International Journal of Automation and Computing 11(2), April 2014, 143-152
DOI: 10.1007/s11633-014-0776-7

Strategies and Methods for Cloud Migration

Jun-Feng Zhao Jian-Tao Zhou
College of Computer Science, Inner Mongolia University, Hohhot 010021, China

Abstract: Legacy system migration to the cloud brings both great challenges and benefits, so there exist various academic research and industrial applications on legacy system migration to the cloud. By analyzing the research achievements and application status, we divide the existing migration methods into three strategies according to the cloud service models integrally. Different processes need to be considered for different migration strategies, and different tasks will be involved accordingly. The similarities and differences between the migration strategies are discussed, and the challenges and future work about legacy system migration to the cloud are proposed. The aim of this paper is to provide an overall presentation for legacy system migration to the cloud and identify important challenges and future research directions.

Keywords: Migration, legacy system, cloud, infrastructure as a service (IaaS), platform as a service (PaaS), software as a service (SaaS).

1 Introduction

Clouds are a large pool of easily usable and accessible virtualized resources such as hardware, development platforms, and services[1]. These resources can be dynamically re-configured to adjust to a variable load, allowing for optimal resource utilization[2]. According to the virtualized resources, three cloud service models emerged: infrastructure as a service (IaaS), platform as a service (PaaS) and software as a service (SaaS)[3−5]. The business model of the cloud is pay-as-you-go, so enterprises can reduce capital expenditure by leveraging the cloud service[6−8]. Based on these characteristics, cloud computing provides many benefits needed by enterprise, such as no up-front investment, lower operating cost, high scalability, and so on[9]. Software migration is the process of switching from one operating environment to another that in most cases is considered to be better[10]. A legacy system is an outdated computer system that remains in use even after more modern technology has emerged either because the organization may have invested considerable time and money in it or the legacy system holds valuable data. In order to take advantage of cloud computing and protect the existing investment to legacy system, enterprise are eager to migrate legacy systems to the cloud. As a result, the study in academia and the practice in industry on migrating legacy systems to cloud computing are very widespread today.

So far, much research has been done in this field. Some works[11−15] focus on decision making support for cloud migration in enterprise, since benefits, risks, costs, and organizational and socio-technical factors must be considered before migration. More researchers concentrate on migration methods, mainly about how to efficiently migrate legacy system to the cloud[16−21]. Some organizations devote themselves to providing development tool for migrating legacy system to the cloud[22−25]. Up to now, some innovative methods have been proposed, related tools have been developed, and lots of organizations have made some trials in migrating legacy systems to cloud computing.

In this paper, we review the research achievements and application status on cloud migration, make a comprehensive analysis of existing migration methods, classify them into five migration strategies, and finally identify the corresponding considerations of each respective strategy. Migration strategy is used to classify cloud migration on the whole. According to the cloud service model, the migration to the cloud is classified into three strategies integrally in this paper. Different migration strategy focuses on different tasks, fits special migration scenarios and possesses characteristic virtues. To the practitioner of migration, the analytic results above will benefit them. In the end, we present some challenges and research topics for future, which may provide some guideline to the research beginner in cloud migration.

The rest of this paper is organized as follows. In Section 2, existing migration strategies are reviewed, and a classification is proposed. Section 3 surveys the existing methods of migration to IaaS and PaaS, and analyzes the similarities and differences of tasks in these two strategies. Section 4 investigates the work about migration to SaaS and compares the proposed migration strategies from different aspects. The related development tools are reviewed in Section 5. Section 6 discusses the challenges and future research topics in cloud migration. Finally, the paper concludes in Section 7.

2 Migration strategy

With regard to the migration classification, there exist different classification cases in different literature. Binz et al.[17] classified the migration into three types: Standardized Format Migration, Component Format Migration and Holistic Migration. The component implemented in standardized, self-contained format is migrated in Standardized Format Migration such as VMware or Open Virtualization

Manuscript received July 31, 2013; revised December 30, 2013
This work was supported by National Natural Science Foundation of China (No. 61262082), Key Project of Chinese Ministry of Education (No. 212025), Inner Mongolia Science Foundation for Distinguished Young Scholars (No. 2012JQ03), Inner Mongolia Natural Science Foundation of Inner Mongolia (No. 2011MS0911), Postgraduate Scientific Research Innovation Project of Inner Mongolia (No. 1402020201013).

is Format images. The format of the respective component transformed into another format in the second type, for example, transforming a virtual machine image or enabling the execution of scripting languages on PaaS. Holistic migration aims to realize migration to complete application built out of multiple components by migrating each component separately. According to the holistic migration, the authors proposed the cloud motion framework that could leverage existing application models and provide support to migrate composite applications to cloud[17]. Reference [18] identified four migration types that could cloud-enable applications by adaptation. The first type replaces components with cloud offerings, which is the least invasive type of migration. The second describes the case that migrates some of the application functionality to the cloud. The third is the classic migration case where the whole software stack of the application is migrated to the cloud. The last is complete migration of the application, which requires the migration of data and business logic for the cloud. While Gartner[26] suggests information technology (IT) organizations consider the following five options when they seek to move legacy systems to the cloud: rehost on infrastructure as a service, refactor for platform as a service, revise for IaaS or PaaS, rebuild on PaaS and replace with software as service. However, Cisco considers three application migration options including SaaS, PaaS and IaaS in white paper for migration of enterprise application to the cloud. They think the migration to SaaS is no longer an application migration but more of a replacement of the existing application with a SaaS. Migration to PaaS is an option for migrating business applications that are based on standard application server software such as JavaEE or .net platforms. Migration to IaaS involves deploying the application on the cloud service provider′s servers. In addition, the criteria that are used for considering every application migration are discussed[27]. Similarly, Solentive software proposed three main approaches for migrating legacy system to the cloud in a white paper, namely Iaas, PaaS and SaaS. The white paper looks at these approaches in detail and analyzes the benefits and disadvantages of each[28].

Through comparing and analyzing, we can categorize the migration into three strategies integrally: migration to IaaS, migration to PaaS and migration to SaaS. The first strategy implements migration only by porting legacy system to the cloud by using IaaS. The legacy system will be migrated to the cloud by system refactoring according to the platform of PaaS in the second strategy. As to the migration to SaaS, it can be divided into three sub-strategies concretely, namely replacing by SaaS, revising based on SaaS and reengineering to SaaS. To the first sub-strategy, legacy systems will be completely replaced by commercial software delivered as a cloud service. Based on the second sub-strategy, some functionality of legacy systems will be replaced by cloud service. To the third sub-strategy, legacy systems will be reengineered to cloud service. In reality, enterprises often migrate their legacy systems to cloud platform by adopting the first strategy. To this strategy, the migration is relatively easy to implement and has good cost-benefit. But the migration could not take full advantage of cloud platform. To the second strategy, legacy systems need to be adapted according to the target platform, which can bring disadvantages include missing capabilities, transitive risk, and framework lock-in. To the SaaS related strategy, if legacy system is replaced by commercial software delivered as a service, the migration effort will be reduced greatly and reengineering is unnecessary. When replacing some business logic with existing cloud service, the adaption to the legacy system is necessary. But to reengineer legacy system to cloud service, the related work will be very challenging, and may require reverse engineering, structure redesign, service generation, and so on. After comparing and analyzing, we map the migration methods mentioned above to these five specific strategies. The mapping result is showed in Table 1.

The proposed strategies realize classification to cloud migration in essence, which can cover all migration cases. Through mapping existing migration methods to the proposed strategies, the migration methods can be better understood and compared explicitly.

3 Migration to IaaS and PaaS

3.1 Migration to IaaS

Infrastructure as a service is a form of hosting, which includes network access, routing services, and storage. IaaS provider generally provides hardware and administrative services used to store applications and a platform for running applications. A virtual machine is built for an application, which is loaded with all the software that will eventually run in the cloud. Then the virtual machine is uploaded to IaaS vendor′s hosting environment and deployed to run. IaaS is the best choice for moving applications to the cloud when there is no time to reengineer the applications for a cloud[29].

With respect to migration method, cloud computing service providers, such as Amazon and Cisco, provide the details for migrating legacy systems to their platforms[27, 30]. Zhang[31] conducted migration studies by migrating Hadoop and RUBiS to EC2 cloud platform.

Table 1 Mapping existing migration methods to proposed strategies

	Migrate to IaaS	Migrate to PaaS	Replace by SaaS	Revise based on SaaS	Reengineer to SaaS
Reference [17]	type I, type III	type II, type III	–	type III	–
Reference [18]	type III	type I,type II	–	type I, type IV	type IV
Reference [26]	rehost, revise	refactor, revise, rebuild	replace	–	–
Reference [27]	to IaaS	to PaaS	to SaaS	–	–
Reference [28]	to IaaS	to PaaS	to SaaS	–	–

Through experiments, the installation mistakes and configuration errors were identified as the two major sources of errors in migration. Then a migration management framework was proposed for providing the installation automation and configuration validation, which uses templates to simplify the largescale enterprise system installation process and uses policy to validate the configuration and monitor the configuration changes[31].

Besides technology domain need to be taken into account, there are a number of factors to weigh before migrating legacy system to IaaS such as meaningful prerequisites, financial and legal considerations, culture shift, and so on. If these conditions cannot be satisfied, migration of legacy system to IaaS will be harder[32]. By presenting the result of a case study that investigated the migration of an IT system from a company′s in-house data center to EC2, reference [16] discussed the implications of migrating legacy systems to the cloud from the perspective of an enterprise or organization. In addition to the benefits of using the cloud, the significant risks, such as deterioration of customer care and service quality, increased dependence on external third party, lack of supporting resources, and so on, will come up. Enterprise decision-makers should consider the overall organizational implications of the changes brought by migrating legacy system to the cloud, so they can avoid implementing local optimizations at the cost of organization-wide performance.

Many general instructions have been proposed to provide guidance for application migration to IaaS. For example, Kothari and Arumugam[33] introduced guidelines to assess the feasibility of migrating applications to the cloud and suggested a general migration strategy for applications. Sattaluri[34] discussed different aspects that need to be considered during the application migration to IaaS. While Yunus[35] presented costs and risks of application migration to IaaS, Mossburg listed four important points that lead to a successful cloud migration. These works provide general instructions for migration to IaaS. Reference [36] gave further details about what to do and how to do for application migration to IaaS. The existing research mentioned above can be divided into two aspects on the whole, namely migration method and migration decision. We present Table 2 to show the focus of the existing research.

Table 2 Focus of the existing research in migration to IaaS

	Migration method	Migration decision
Reference [27]	+	−
Reference [30]	+	−
Reference [31]	+	−
Reference [32]	−	+
Reference [16]	−	+
Reference [33]	+	+
Reference [34]	−	+
Reference [35]	−	+
Reference [36]	+	−

In short, users have full privileges on the allocated virtual machine (VM) in migration to IaaS. They could do anything to the VM, but there is something need to be considered before conducting migration:

1) Dynamic resource requirement.
2) Restriction to data storage location.
3) Requirement of special hardware devices.
4) Amount of data stream.

3.2 Migration to PaaS

Platform as a service is an application development and deployment platform delivered as a service to developers over the web, which provide the hardware plus a certain amount of application software such as databases, middleware, and development tools. Migration based on PaaS is not mandatory for resource management, but it is required to make the legacy system compatible to the requirement of PaaS provider.

Microsoft, Cisco, and Solentive provide guide for migration to PaaS from technology domain[27, 37, 38], but these guidelines are restricted to the PaaS they provide and not applicable to other general cases. For the general cases, reference [36] specified further checking steps for application migration to PaaS, including programming language, database, restrictions and limitations of the selected PaaS besides checking specific requirements related to hardware, software and input data as discussed in migration to IaaS. Furthermore, the general solutions to solve incompatibility issues of database migration were discussed. Tran et al.[39] defined the scope of migration software system for the cloud. They identified all activities in migration that start from getting familiar with the application, the target cloud platform, and the third party tool, then to build the environment and get ready for migration, as well as to modify and test to ensure that the application properly runs in the cloud. Finally based on the experience of migrating PetShop. Net to Windows Azure and migrating Java Pet Store to Amazon EC2, they compared the difference between migrating legacy system to PaaS and to IaaS.

In brief, PaaS supplies a complete cloud IT stack for software development and delivery, which makes it possible to build "true" cloud applications and release them in a scalable and elastic environment. Moreover, it also produces numerous restrictions at every technology layer of the application stack:

1) Programming language.
2) Database.
3) Middleware.
4) Third party library.
5) Restriction of the selected PaaS.

4 Migration related SaaS and comparison among strategies

SaaS is a software delivery model in which software and associated data are centrally hosted on the cloud. SaaS is typically accessed by users using a thin client via a web browser, which has many advantages over the traditional software delivery model. The recurring revenue stream, simpler maintenance and application update, and the lower cost of delivery and distribution are especially attractive for both application provider and end users. These advantages are driving a growing number of organizations of all sizes to adopt SaaS solutions in order to achieve their business

objectives.

The migration strategy through replacing legacy system by SaaS does not refer to reengineering the legacy system, and the only work needs to be done is to export local data to the cloud database. For the migration strategy of revising based on SaaS, some functionality of legacy system will be outsourced to the cloud, and then the business process is used to integrate cloud and non-cloud services. There are significant advances in service and data integration[40, 41] and service composition[42, 43] in service oriented architecture (SOA). For the migration strategy of reengineering to SaaS, SOA and Cloud need to be considered jointly. The SOA and Cloud are enabling technologies for each other. SOA decomposes the whole architecture into coarse-grain components providing business services. The scalability and possibility to multiply more instances of one component can be obtained by decomposing the architecture, which fits rightly to the Cloud deployment paradigm. On the contrary, due to network latencies in the cloud environment, developer must ensure that the application can run on a set of low performance resources. This necessitates decomposition and decoupling of the application architecture, which is similar to SOA paradigm. So, SOA and Cloud are perfectly complementary. SOA provides the guideline for making the software architecture scalable, while Cloud enables deployment of architecture scale[44]. So the convergence of cloud computing and SOA supports the development of SaaS. When reengineering a legacy system to SaaS, migrating the system to SOA necessitates in general. Reference [45] stated that recasting existing systems into services is a prerequisite to move towards cloud computing.

4.1 Migration to SOA

SOA is a set of principles and methodologies for designing and developing software in the form of interoperable services. These services are well-defined business functionalities and built as software components that can be reused for different purposes. SOA can expose the functionality of the legacy systems as services, presumably without making significant changes to the legacy systems, so migration of legacy system to service-oriented environments has significant value. So far researchers have paid much attention to migration of legacy systems to SOA. A lot of migration methods have been proposed, which can be roughly classified into three perspectives. The first concentrates on planning the migration. The feasibility of migration process is assessed based on business drivers and characteristics of legacy system. The second focuses on implementing the migration which usually covers techniques to adapt a segment of legacy code to web services. The last perspective aims for fulfilling the whole migration process that is comprised of two major steps: forward service development and reverse service extraction.

The representative work of the first perspective is service migration and reuse technique (SMART). Carnegie Mellon Software Engineering Institute developed SMART to help organizations make initial decisions about the feasibility of reusing legacy components as services within an SOA environment. SMART considers the specific interactions that will be required by target SOA environment and any changes that must be made to the legacy components[46]. Reference [47] presented an overview of the application reengineering decisions in SOA, which identified the key issues involved in each decision, and proposed a decision model for application reengineering for SOA.

Zhang et al.[48] proposed an architecture-based service-oriented approach to modernize legacy system in a service-based computing environment. The approach emphasizes service identification and packaging. The modernization of the legacy system is implemented in five steps, including legacy system evaluation, architecture recovery, service identification, service packaging and service publication; Oldevik et al.[49] proposed a model-driven approach for migrating legacy systems to service-oriented architecture. The migration strategy is implemented by wrapping existing legacy components. Unified modeling language (UML) is used to specify migration models, or wrappers, that are fed to model-driven code generators to generate a deployable service. The migration approach was tested on an oil drift prediction system by modeling and generating the services and wrappers required for integration with the various legacy components[49]. These works does not involve re-factoring or re-engineering, they only focus on the implementation of migration by wrapping.

Reference [50] proposed a service-oriented software reengineering (SoSR) methodology that can be applied to reengineer a legacy software system into a service-oriented software system. The methodology is formed by a set of best practices that is architecture-centric, service-oriented, role-specific, and model-driven. Winter and Ziemann[51] proposed an approach toward migrating legacy systems to service oriented architectures by relating the intended enterprise model to the legacy software system. The approach is model-based, where appropriate models represent software artifacts on all levels of abstraction. The activities in the migration process, such as reverse engineering, enterprise modeling, forward engineering, and legacy migration, rely on these models that serve as base for extracting abstractions or transformations. These approaches cover the whole migration process.

Although there is much work on SOA migration, they mainly differ in the way they provide solutions for two key problems, namely what can be migrated and how the migration is performed. In order to understand, compare, and analyze the existing methods, a conceptual framework for SOA migration was proposed by Maryam Razavian and Patricia Lago, which provides a holistic illustration of the SOA migration process, along with the distinct conceptual elements of the process. SOA-MF addresses the question of "what does the migration of legacy systems to SOA entail". The migration process follows the horseshoe model by first recovering the lost abstractions and eliciting the legacy fragments that are suitable for migration to SOA, then altering and reshaping the legacy abstractions to the service based abstractions, and finally renovating the target system based on transformed abstractions as well as new requirements. It could be performed at different levels of abstraction ranging from "code-level" to "enterprise-level"[52]. After analyzing the migration approaches mentioned above according to SOA-MF,

we present Table 3 that maps them to the abstraction tiers described in SOA-MF.

Table 3 Mapping existing migration approaches to abstraction tier

	Code	Basic design	Composite design	Concept
Reference [41]	−	−	−	+
Reference [43]	−	+	+	−
Reference [44]	−	−	+	−
Reference [46]	−	−	−	+
Reference [48]	−	+	+	−
Reference [49]	−	+	+	−
Reference [50]	−	+	+	+
Reference [51]	−	+	+	+

4.2 Migration related SaaS

In order to provide some guidance on modifying architecture of a service-based system for cloud computing, Babar and Chauhan[20] revealed some insights when they undertook a research and development project aimed at modernizing an open source software framework, Hackystat, for leveraging the flexibility and scalability of the cloud computing paradigm. In the process of migration, the key requirements that can be migrated to the cloud are identified firstly. Having identified the key requirements, the next task is to analyze those requirements in the context of the existing architecture in order to gain a good understanding of the kinds of architectural change needed to be made. Finally, the architecture will be modified in accordance with the identified architecture change. Because this work focused on the migration of the service-based system to cloud computing platform, it did not involve reverse engineering.

Zhang et al.[19] proposed a generic methodology to guide how to migrate legacy system to cloud platform. The generic methodology includes the following steps: representation of the legacy application, redesign the architecture model with identified services, model driven architecture (MDA) transformation, web service generation, invocation of legacy functionalities, selection of a suitable cloud computing platform, and provision of cloud web service to the end users. This seven-step methodology guides developers to migrate legacy systems step by step and improves the productivity and effectiveness of migration.

In addition to the migration methodology, some research on the implementation of migration to SaaS was carried out. Current approaches are often limited to specific cloud environments or do not provide automated support for the alignment with a cloud environment. Frey and Hasselbring[53] proposed a model-based approach CloudMIG for migrating legacy software system to scalable and resource-efficient cloud-based application. CloudMig concentrates on the SaaS provider perspective and facilitates the migration of enterprise software system towards generic IaaS and PaaS-based cloud environment. CloudMIG can generate considerable parts of a resource-efficient target architecture utilizing a rule-based heuristics, so it helps SaaS providers to semi-automatically migrate existing software to the cloud computing platform. Li[21] looked into the migration towards the cloud in his dissertation for master′s degree. The research focuses on the derivation of legacy structure, the redesign of the system structure and deployment of the redesigned system. A framework MOMISC was presented, which offers a set of solutions and tools to support migrating existing systems to the cloud. Based on the framework, a hotel booking application was migrated to EC2 for verifying the feasibility of the framework.

Through investigation of existing research, reuse and migration of legacy applications to interoperable cloud services (REMICS), a research project supported by the European Commission that started in 2010 for a period of three years, has identified further work that needs to be done in the project. Specifically, knowledge discovery involves business and rule recovery that are necessary for identifying services and designing new business processes except reverse engineering of legacy code. Comprehensive methodology is employed to address dedicated design patterns and transformations, especially for migration to the cloud. Migration tools and methods need integration with model-based development methods. PIM for SOA and cloud computing can adapt many different platforms in the cloud and diverging technologies. These problems have been partially addressed by developing methods, languages, transformations and tools. The REMICS methodology will integrate all these in an agile, model-driven, and service-oriented methodology for modernizing legacy systems in the end. The migration starts with the recover activity which extracts the architecture of the legacy application. The recover activity is implemented by knowledge discovery and reverse engineering. The source architecture helps to analyze the legacy system, identify the best ways for migration, and benefit from model-driven engineering technologies for generation of the new system. The migrate activity reforms the source architecture into target architecture for service cloud platform. In migration activities, SOA and cloud computing patterns are applied, legacy components may be replaced or wrapped, architecture may be redesigned by service composition. After obtaining target architecture, the service cloud implementation is achieved by MDA. In short, REMICS proposes a leap progress in legacy systems migration to service clouds, which significantly improves the baseline ADM concept[44, 54−56]. This project involves the most advanced research in migrating legacy system to service cloud now.

In summary, migration to SaaS requires to consider the specific migration strategy according to legacy system and existing SaaS. If existing SaaS has the same business functionality of legacy system, users can replace legacy system by SaaS. When some business functionality has been realized by existing SaaS, legacy system can be modernized by revising legacy system based on existing SaaS. Users can take full advantage of the virtue of the cloud by reengineering legacy system to SaaS, but the challenge cannot be ignored.

4.3 Comparison between the migration strategies

The most ideal strategy for a special migration to the cloud relies on the individual needs of each organization and the condition of the legacy system, so organizations need to choose the most rational strategy before the migration.

With the PaaS strategy, organizations need to determine the extent of modifications required for the application to be compatible with the cloud platform. If extensive modifications are needed for PaaS, the IaaS strategy may be a better choice. As an effective software-delivery mechanism, SaaS could be an ideal choice for independent software vendors, but the challenge for reengineering to SaaS is obvious. Through integral analysis, we present Table 4 to compare the characteristics of these five migration strategies.

5 Related development tools

MoDisco is a generic and extensible open source reverse engineering solution, which intensively uses MDE principles and techniques to improve existing approaches for reverse engineering. In comparison to many development tools that focus on UML generation from a specific technology and vice versa, MoDisco provides generic support for different target meta models and extensibility to other technologies[22]. MoDisco supports four use-cases of existing software modernization. The first use-case is quality assurance which aims at verifying whether an existing system meets the required qualities. The second is documentation which focuses on the extraction of information from an existing system to help understand one aspect of the system. The third is improvement which concentrates on transformation of an existing system to integrate better coding norms or design patterns. The last is migration which implements transformation to the component, the framework, the language, or the architecture of legacy system. In each case, modernizing an existing software system includes three phases. Firstly the information should be extracted out of the artifacts of the system. Then the extracted information will be understood in order to take good modernization decisions. Finally the information is transformed to new artifacts which may be metrics, document, code and so on[23].

Blu Age is another agile solution for application modernization which focuses on extracting legacy architecture into a PIM presentation and regenerates it to a modernized system using MDA approach. Blu Age application is based on three complementary products. Blu Age reverse modeling extracts the business code of legacy application and transforms it into UML2 model independent of any technology. Blu Age database modernization migrates database technologies by modernizing data and strengthening their integrity. Blu Age forward engineering generates applications by compiling UML2 models. Blu Age engineering automates 100% of the developments and reduces costs significantly[24].

Modelio is an open source modeling environment which supports SoaML, the SOA architecture modeling standard, with specific editors dedicated to SOA architecture modeling and architecture implementation model generation. Modelio′s core architecture is based on a meta-meta infrastructure. The core architecture supports extensibility mechanisms, notably for UML profile definition, and provides the concept of modules, which package extensions and can be dynamically applied to or withdrawn from an existing model[25].

Modisco and Blu Age are both ADM compliant case tools which can extract architecture model from legacy system. The extracted model will be the start point of the following forward engineering. Modelio imports the recovered system model and generates SOA models through componentization and refactoring. SOFTTEAM developed a series of migration tools based Modelio environment. Now the link between Blu Age and Modelio is fully functional.

6 Challenges and future work

Based on the above comparison and analysis, we identified the following challenges that could be research topics in the future.

1) Holistic methodology

Up to now, the existing research mostly focused on the consideration of a special migration approach. A holistic methodology of migrating legacy systems to the cloud is needed. Firstly, the migration of legacy systems to the cloud should be divided into reasonable types, the Cloud providers and legacy systems should be classified, too. Then, according to a specific kind of legacy system, the selection of Cloud providers, the migration type to be applied, and the required adaption for the migration should be made. At the same time, the advantages and risks accompanying the migration should be advertised. Based on the information, organizations can make decision to migration strategy. Finally, the methodology should propose adaption processes in accordance with the given migration. With the guidance of this holistic methodology, organization can efficiently migrate legacy systems to the cloud, rather than hesitate about what to do and how to do when facing a migration task.

2) Scalability in IaaS

Table 4 Comparison between the migration strategies

	Migrate to IaaS	Migrate to PaaS	Replace by SaaS	Revise based on SaaS	Reengineer to SaaS
Migration workload	Little	Moderate	Little	Moderate	Much
Migration complexity	Easy	Moderate	Easy	Moderate	Difficult
Adaption	No need	Modify application to be compatible with PaaS	No need	Service and data integration, service composition	Reverse engineering, redesign structure, forward engineering
Effect	Save capital expenditure for hardware	Free from resource management	Flexible pricing mechanism, convenient maintenance	Flexible pricing mechanism, convenient maintenance, reuse	Flexible pricing mechanism, convenient maintenance, reuse, scalability

For the migration to IaaS, if the amount of resources required by the application is stable, it is unnecessary to adapt the application for migration. However, for some applications, resource requirements might vary significantly from time to time, organization must implement a new resource management component so that resources could be acquired and released automatically on demands. This component can efficiently optimize resource consumption in the cloud. The discussion on how to realize resource management is absent at present, so related solutions should be identified to solve this problem.

3) Usage of essential cloud characteristics

For the migration to PaaS, in addition to the adaption to programming languages, databases and third-party components, organization needs to consider redesigning some business processes in order to exploit essential cloud characteristics. For example, PaaS provides MapReduce programming model which is a simple data-parallel programming model designed for scalability and fault-tolerance. MapReduce can automatically parallelize and execute the program on a large cluster of commodity machines. When the legacy system involves processing of large data sets, the related business process needs to be redesigned for migration in accordance with MapReduce programming model. Nevertheless, existing work have not paid enough attention to take full advantage of the essential cloud characteristics.

4) Architecture refactoring

To reengineer legacy system to SaaS, the architecture of existing legacy applications need to be refactored using design patterns for SOA and cloud computing. SOA and cloud computing technologies complement each other. SOA enables software architecture better scalability and reuse of application components. While the Cloud computing addresses the deployment architecture that easily scale. SOA has been paid much attention for several years, and currently it is in an established state. On the contrary, there is little research in cloud design pattern due to the novelty of the domain.

5) Integrated development environment

For the migration to SaaS, it refers to an integral reengineering. Good tools are prerequisite to successful execution of a job. An integrated development environment can promote migration efficiency, which should incorporate model recovery tool, migration process tool, component recovery tool, pattern composition tool and service generation tool. Model recovery tool is used to extract model from the legacy system. Migration process tool offers support for the approach of migrating legacy systems to SOA application deployed on cloud computing platforms. The component recovery tool is responsible for component identification, discovery of services, interfaces and dependencies. The pattern composition tool adopts SOA and cloud computing design patterns to source architecture, which implements the iterative refactoring of legacy architecture to the targeted service cloud architecture. Service generation tool automates the target code framework or target code generation according to MDA. A series of tools are planned to be developed in REMICS, and some have been released. Since the links between these development tools are mostly manual, the integration between them is necessary. Otherwise more outstanding tools are beneficial to facilitate the migration of legacy systems to SaaS.

6) Other research topics

Besides migrating legacy system from traditional environments to cloud platforms, application migration between cloud platforms may be necessary in the future. So how to migrate applications from one cloud platform to another will be a research topic. In addition, testing strategies need to be studied to verify whether the generated system after migration has the same functionality and better performance as before.

7) Our ongoing work

Now we are focusing on migrating a workflow management system to cloud platform Aneka. Through migration, the existing workflow management system running in the in-house data center will be provided as SaaS to end users. The target workflow system will concentrate on scientific computing on large data, which can take full advantage of the virtue of the cloud. In the migration process, the essential cloud characteristics will be considered adequately. For example, Mapreduce programming model can be adopted to implement parallel processing of big data and to analyze the execution log of workflow system. In order to achieve the aforementioned objectives, we will employ reengineering to SaaS to carry out the system migration, and further explore the reengineering to SaaS and try to propose some innovative methods. The key work and challenges involved are as follows:

i) Software structure recovery

The workflow management system is implemented using an object-oriented language. We will discover the static model of this system using Modisco. After obtaining the static model, the classes will be grouped into layers through a link analysis algorithm, which usually include user interface layer, business logic layer, and data access layer. Because the business process of a system is realized in logic layer, the business logic layer needs to be further analyzed for service identification in migrating legacy system to SaaS. To further analyzing the logic layer, special clustering algorithm will be adopted for extracting function modules in legacy systems. By adopting the orthogonal structure analysis method, the obtained function modules through software structure recovery will be more accurate service candidates.

ii) Service identification

For the target service, we will conduct analysis from two aspects, namely, target system requirement and legacy system implementation. Based on requirement analysis of the target workflow system, some business functions will be identified which are appropriate for providing as services. The function modules in the legacy system implementation have been discovered in the first step. Based on these works, we will match the business functions with the function modules. The candidate services in legacy system and new services will be found. As a result, the function modules in legacy system can be efficiently reused.

iii) Model refactoring

After service identification, we need to refactor the existing software architecture to integrate them. Moreover, for some special candidate services such as those related to processing the big data, we need to recover the dynamic model of their implementation in the legacy system. Through an-

alyzing the dynamic model we will find the business logic that can be improved by the virtue of cloud computing and adapt the dynamic model. Then, we will reimplement the business process based on the cloud programming model so that the target system can take full advantage of the cloud characteristics.

Finally, we explicitly describe the research status on cloud migration by Fig. 1. The coordinate x represents migration strategies, and the coordinate y describes the related research corresponding to the specific migration strategies. The existing research fields are denoted by solid graph elements, and the unresolved research fields by hollow elements.

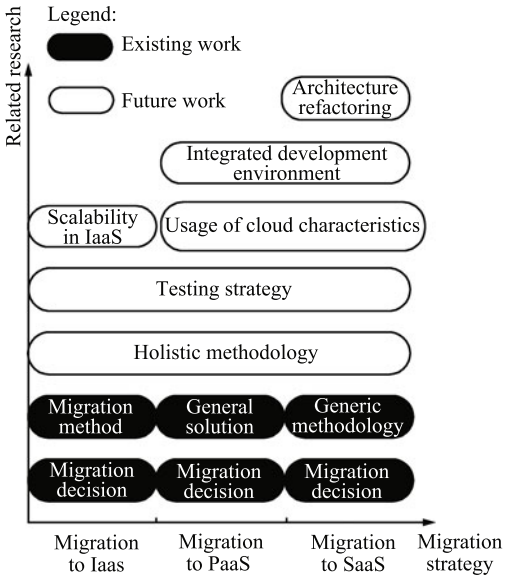

Fig. 1 The distribution of related research

7 Conclusion

Software migration is the process of moving legacy systems from one operation environment to another, that is, in most cases, thought to be better. Migrating legacy system to cloud computing can effectively protect software assets and take advantage of cloud computing. Many research projects have been carried out, and some innovative methods and tools have been proposed so far. By investigating the existing literature, we classify the migration into five strategies, then review and compare the related researches on every migration strategy. In addition, related development tools are surveyed. Based on the existing research achievements and application status, some future work are identified including holistic methodology, redesign and adaption to application for special migration, architecture refactoring, integrated development environment, and so on. Besides, we notice the most advanced research in migration of legacy software to cloud computing has been performed in REMICS, so some further work in this research field should be carried out based on the achievement obtained in this project.

References

[1] L. M. Vaquero, L. Rodero-Merino, J. Caceres, M. Lindner. A break in the clouds: Towards a cloud definition. *ACM Sigcomm Computer Communication Review*, vol. 39, no. 1, pp. 50–55, 2008.

[2] J. Geelan. *Twenty Experts Define Cloud Computing*, [Online], Available: http://cloudcomputing.syscon.com, July 18, 2008.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, M. Zaharia. Above the Clouds: A Berkeley View of Cloud Computing, Technical Report, Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, 2009.

[4] R. Buyya, C. S. Yeo, S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *Proceedings of the 10th IEEE International Conference on High Performance Computing and Communications*, IEEE, Dalian, China, pp. 5–13, 2008.

[5] C. Y. Low, Y. Chen, M. C. Wu. Understanding the determinants of cloud computing adoption. *Industrial Management & Data Systems*, vol. 111, no. 7, pp. 1006–1023, 2011.

[6] G. Gruman, E. Knorr. *What cloud computing really means*, [Online], Available: http://www.infoworld.com, July 18, 2013.

[7] Roy Bragg. Cloud Computing: When Computers Really Do Rule, [Online], Available: http://www.technewsworld.com, July 16, 2008.

[8] P. Watson, P. Lord, F. Gibson, P. Periorellis, G. Pitsilis. Cloud computing for e-science with Carmen. In *Proceedings of the 2nd Iberian Grid Infrastructure Conference Proceedings*, Porto, Portugal, pp. 3–14, 2008.

[9] Q. Zhang, L. Cheng, R. Boutaba. Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010.

[10] Definition Migration, [Online], Available: http://searchcio.techtarget.com, July 17, 2013.

[11] A. Khajeh-Hosseini, I. Sommerville, J. Bogaerts, P. Teregowda. Decision support tools for cloud migration in the enterprise. In *Proceedings of the 2011 IEEE International Conference on Cloud Computing (CLOUD)*, IEEE, Washington, DC, USA, pp. 541–548, 2011.

[12] A. Khajeh-Hosseini, D. Greenwood, J. W. Smith, L. Sommerville. The cloud adoption toolkit: Supporting cloud adoption decisions in the enterprise. *Software: Practice and Experience*, vol. 42, no. 4, pp. 447–465, 2012.

[13] C. Ward, N. Aravamudan, K. Bhattacharya, K. Cheng. Workload migration into clouds challenges, experiences, opportunities. In *Proceedings of the 2010 IEEE International Conference on Cloud Computing (CLOUD)*, IEEE, Miami, FL, USA, pp. 164–171, 2010.

[14] Cloud Adoption Assessment, [Online], Available: http://www.eduserv.org.uk, July 20, 2013.

[15] M. Menzel, R. Ranjan. CloudGenius: decision support for web server cloud migration. In *Proceedings of the 21st International Conference on World Wide Web*, ACM, New York, NY, USA, pp. 979–988, 2012.

[16] A. Khajeh-Hosseini, D. Greenwood, I. Sommerville. Cloud migration: A case study of migrating an enterprise IT system to IaaS. In *Proceedings of the 2010 IEEE International Conference on Cloud Computing (CLOUD)*, IEEE, Miami, FL, USA, pp. 450–457, 2010.

[17] T. Binz, F. Leymann, D. Schumm. CMotion: A framework for migration of applications into and between clouds. In *Proceedings of the 2011 IEEE International Conference on Service-Oriented Computing and Applications*, IEEE, Irvine, CA, USA, pp. 1–4, 2011.

[18] V. Andrikopoulos, T. Binz, F. Leymann, S. Strauch. How to adapt applications for the cloud environment. *Computing*, vol. 95, no. 6, pp. 1–43, 2013.

[19] W. Q. Zhang, A. J. Berre, D. Roman, H. A. Huru. Migrating legacy applications to the service Cloud. In *Proceedings of the 14th Conference Companion on Object Oriented Programming Systems Languages and Applications*, pp. 59–68, 2009.

[20] M. A. Babar, M. A. Chauhan. A tale of migration to cloud computing for sharing experiences and observations. In *Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing*, ACM, New York, USA, pp. 50–56, 2011.

[21] W. B. Li. Model Driven Migration of Legacy Systems to the Cloud, Master dissertation, Harbin Institute of Technology, China, 2010. (in Chinese)

[22] H. Bruneliere, J. Cabot, F. Jouault, F. Madiot. MoDisco: A generic and extensible framework for model driven reverse engineering. In *Proceedings of IEEE/ACM International Conference on Automated Software Engineering*, ACM, New York, USA, pp. 173–174, 2010.

[23] MoDisco, [Online], Available: http://www.eclipse.org/MoDisco/, July 22, 2012.

[24] Blu Age Legacy Application Modernization, [Online], Available: http://www.bluage.com/en, July 23, 2012.

[25] Modelio, [Online], Available: http://www.modelio.org, June 23, 2012.

[26] 5 Ways to Migrate Applications to the Cloud, [Online], Available: http://www.cioupdate.com, May 20, 2012.

[27] Planning the Migration of Enterprise Applications to the Cloud, Cisco White paper, 2010.

[28] Migrating to the Cloud 3 Main Approaches, Solentive White paper, 2011.

[29] S. Bhardwaj, L. Jain, S. Jain. Cloud computing: A study of infrastructure as a service (IAAS). *International Journal of Engineering and Information Technology*, vol. 2, no. 1, pp. 60–63, 2010.

[30] Migrating your Existing Applications to the AWS Cloud, Amazon White paper, 2010.

[31] G. Zhang. Data and Application Migration in Cloud based Data Centers: Architectures and Techniques, Ph. D. dissertation, Georgia Institute of Technology, USA, 2011.

[32] Infrastructure as a Service, [Online], Available: http://www.educause.edu, July 20, 2013.

[33] C. Kothari, A. K. Arumugam. Cloud application migration, [Online], Available: http://cloudcomputing.syscon.com/node/1458739, July 12, 2010.

[34] R. Sattaluri. Application migration considerations for cloud computing, [Online], Available: http://www.syscon.com/node/1686320, January 28, 2011.

[35] M. Yunus. Understanding enterprise-to-cloud migration costs and risks, [Online], Available: http://www.ebizq.net/topics/cloud_computing/features/12741.html, June 14, 2010.

[36] Q. H. Vu, R. Asal. Legacy application migration to the cloud: Practicability and methodology. In *Proceedings of the 2012 IEEE Eighth World Congress on Services*, IEEE, Washington, DC, USA, pp. 270–277, 2012.

[37] Microsoft. Moving applications to the cloud on the Microsoft Windows Azure Platform. *MSDN Magazine*, pp. 68–73, March, 2013.

[38] Microsoft. Tips for migrating your applications to the cloud. *MSDN Magazine*, pp. 36–45, August, 2010.

[39] V. Tran, J. Keung, A. Liu, A. Fekete. Application migration to cloud: A taxonomy of critical factors. In *Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing*, ACM, New York, USA, pp. 22–28, 2011.

[40] H. R. M. Nezhad, B. Benatallah, F. Casati, F. Toumani. Web services interoperability specifications. *Computer*, vol. 39, no. 5, pp. 24–32, 2006.

[41] A. Halevy, A. Rajaraman, J. Ordille. Data integration: The teenage years. In *Proceedings of the 32nd International Conference on Very Large Data Bases*, VLDB Endowment, pp. 9–16, 2006.

[42] S. Dustdar, W. Schreiner. A survey on web services composition. *International Journal of Web and Grid Services*, vol. 1, no. 1, pp. 1–30, 2005.

[43] A. Brogi, S. Corfini, R. Popescu. Semantics-based composition-oriented discovery of web services. *ACM Transactions on Internet Technology*, vol. 8, no. 4, Article 19, 2008.

[44] *State of the art on modernization methodologies, methods and tools*, [Online], Available: http://www.remics.eu/publicdeliverables, July 20, 2013.

[45] G. Canfora, M. D. Penta, L. Cerulo. Achievements and challenges in software reverse engineering. *Communications of the ACM*, vol. 54, no. 4, pp. 142–151, 2011.

[46] G. Lewis, E. J. Morris, D. B. Smith, S. Simanta. Smart: Analyzing the Reuse Potential of Legacy Components in A Service-oriented Architecture Environment, Technical Report, SEI, volume CMU/SEI-2008-TN-008, 2008.

[47] A. Umar, Z. Adalberto. Reengineering for service oriented architectures: A strategic decision model for integration versus migration. *Journal of Systems and Software*, vol. 82, no. 3, pp. 448–462, 2009.

[48] Z. P. Zhang, R. M. Liu, H. J. Yang. Service identification and packaging in service oriented reengineering. In *Proceedings of the 17th International Conference on Software Engineering and Knowledge Engineering (SEKE 2005)*, Taipei, Taiwan, China, vol. 5, pp. 620–625, 2005.

[49] J. Oldevik, G. K. Olsen, U. Brönner, N. R. Bodsberg. Model-driven migration of scientific legacy systems to service-oriented architectures. In *Proceedings of the 1st International Workshop on Model-driven Software Migration*, pp. 4–7, 2011.

[50] S. Chung, J. B. C. An, S. Davalos. Service-oriented software reengineering: SoSR. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, IEEE, Waikoloa, HI, USA, pp. 172c, 2007.

[51] A. Winter, J. Ziemann. Model-based migration to service-oriented architectures. In *Proceedings of the 11th European Conference on Software Maintenance and Reengineering (CSMR 2007)*, Vrije Universiteit Amsterdam, pp.107–110, 2007.

[52] M. Razavian, P. Lago. Understanding SOA migration using a conceptual framework. *Journal of Systems Integration*, vol. 1, no. 3, pp. 33–34, 2010.

[53] S. Frey, W. Hasselbring. Model-based migration of legacy software systems to scalable and resource-efficient cloud-based applications: The cloudMIG approach. In *Proceedings of the 1st International Conference on Cloud Computing, GRIDs, and Virtualization*, pp. 155–158, 2010.

[54] PIM4Cloud, [Online], Available: http://www.remics.eu/publicdeliverables, July 10, 2013.

[55] Remics Migrate Toolkit, Preliminary Release, [Online], Available: http://www.remics.eu/publicdeliverables, July 11, 2013.

[56] Remics Migrate Toolkt, interim release, [Online], Available: http://www.remics.eu/publicdeliverables, July 12, 2013.

Jun-Feng Zhao received his B. Sc. degree in computer science from Inner Mongolia Normal University, China, and his M. Sc. degree in software engineering from National University of Defense Technology, China in 2006. He is currently a lecturer and a Ph. D. candidate at College of Computer Science, Inner Mongolia University.

His research interests include software reengineering, formal modeling and cloud computing.

E-mail: cszjf@imu.edu.cn

Jian-Tao Zhou received her Ph. D. degree from Tsinghua University in 2005. She is a professor and Ph. D. supervisor in College of Computer Science, Inner Mongolia University.

Her research interests include network computing and formal methods.

E-mail: cszjtao@imu.edu.cn (Corresponding author)