenants' data from to be accessed by other tenants. Each tenant should be authorised and able to access to its own data. | T7 | [S88], [S89], [S90] | Obstacle prevention |
| T23 | Define retry policies | Define retry policies and implement them in the system for the operation to succeed. | O3 | [S66], [S75] | Goal restoration |
| T24 | Refine network topology | Define a proper network topology with a consideration of proximity of servers and system components, proper provider equipment, the location of the data centres, router hops, and infrastructure bandwidth. | O27, O28, O47, O45 | [S65], [S66], [S74], [S77], [S78] | Obstacle reduction |
| T25 | Examine cloud service behaviour | Use benchmarking tools to investigate performance of the cloud under investigation before decision making. | O27, O11, O12, O17 | [S32], [S65] | Obstacle prevention |
| T26 | Acquire more cloud resources | Rent more VMs or higher spec ones to deal with slow CPU clock rates, use physical disk shipping to reduce effects of network latency/transfer rates. | O27 | [S2], [S92] | Obstacle reduction |
| T27 | Use multiple cloud servers | Deploy and replicate system components in several clouds. | O27 | [S45] | Obstacle reduction |
| T28 | Add intermediation | Implement an intermediate layer (mediator components) between legacy system and cloud services that decouple legacy systems from cloud specific APIs. This helps to create intermediate APIs and get indirect service from the cloud. | O6, O29, O47 | [S63], [S64] | Obstacle prevention |
| T29 | Make system stateless | Provide support in the system to the handle safety and traceability of tenant’s session when various system instances are hosted in the cloud. | O43 | [S78], [S91] | Obstacle prevention |
| T30 | Prioritize tests | Perform test cases on the basis of their importance and criticality. | O32 | [S95] | Obstacle prevention |
| T31 | Resolve licensing issue | There alternative sub-tactics: (i) negotiate with system owner to make a suitable licensing model which satisfies all parties, (ii) extend legacy system with a new component (e.g. VPN tunnel) in a way that cloud services can be indirectly offered to them, and (iii) enable a license tracking mechanism through monitoring connections between the software system and cloud resources. | O37 | [S72], [S74] | Obstacle prevention |
| T32 | Define weak inconsistency | Implement an eventual consistency or similar weak consistency model for data. | O52 | [S8] | Obstacle reduction |
| T33 | Check compliance | Check if cloud adoption is compliance with the auditors and cloud providers. | O40 | [S45], [S57] | Obstacle prevention |
| T34 | Clarify roles | Clarify roles and responsibilities relevant to cloud adoption. | O38, O39 | [S40], [S45] | Obstacle reduction |
| T35 | Aware top-level management | Make management aware of the extra effort that might be required for cloud adoption in the organisation. | O31, O33, O35, O38, O39, O41 | [S94] | Obstacle reduction |
| T36 | Degrade goal | Resolve an obstacle by degrading goal definition and refining its assumption for required levels of satisfaction so that the refined goal makes have more freedom for violation. | Applicable to resolve all obstacles | KAOS methodology | Goal weakening |
| T37 | Restore goal | Add a new goal for restoring the satisfaction of the obstructed goal when violated. | Applicable to resolve all obstacles | KAOS methodology | Goal restoration |
| T38 | Mitigate goal | Add a new goal for mitigating the consequences of an obstacle if it occurs. | Applicable to resolve all obstacles | KAOS methodology | Goal mitigation |
| T39 | Fix inconsistencies | Perform manual or semi-automate steps to resolve inconsistencies which have | O52 | [S8] | Goal mitigation |

| | | | | | |
|---|---|---|---|---|---|
| | | occurred after data operations. | | | |
| T40 | Define compensation | Specify penalties (e.g. financial or getting more quote) to be paid by cloud provider in the case of a disruption. | O1, O2 | [S2], [S3], [S4] | Goal mitigation |
| T41 | Do nothing | Leave obstacle unresolved. | Applicable to resolve all obstacles | KAOS methodology | Do nothing |
| T42 | Use rigorous authentication | Use strong passwords and authentication mechanism when running system in cloud environment. | O53 | [S104] | Obstacle prevention |
| T43 | Keep virtualization at the system level | Create virtualization and isolation boundary at the legacy system level rather than at the server level through container concept. Such a container (i) handles resource allocation meaning that in the case of excessive resource consumption by a system operating in the cloud, only individual container is affected and whole virtual machine is left unaffected and (ii) reduces incompatibility problems between applications across multiple platforms. | O24, O66, O67 | [S110] | Obstacle prevention |
| T44 | Use dedicated virtual machine | Run the legacy system on dedicated virtual machine in the sense that the virtual machine is entirely performed on separate resources such physical servers, network, switch, bandwidth, disk, CPU, memory to satisfy expected quality of service. All resources are physically dedicated to the virtual machine. | O4, O6, O8, O9, O13, O14, O18, O27, O29, O53, O58, O59 | [S111] | Obstacle reduction |
| T45 | Define bidding strategy | Identify heavy processing tasks of the legacy system (e.g. image, video, conversion and rendering) and define a bidding strategy for spot instance to lessen the cost of using cloud resources. | O54 | [S68] | Obstacle prevention |