# Software Migration Frameworks for Software System Solutions: A Systematic Literature Review

Muhammad Shoaib[1], Adeed Ishaq[2] , Muhammad Awais Ahmad[2] , Sidra Talib[2], Ghulam Mustafa[2], Aqeel Ahmed[2]
[1]Department of Computer Science, National College of Business Administration & Economics DHA Campus Lahore, Pakistan,
[2]Department of Computer Science & IT,
The University of Lahore,
Lahore, Pakistan.

***Abstract*—This study examines and review the current software migration frameworks. With the quick technological enhancement, companies need to move their software's from one platform to another platform like cloud-based migration. There are different types of risks involved during migration. By performing migration activities correctly these risks might be reduced. Due to the absence of resources, such as workforce, time, budget in small organizations, the software migration is not performed in optimized way. Therefore, many functionalities are not implemented exactly after migration. In this paper, we have described different methods and frameworks which provide guideline for developers to enhance software migration process.**

***Keywords—Software migration; frameworks; system migration; cloud migration; migration risk***

## I. Introduction

According to different researchers migration can be done on small scale or large scale. Migration of the single system is the example of small scale where large scale migration involves more than one system. A number of migration types such as code migration, platform migration, operating system migration, database migration, user interface migration, programming languages migration, architecture migration and infrastructure migration. Mostly organization migrate their software products from one existing system to target system to get benefit of rapid variations of development tools and techniques.

The legacy systems are normally 10 to 30 years old, during this period they became critical and mature enough that their establishment required huge investment for several organizations. Software systems that have multi millions lines become very difficult to migrate because of their large scale, inconsistent documentation, aging implementation technologies and incomplete specifications. During a system's life, it may have to be modified to run in different environments.

Software migration is a complex process involving different stakeholders, performing migration activities at different point in time. It requires understanding of existing system, developing contingency plans to risk mitigation and ensuring that the application will continue to work in the purposed environment.

Due to the critical nature of software migration, it requires a systematic process prescribing (1) migration activities, (2) roles performing the activities, (3) migration phases (4) work products and (5) guidelines.

The main purpose of this study is to explore current existing frameworks that are useful for software migration, identify risks in migration, migration challenges and their solutions. In the remaining paper, Section 2 presents research methodology and Section 3 describes related work. The rest of paper presents a background material on software migration frameworks and discussed their challenges. Further we discuss risks in migration. Finally, we end the paper with comparison of different frameworks, discussion on our findings, conclusion and direction for future work.

## II. Research Methodology

In our research process, we followed the guideline proposed by Kitchenham and Charters [49]. We formed research questions, search strategies, defined inclusion criteria, exclusion criteria and Data Extraction and Assessment of Study Quality. In order to fulfil the goal of the study, we have collected data through Research papers and followed the systematic literature review process for the identification of QAs in software migration.

### *A. Research Questions*

We have designed the following research questions (RQs):

RQ1: Identifying the existing challenges in software migration frameworks?

RQ2: What are the frameworks used in Software Migration?

RQ3: Identifying the real world practices in Software Migration?

### *B. Search Strategy*

#### *1) Identifying Search Terms*

From research questions, we extract major terms and check their synonyms for obtaining related research papers. These terms verified in relevant papers. OR operator is used for concatenation of synonyms words. AND operator is used for concatenation of major terms.

#### *2) Search Strings*

("Software Migration" OR "System Migration" OR "Software Modernization" OR "Legacy Modernization" OR "Software Migration Maintenance") AND ("Framework" OR

"Process Models" OR "Activities" OR "Task" OR "Factors" OR "Challenges" AND (characteristics OR features OR barriers OR risks OR problems OR "issues").

*3) Trial Search*

We search related data from the following electronic data sources which are presented in Table 1.

TABLE I. TRAIL SEARCH RESULTS

| S.No | Digital library | Search | Conduction Date |
|---|---|---|---|
| 1 | IEEE Explore | 144 | 1 May 2017 |
| 2 | ACM Digital Library | 118 | 10 May 2017 |
| 3 | Google Scholar | 101 | 20 May 2017 |
| 4 | Science Direct | 12 | 30 May 2017 |
| 5 | Site Seer Digital Library | 11 | 10 June 2017 |
| 6 | Springer link | 8 | 10 July 2017 |

### C. Study Selection Criteria

With the help of search strategies used above, we took an enormous amount of products listed in databases. Inclusion criteria used to find the related peer reviewed material (conference proceedings and journal papers) by using different search terms.

*1) Included Criteria*

Studies that are described theoretical concepts in the context of software migration. Studies that are described software migration in the context of information technology or part of their migration issues. Studies that describe software migration and studies that directly answer one of the research questions.

*2) Excluded Criteria*

It includes the studies that are described software migration without the scope of software engineering such as manufacturing studies that are part of websites. Studies that are described software migration but not in software engineering at all.

*3) Data Extraction and Assessment of Study Quality*

As our inclusive criteria is defined, that data which is relevant to our research, fulfill our requirements and defining the scope of our research, will guide us for quality study. Relevant data is gained and now assessment started for quality study.

## III. RELATED WORK

Different types of software migration described by different researches like Migration of Legacy Software Systems into Web Service-based Architectures [50], SOA migration [51], Cloud Migration Research [54], a comparative evaluation of cloud migration optimization approaches [52], Exploring the factors influencing the cloud computing adoption [53], A Fresh Perspective on Total Cost of Ownership Models for Flash Storage in Datacenters [55], Understanding Performance of I/O Intensive Containerized Applications for NVMe SSDs [56], AutoReplica: Automatic Data Replica Manager in Distributed Caching and Data Processing Systems [57], FiM: Performance Prediction Model for Parallel Computation in Iterative Data Processing Applications [58], Accelerating Big Data Applications Using Lightweight Virtualization Framework on Enterprise Cloud [59], GREM: Dynamic SSD Resource Allocation in Virtualized Storage Systems with Heterogeneous IO Workloads [60], Improving Virtual Machine Migration via Deduplication [61], Improving Flash Resource Utilization at Minimal Management Cost in Virtualized Flash-based Storage Systems [62], eSplash: Efficient Speculation in Large Scale Heterogeneous Computing Systems [63], SEINA: A Stealthy and Effective Internal Attack in Hadoop Systems [64], AutoPath: Harnessing Parallel Execution Paths for Efficient Resource Allocation in Multi-Stage Big Data Frameworks [65] and EA2S2: An Efficient Application-Aware Storage System for Big Data Processing in Heterogeneous Clusters [66].

Service oriented introduced as solution for rehabilitation of legacy system. SOAs are associated to business, so it should be compulsory to clearly understand about business as what is business vision, their actors, business use cases and processes, goals and key performances and their customers. The current study provides SOA migration approaches and activities to produce that approaches like categorization. Also, derive a reference model, called SOA migration frame of reference [51].

In [50] the grouping of legacy software migration procedure performed which highlighted at least one advantage and disadvantage of each procedure in term of their benefits. Also it describe the way migration process should carried out to limit the bottlenecks, where wrapping the existing system source code directly to generate web services.

Ideal environment for provisioning applications, engineering and hosting is called cloud computing. Legacy applications have unique characteristics such as modernization and migration to cloud environments. The cloud migration is advancing but is not mature enough that is still in early stages of research. This review explore the needs of migration frameworks, how to enhance trust in cloud migration, less number of tools to automate migration task, describe the self-adaptive cloud-enabled systems and needs for their architecture [54].

In [52] the current cloud migration optimization approaches are identified and classified. It also performs comparison between them that show the gap in current approaches and highlight the future directions.

[53] This literature review provides clear understanding of cloud migration security framework. Also the need of secure cloud migration model is identified so it can be trusted by organizations and at the end a conceptual model for cloud migration is proposed.

### A. Risks in Software Migration

Software projects are classified in three categories: development, maintenance and migration. In migration different transformational phase perform by different researchers. Changing the existing system involved many types of risks during transforming. Table 2 presenting the different types of risks associated during migration process, which are extracted and highlighted by several researchers during different studies they conducted.

TABLE II. DIFFERENT TYPES OF RISK IN DIFFERENT STUDIES

| Sr. # | The Researcher(s) | Risks |
|---|---|---|
| 1 | (Boehm, 1991) [39] | Requirements understanding, less expertise, scope change, computer capacity, performance of products |
| 2 | (Barki, Rivard, & Talbot, 1993) [40] | Less expertise, technological advancement, size of application, Application complexity and environment of organization |
| 3 | (Nidumolu, 1995) [41] | Coordination in project |
| 4 | (Wallace, 1999) [42] | Environment of organization, Users and Development team |
| 5 | (Oz & Sosik, 2000) [43] | Leadership faults, inappropriate communication, absence of skills, less project management and deviation from actual scope line |
| 6 | (Schmidt, 2001) [44] | Project Management, Relationship Management, Scope, Planning, Development process, Requirements, Funding, Scheduling and Technology |
| 7 | (Tesch, 2007) [45] | Relationship Management, Ownership, Funds and Schedules, Scope and Requirements |

### *B. Different Concepts Related to Migration used by Different Research Eruations*

Different researchers purposed different techniques for migration for an instance virtualization. This approach refers to the ability for running an application within a virtual space that we can manage as an individual item. This approach is considered well for those applications that have relatively small data centre. Virtualization is a good way to repurpose aging servers [1]. To minimize the risk factors associated in this approach, incremental approach is used [7] where different components of generic information system are migrated. New approaches like network centric computing, user interface technology and so on in software engineering are establishing day by day. Due to the enhancement of these technologies, it's become very difficult to estimate the effects of these technologies. Different studies show that data management strategies can be applied when we perform migration. In recent age software systems were being used in high integrity and high security environments. Software systems used to provide support to our society's services such as electric utility, healthcare field, and communication over the internet, services in banking and finance area which are very critical. These systems will provide services continuously. Keeping services in running situations provide a sensible level of insurance that a framework will helps in gaining goal in required time and reduce chances of failure [8].

Software modernization is refer to the technique where old legacy system is converted or rewrite in new libraries, protocols, programming language or hardware platforms. The meaning of recollecting is to increasing the value of legacy systems through migration, which transform it into new structure. After the first release of software, software systems are constantly developed to fulfil the varying requirement of the users. For the sake of correcting changes, counting new functionality and implementing new one technologies software originators are doing variations in the source code. Unluckily, Lehman's laws of evolution described that the quality decreases of evolving program due to coding structure and turn into higher difficulty [2]. For that reason, legacy system becomes very complex to understand. If we cannot understand the system than its maintenance cannot be performed, the reason being that after many years of evolution due to little understanding about it. If we want to span legacy system lifecycle, we have to follow the object oriented platforms instead of procedural systems [3]. Migration of software requires deep knowledge and expertise in the application domains. A powerful knowledge representation technique called ontology [4] that deals with correct and more accurate specification of shared conceptualization of a domain. It provides facility in sharing of knowledge and software reuse in Artificial Intelligence. Domains of many applications make use of ontologies [5], [6]. In this situation where failures occur services will be still given using new platform. Service migration is used for the purpose of suspending the current state of a service on existing system and running this service to another system. It is also involve in the movement of core services program to different platform and freezing where the computation was ceased on the newer platform. For various critical safety systems, migration is known as the heart of the structural design of software [9]. Migration is considered an important strategy for system survivability service. In those conditions where we have seen that reproductions of a system is very difficult or may go outside the boundary of budgetary constraints service migration will be valuable for system migration. Fault tolerance is survivability techniques which can be used for damage avoidance and mask providing. In the compression of these services with service migration various problems found in above techniques which are very costly or may be impossible for large scope and high complexity systems [10].

The way of computing, storage and networking means are changed in Clouds [11]. These services are purchased and consumed by services providers across the cloud stack layers [12]. In everyday life the services such as social network, web content access and ecommerce plays important role in our lives. Mostly data centres host a large amount of these services that are managed by live VM migration in an effective way with the enhancement of server virtualization. The administrators of a network in live VM migration transfer the services to various servers over the network for better load balance. Due to VM migration, we can control load balance and save energy without carrying any services disturbance. Moreover the network administrator has many data centres by a single organization such as cloud bursting [13] via live VM migration across data centers.

### *C. EPSS-Electronic Performance Support System*

A French company developed a system for software migration named as EPSS (Electronic Performance Support System), where front-end was coded in JAVASCRIPT, AJAX and HTML and data management was coded in JAVA/J2EE. On other side server was coded using JAVA language where client side was coded in AJAX. The main purpose of this is to make EPSS fully compatible with Oracle E-business.

*Drawbacks:*

- No proper method for migration was defined.
- Poor visibility towards goal.
- The functional coverage of the system was not properly defined.
- It does not have the source analysis model.

*D. FASMM-Fast and Accessible Software Migration Method*

To avoid all EPSS drawbacks there proposed a new framework named as FASMM. This method involves some functional actors which are business analyst. The main purpose is to define the goal of the user that project is likely to achieve. This method mainly describes five intentions. [22]

- Getting model
- Migration of functionality
- Validation
- Describing rules of transformation
- Improvement of dictionary

*Getting Model*: Basic purpose is to define the border of the function which is to be migrated. For getting modelling two different techniques used.

- Functional Modelling
- Technical Modelling

*Migration of functionality:* After this, developers need to validate functional modelling and technique modelling by consensus. For this two strategies are identified.

*Validation:* Check the validity of use case diagrams. Perform functional validation.

*Discover rules of transformation:* The rules for transformation will be defined by experience, by code review / heuristic, by problem identification and by model analysis.

*Improvement of dictionary:* In this step, the developer has to ensure the improvement of the dictionary.

*E. SOMA-Service-Oriented Modeling and Architecture*

In [17] a Service Oriented System was designed and implemented by use of an incremental process. Basically SOMA defines, planning of software system, designing of software system, implementation of the software system and installation of the SOA system.

*F. Quality Driven Software Migration framework*

Non-functional requirements play an important role in software development process. This paper defines a systematic and quantitative approach through which quality of software system produce due to the direction of migration process. It will enhance the software properties like reusability and higher maintainability etc. In such quantity driven process of migration three main issues are defined.

- Quality goals which are compulsory in under development system.
- Operationalize these goals during migration process.
- Validation of these specific goals in under development system

The first problem is resolved by getting domain knowledge, interviewing the customer and documentation. Second issue is resolved by identifying a set of transformation. To get desire results they suggest a controllable methodology.

*Advantages:* This model enables the coalition of system qualities (specific) with source code also it allow selection as well as the application of transformation rules[3].

*G. A Framework for Process Migration in Software DSM Environments*

Migration scheme is evaluated by Trademarks, which is a commercial product that basically presents the state of art in the realism of software DSM system [24]. Migration Framework is a part of COROL project. This paper discusses the difference between single process migration and parallel process migration. Ivan Zoraja [25] compares three monitoring systems developed at LRR-TUM Munchen which address different programming paradigms.

According to the structure of shared address space, following breakdowns can obtained.

- Paged based system
- Fine grained system
- Object based system

*H. Software Engineering Challenges for Migration to the Service Cloud Paradigm [26]*

- SEA-Service Architecture Engineering [27]
- SOAD-Service Oriented Analysis and Design [28]
- Gartner Research developed SODA (Service Oriented Development of Applications). That is appropriate for software reuse.
- SOMA (Service Oriented Modelling and Architecture) [17] developed by IBM.

*Key Problems for Migration to the Service Cloud Paradigm: [26]*

- Context establishment
- Software architecture modernization
- Modernize data
- Managing the quality of non-functional requirements services in cloud
- Validation and verification should perform in cloud
- Introduce the quickness in migration process
- Advance business models

### I. Java based Applications Migration with Variety of Software Solutions

The system that provides many solutions for Business operations can be considered as information system. Business solution provided by these systems can be deployed on various platforms to fulfill the demands of the users. With the use use of these information systems, that actualy provide business operations has now become standard model. The proposed solutions mostly adopted in enterprise software solution for commercial support, open source application servers support and databases liked named AAMS (Authentication & Authorization Management System) support. Here we compare AAMS with different servers and databases [29].

*Implementation of AAMS:* AAMS is on the paper in java programming language it has following modules:

- Authentication
- Authorization
- Account Management

*Migrating the Enterprise:* Here they introduce a new concept called, E-stack, which is used to track the management architecture and execution of business of any company. We will follow E-stack and examine the four possible migration options. We will also familiarize a set of matrices which will be used to select the correct option [30].

*Problems and their solutions*: Constructed E-stack serve as the base to identify the key components in the migrating system. These components can either be integrated or can be individualy deliver services to application or can support as administrator of the application.

*During migration following problems can occur:*

- Full attention is provided to the component of application.
- The software that is being built does not have effectiveness and capability to fulfil business requests.
- We do not find compatible platform to migrate software from one infrastructure to other infrastructure.

### J. SNOW-Software Systems for Process Migration in High-Performance Heterogeneous Distributed Environments

SNOW[31] is basically Scalable Network of Workstation that supports client level process migration. It uses large scale distributed systems for process migration in a heterogeneous environment. When we transfer a process between two systems containing different software and hardware aspects it is called process migration of heterogeneous systems. Memory state, Communication state and Execution state are three main challenges faced in heterogeneous systems for migration process. SNOW facilitates in migration of all states. A compiler is used in execution state of migration which is done through analysis of source code. Memory pieces are referred as memory block. For this purpose they define memory space representation (MSR).

### K. ARTIST Methodology and Framework: A Methdology used for Migrating the Legacy Software System on Cloud

In EU Project, a model driven transformation and movement proposed. For legacy application migration ARTIST suggested a “one stop shop”. It is big issue in cloud environments for all participated entity to transform and adaptation of legacy system. Not only regarding technical point of view as well as it’s a critical issue for business level in the following the commercial procedures and representation of transformed application [32].

*Artist Migration Methodology*

*Pre-migration:* It is a step in which a knowledge about the practical and financial availability will be accomplished.

*Migration:* In this step migration process will be carry out with the help of reverse engineering (RE) and forward engineering (FE) for the purpose of implementation of legacy system in the cloud.

*Post-migration:* The parts of transformed application will be implemented on the destination environment and then it will be tested.

*Migration Artifacts Reuse & Evolution:* In this step all those events are performed regarding maintenance after transformation to the cloud.

*Development Phase:* The main conceptual level of many steps comprising the methodology.

*Disciplines:* Every step and task will be recognized as practical, procedure or commercial.

*Migration Phase:*

- Detection of model and its understanding.
- Requirements of objective environment Model Driven Forward Engineering (MDFE) can be used for transforming and implementing the migration into the destination cloud.

*Migration Artifacts Reuse & Evolution Phase:* The major determination of this step is to adoptive reprocess of ARTIST artifacts and provides more efficient evolution of software to a many cloud supplier if they desired.

### L. Search based Migration of Model Variants to Software Product Line Architectures [33]

*Feature Traceability (Set of UML class diagrams):* In this phase traceability links between each feature are extrected automatically.

*Reverse Engineering of Feature Model:* Set of different features which denote collection of features provided by variant.

*PLA construction:* The production of class diagrams with smarty notation is the key purpose of this phase.

### M. OPTIMA: An Ontology based Platform Specific software Migration Approach

Ontology known as a standout amongst the capable learning representational techniques which gives a formal

represenation and express detail about imported conceptualization of a domain. In which requirement had been formed for computerized reasoning to encourage learning imparting in reuse. A few requisition domains would settling on utilization of ontologies will include those learning measurement to their devices. A specific platform based on Ontology used for the software migration is called OPTIMA [34]. Maintainers requires more attempts to perform programming migration operations, system APIs will be cause of critical situation during migration activities. On restricted down the inspection difficulty in that study, these technique adopts in interface determinations concerning illustration the fundamental detail method to highlight three reasons:

*Firstly:* Interfaces are drawn cheaply from system models as well as from legacy systems.

*Secondly:* Interfaces might be used to isomorphic partner speak to both framework models that are architected throughout the ahead building phase, What's more legacy wrappers that would obtained throughout the reverse building phase and specifying the conceptual benefits represented by these legacy frameworks.

*Thirdly:* The establishment of mappings between ontology and source code, they might clearly converted under parameterised parts that holds their major aspect for running ahead legacy wrappers.

*N. ProfMig-A Framework for Flexible Migration of Program Profiles across Software Versions*

Offline programs cost more in term of frequent update. Here proposed a systematic solution for profile migration between different cross-version programs. Cross version program behavior profile migration is efficiently reduce valid profiles of old version into a new version which is introduced by them. The idea seems natural and has only been preliminary explored. Before design of the migration system it is the need to comprehend the factor affecting the system. The main source of complexity is bad behavior of program & analysis of scope changing code.

*Complexities from program behavior:*

- Behaviour unit
- Order and Nesting

*Complexities from profile formats:*

- Level of abstraction
- Meta data

*Complexities from impact analysis*: Change impact analysis [35] is technique in which information about code change impact analysis in the most recent adaptation of software.

*Solution of complexities:* Among the listed complexities some just need reasonable design consideration while other includes analysis of impact factors in the behavior. Here describe the solution of the problem by presenting a general framework named ProfMig. [36]

*O. Incorporating Quality Requirements in Software Migration Process*

As the software product released it constantly being evolved to fulfill the varying requirement form the user side. Software developer as well as software maintainer are executing changes in the source code that is being used by the user to remove faults, to add some new features and to make sure that software will work in new technologies. If they don't do these things there are so many chances of software uncontrolled. The value of a developing program cause the failure and program construction leads to higher complexity as highlighted by Lehman's laws of evolution. Due to these reasons legacy systems become very difficult to understan and difficult to maintene after many years of continuous development. A synthesized domain model for a diffeenrt procedural language like COBOL, Pascal, C and FORTRAN presented here. Mutual features between these languages such as functions types, procedures and sub procedures can be showed by XML also, we can put on standardized conversions over systems written in a diversity of procedural language with the use of this model. Furthermore, it suggest a signifecant framework that will observe and assess software qualities at every step of the reengineering process [37].

- The quality was not maintained previously.
- Scope was not defined properly.
- Previous method cannot maintain the lifecycle of legacy systems.
- Previous method doesn't include the non-functional requirements of system.

*P. Moving and Relocating-A Logical Framework of Service Migration for Software System Survivability[38]*

- Moving and Relocating
- System model and service migration
- The logic constraint solving and proof search

A detail comparison of different software migration frameworks expresses in Table 3. In this table framework name, migration phases, migration activities, roles performing the activities, work products, future guidelines and migration type explain in different columns.

TABLE III. DIFFERENT FRAMEWORK/ PROCESS/ TECHNIQUES FOR MIGRATION AND THEIR CHALLENGES

| Frame Work Name | Migration phases | Migration activities | Roles performing the activities | Work products | Future guidelines | Migration type |
|---|---|---|---|---|---|---|
| ARTIST Migration Methodology and Framework | 1) Pre Migration<br>• Technical feasibility analysis<br>• Business feasibility analysis<br>2) Migration<br>• Application discovery and understanding<br>• Target environment specification<br>• Modernization<br>3) Post Migration<br>• Scale<br>• Multitenant<br>• Monitor able<br>• Bill automatically<br>• Highest security standards | 1) Pre-Migration<br>• Technical & business evaluation<br>• Technical & business feasibility analysis<br>• Decide whether to migrate or not<br>• Methodology customization<br>2) Migration<br>• Verification of behavioural equivalence<br>• Verification of non-functional requirement<br>• Certification of the migrated product<br>• Migration artefacts reuse and evolution<br>3) Post Migration<br>• Capture changes<br>• Detect inconsistences<br>• Resolve inconsistences<br>• Implement Changes | Application owner and developer | ARTIST ARCHITECTURE<br>ARTIST Methodology<br>Process Tool | Enhance their applicability and impact in real business scenarios | Legacy application to modern cloud environment |
| FASMM (Fast and accessible software migration method) | 1) Get model<br>• Functional Modelling<br>• Technical Modelling<br>2) Migrate functionality<br>3) Validate<br>4) Discover transformation rule<br>5) Enrich dictionary | 1) Get model<br>• Reviewing existing model Reverse engineering<br>• Reviewing existing code<br>• By experience<br>• Identify reusable parts<br>• Validation<br>• Identify unsatisfactory part<br>2) Migrate functionality<br>• Semi-automatic strategy<br>• Manual strategy<br>3) Validate<br>• Functional test<br>• Technical test<br>4) Discover transformation rule<br>• Model analysis<br>• Problem identification<br>• Heuristic and code review<br>• By experience<br>5) Enrich dictionary<br>• Correction<br>• Invalidation<br>• Creation | Developers | FASMM approach for any company wants migration towards new technology | Can build a component base to store FASMM components for improve the migration methods knowledge | Migration of software from one source to target technology |

| | | | | | | |
|---|---|---|---|---|---|---|
| Software defined line virtual machine migration | 1) Pre-migration stage<br>2) Migration stage<br>3) Post migration stage | 1) Pre-migration stage<br>• Migration condition verify<br>• Migration iterations<br>• Establish network path for migration<br>• Coordinate to establish cross data centre network path<br>2) Migration stage<br>• Storage and memory migration<br>• Analysing activities flaws<br>• Calculating new flaws table<br>• Pre-update flaws table<br>• Coordinate the controller to finish pre-network state update<br>3) Post migration stage<br>• Shutdown the original VM<br>• Start the VM in the new location<br>• Update the VM location information | During development implemented by developers | Dynamically reallocation VMs providing various services to different servers for better load balance and energy saving etc. | Motivated by the paper the future work can be increased | Virtual machine (VM) migration |
| SNOW Software systems for process Migration in High Performance | 1) Execution state migration<br>2) Memory state migration<br>3) Communication state migration | 1) Execution state migration<br>• Developed a pre-compiler<br>• Determine poll-points<br>• Recognize of process migration<br>• insert macros in the source code<br>2) Memory state migration<br>• present a logical memory model<br>3) Communication state migration<br>• Protocols are selected | Codded by developer | Process migration methodology for stack based languages in a heterogeneous network environments | The better understanding of data structure and data size can reduce the cost of migration | Process migration of a software system |
| Model driven software migration into service oriented architectures [23] | 1) Business Modelling<br>2) Solution management<br>3) Service identification<br>4) Service specification<br>5) Service realization<br>6) Service implementation<br>7) Service deployment | 1) Business Modelling<br>• All possible information and state of company is analysed<br>2) Solution management<br>• Select a technique to solve project specification problems.<br>3) Service identification<br>• Identify a business process service<br>4) Service specification<br>• Description of service design<br>5) Service realization<br>• Take decision of which services will be implemented<br>• Provide detail how to implement them.<br>6) Service implementation<br>• Implementation of a service<br>7) Service deployment<br>• Acceptance test are performed | Developer | Described a model driven approach to migrate legacy systems. Extended IBM's SOMA method. | The approach will be tested and adapted in industrial scale project in future | Legacy system into Service Oriented Architecture (SOA) |

| | | | | | | |
|---|---|---|---|---|---|---|
| OPTIMA An ontology-based Platform specific software migration approach | 1) Instance based definition<br>2) Application specified design<br>3) API based classification<br>4) Behaviour cantered organization<br>5) Cardinality restricted relations<br>6) Understanding aimed naming<br>7) Aspect oriented restructuring<br>8) Multi layered structure | 1) Instance based definition<br>• Define rules that one instance has one definition.<br>2) Application specified design<br>• Definition of concept.<br>3) API based classification<br>• Concepts in the ontology are divided into three categories<br>I. Code level concept<br>II. Code behaviour level concept<br>III. Code attributes level concept<br>IV. Behaviour cantered organization<br>• Concepts organized according to their behaviour.<br>4) Cardinality restricted relations<br>• Introduced Cardinality<br>5) Understanding aimed naming<br>• Determine names of concepts, relations and instance<br>6) Aspect oriented restructuring<br>• Check whether Ontology design process support restructuring or not<br>7) Multi layered structure<br>• Making sure that Ontology design support with multi layered structure | Applications owner or Developer | An OPTIMA approach is proposed to provide understand ability, specifications, reusability, knowledge acquisition and reliability for software migrations | Adding knowledge dimension to software migration approach will be feasible way to facilitate software migration process by making it more efficiently and accurately | RTOS specific software migration (RT Linux to Thread) |
| ProfMig A frame work for flexible Migration of program profile across software versions | 1) Relation between profiles and change impact analysis<br>2) Analyse the effect of various factor of program<br>3) Propose a simple norm of identity<br>4) Development of ProfMig<br>5) Introduce a set of metrics | 1) Relation between profiles and change impact analysis<br>• Perform change impact analysis.<br>2) Analyse the effect of various factor of program<br>• Perform analysis of factors and its complexities<br>3) Propose a simple norm of identity represent of multiple profile migration steps into cohesive process<br>4) Development of ProfMig<br>• Developed ProfMig framework<br>5) Introduce a set of metrics<br>• Test which framework is best | Developer | The ProfMig framework is proposed for migration of profile of different versions of a software.<br>On the behalf of experiments, it proved that profile migration is feasible in practice | The result indicates some opportunities for future improvement.<br>It suggests the need for better impact analysis techniques to be developed in the future. | Cross-version program profile Migration |
| Reverse Engineering strategies for software migration | 1) Re documentation<br>2) Restructuring of source code<br>3) Transformation of source code<br>4) Abstraction of recovery<br>5) Reimplementation | 1) Re documentation<br>• Capture change<br>• Document change<br>2) Restructuring of source code<br>• Understanding of code<br>• Understanding its functionality<br>3) Transformation of source code<br>• Migration of source code from one language to another language | Developer | Reverse Reengineering strategies server business needs better and migration them to modern architectures | Given the tremendous software asserts in many corporation reengineering these asserts to serve business need better and migrating them to modern architectures will be easy | Legacy system to Modern Architecture |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | 4) Abstraction of recovery<br>• Abstract of recovery steps are performed<br>5) Reimplementation<br>• Reimplementation in performed. | | | | |
| Moving and Relocating: A Logical framework of service migration | 1) The logic language<br>2) Representing a service migration<br>3) Logic interference rules | 1) The logic language<br>It includes<br>I. Entities<br>II. Platform<br>III. Service<br>IV. Actions<br>V. Event<br>VI. Time points<br>VII. Time interval etc.<br>2) Representing a service migration<br>• Describe how to use a logic to represent a service migration process<br>3) Logic interference rules<br>• Present it using sequent calculus | Developer / maintainer | Logical framework for service migration | The requirements for dependable, robust and resilient software systems have increased with the society's growing dependency on the critical systems | Service migration of a software system |
| A Best Practice of java based application migration with variety of software solutions | Migration Principles of Application Server<br>1) System property file<br>2) Adopt JNDI (java naming and dictionary interfaces)<br>3) Encode parameter<br>4) Development descriptor<br>Migration Principles for Database Sever<br>1) Database reversed words<br>2) Common data types<br>3) Common SQL syntax<br>4) Common SQL style<br>5) System time | Migration Principles of Application Server<br>1) System property file<br>• Gain knowledge about systems files<br>2) Adopt JNDI (java naming and dictionary interfaces)<br>• Provide description about AAMS implementation and JNDI<br>• Provide detail about mechanism to Adopt JNDI<br>3) Encode parameter<br>• Encoded all the parameter<br>4) Development descriptor<br>• Development descriptor are described<br>Migration Principles for Database Sever<br>1) Database reversed words<br>• Select compounds words to name for database tables and fields.<br>2) Common data types<br>• Select common data type of database fields and tables.<br>3) Common SQL syntax<br>• Follow the SQL syntax<br>4) Common SQL style<br>• Use SQL common SQL style<br>5) System time<br>• Obtain the system time by using java implementation | programmers | The paper proposed some common principles of system migration and system design rules.<br>The paper proposed an instance of A&A (authentication and authorization) management system(AAMS) to migrate different kind of application servers and databases | In the future, it will be application servers and relational databases added to raise the migration flexibility and make the common principles of migration more robust | Java base application migration from one server to another and from one database to another database |

| | | | | | | |
|---|---|---|---|---|---|---|
| Incorporating quality requirements in software migration process [37] | 1) Access quality status<br>2) Identify critical quality bottlenecks<br>3) Establish quality objective<br>4) Construct quality models<br>5) Quality measurements | 1) Access quality status<br>• Analysis are performed using different perspectives<br>I. Characteristics<br>II. Strengths<br>III. Weakness<br>2) Identify critical quality bottlenecks<br>• Error prone area are identified<br>• Selection of key bottleneck<br>3) Establish quality objective<br>• Selection of quality objectives<br>• Selection of target application scenarios<br>4) Construct quality models<br>• Different quality models are built and represented.<br>5) Quality measurements<br>• Selection of a set of metrics to compute the quality measurements | Developers | A prototype software toolkit is developed This toolkit developed a system segmentation algorithm to break a large system into a set of smaller working area | A plan to focus on extracting behaviour model for the original systems and to trace these behaviours in the migrated systems | Legacy system to modern object-oriented platform |

## IV. Discussion

Different migration techniques and frameworks are described in this paper. The characterization of software migration process and comparison of systematically selected studies by highlighting existing study gaps is the main concern of this study.

Software migration is the data movement from one database system to any other types of database system. Using software migration tools we have done migration of working software from source to targeted PC. Due to fast changes and inventions in technologies, the need of migration is increasing in old systems. For example, Transforming set of programs or guidelines like PLC (programmable logic controller) programs from one platform to another. Shifting a live process from one system to another system like migrate a process from lower specification system to higher specification system is called migration process. The two systems may have different specification. To acquire sharing of resources, data access locality, load distribution, fault resilience and mobile computing can perform process migration. In large scale distributed surroundings like computational grids [14], we perform runtime process migration for resource utilization from lower to higher specification system. This thing leads to improve performance in individual application and also to produce high throughput for distributed systems. Different methods are defined for migration of software system or legacy systems [15]-[17].

An incremental approach for migration of complete systems is defined [7]. Another general migration technique towards migration projects are proposed by the reengineering factory [18]. SMART approach [19] and Butterfly approach [16] both support Data migration. Using [20], we can get more detailed overview about strategies of migration. Reference Migration Process provided a general process model for software migration [20]. IBM's SOMA method is one of the best known strategies according to martin [21]. SOMA process model for SOA development and evolution that used as a methodological framework for recognising and encompassing migration activities and their technological support. For identification, specification and implemntation services seven different phases are being used that move iteratively [17].

## V. Conclusion

The key finding of this study is to elaborate current research knowledge about software migration. Software migration in simple words we can say moving of application to their desire place in the datacenter. We can use different way to migrate software. We can migrate software using outsourced and external hosting as well. Normally, whenever someone purchases large application, he may like to host it for the purpose that the maintenance of the application will be easy. Regarding migration purpose, we keep in mind that what services the vendor may be able to provide better or at a lower cost, so it could prove to be more cost effective option in the future.

The main areas, we have explored in software migration are framework, process, activities, challenges and their solutions. A detailed comparison of different selected studies is performed in this paper to point out the existing research gap as well as to explore future directions of work.

- Core migration frameworks with distinct activities which describe planning and migration process in different scenarios are extrected.

- Different methods and technique that are used in software migration described in detail. Following steps taken by researchers for achieving required output.
- Planning: Analyze and plan for migration strategies.
- Execution: modification of code, retrieval and transformation of data.
- Evaluation: deployment, testing and validation.

For further field development we believe that software migration and software engineering researchers have to propose a common framework.

## VI. Future Work

Here, we used AAMS which is proposed system for migrating business operation cross platform because it reduces time and lower construction cost during migration process. A common framwork for softwrae migration can be helpul for understanding the data structure and data size effectively that can reduce the cost of migration, also it will enhance the applicability and impact in real business environment. There will be a time in near future when common principles will be made more robust and also for raise in flexibility of migration more databases and application servers will be there [29].

## References

[1] Olzak, T., et al., Microsoft virtualization: master Microsoft server, desktop, application, and presentation virtualization. 2010: Syngress.

[2] Lehman, M.M., Programs, life cycles, and laws of software evolution. Proceedings of the IEEE, 1980. 68(9): p. 1060-1076.

[3] Zou, Y. Incremental quality driven software migration to object oriented systems. in Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on. 2004. IEEE.

[4] Fensel, D., et al., OIL in a nutshell, in Knowledge Engineering and Knowledge Management Methods, Models, and Tools. 2000, Springer. p. 1-16.

[5] Andrade, H. and J.H. Saltz. Towards a Knowledge Base Management System (KBMS): An Ontology-Aware Database Management System (DBMS). in SBBD. 1999. Citeseer.

[6] Corcho, O., M. Fernández-López, and A. Gómez-Pérez, Methodologies, tools and languages for building ontologies. Where is their meeting point? Data & knowledge engineering, 2003. 46(1): p. 41-64.

[7] Brodie, M.L. and M. Stonebraker, Migrating legacy systems: gateways, interfaces & the incremental approach. 1995: Morgan Kaufmann Publishers Inc.

[8] Ellison, R.J., et al., Survivability: Protecting your critical systems. IEEE Internet Computing, 1999. 3(6): p. 55.

[9] Keromytis, A.D., et al. A holistic approach to service survivability. in Proceedings of the 2003 ACM workshop on Survivable and self-regenerative systems: in association with 10th ACM Conference on Computer and Communications Security. 2003. ACM.

[10] Strunk, E.A., J.C. Knight, and M.A. Aiello. Assured Reconfiguration of Fail-Stop Systems. in Dependable Systems and Networks, 2005. DSN 2005. Proceedings. International Conference on. 2005. IEEE.

[11] Buyya, R., C.S. Yeo, and S. Venugopal. Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. in High Performance Computing and Communications, 2008. HPCC'08. 10th IEEE International Conference on. 2008. Ieee.

[12] Mell, P. and T. Grance, The NIST definition of cloud computing. 2011.

[13] Wood, T., et al. CloudNet: dynamic pooling of cloud resources by live WAN migration of virtual machines. in ACM Sigplan Notices. 2011. ACM.

[14] Kesselman, C. and I. Foster, The grid: blueprint for a future computing infrastructure. chapter2 Morgan Kaufmann Publication, 1999.

[15] Brodie, M.L. and M. Stonebraker, DARWIN: On the incremental migration of legacy information systems. Distributed Object Computing Group, Technical Report TR-0222-10-92-165, GTE Labs Inc, 1993.

[16] Wu, B., et al. The butterfly methodology: A gateway-free approach for migrating legacy information systems. In Engineering of Complex

Computer Systems, 1997. Proceedings., Third IEEE International Conference on. 1997. IEEE.

[17] Arsanjani, A., et al., SOMA: A method for developing service-oriented solutions. IBM systems Journal, 2008. 47(3): p. 377-396.

[18] Borchers, J., Erfahrungen mit dem Einsatz einer Reengineering Factory in einem großen Umstellungsprojekt. HMD-Praxis der Wirtschaftsinformatik, 1997. 194: p. 77-94.

[19] Lewis, G.A. and D. Smith. SMART tool demonstration. in Software Maintenance and Reengineering, 2008. CSMR 2008. 12th European Conference on. 2008. IEEE.

[20] Sneed, H.M., E. Wolf, and H. Heilmann, Softwaremigration in der Praxis: Übertragung alter Softwaresysteme in eine moderne Umgebung. 2010: dpunkt. Verlag.

[21] Martin W (2009) SOA Check 2009: Status Quo und Trends imVergleich zum SOA Check 2008 und 2007.; Available from: http://www.soa-check.eu/download.php?cat=30_Archiv&file=Download_Summary_SOA_Check_2009.pdf.

[22] Forite, L. and C. Hug. FASMM: Fast and Accessible Software Migration Method. in Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on. 2014. IEEE.

[23] Fuhr, A., et al., Model-driven software migration into service-oriented architectures. Computer Science-Research and Development, 2013. 28(1): p. 65-84.

[24] Zoraja, I., A. Bode, and V. Sunderam. A framework for process migration in software DSM environments. in Parallel and Distributed Processing, 2000. Proceedings. 8th Euromicro Workshop on. 2000. IEEE.

[25] Zoraja, I., G. Rackl, and T. Ludwig. Towards monitoring in parallel and distributed environments. in International Conference on Software in Telecommunications and Computer Networks–SoftCOM'99. 1999.

[26] Mohagheghi, P. and T. Sæther. Software engineering challenges for migration to the service cloud paradigm: Ongoing work in the remics project. in Services (SERVICES), 2011 IEEE World Congress on. 2011. IEEE.

[27] Butler, J., The architecture component of the SAE reference framework for SOA. CBDi Journal, http://www. cbdiforum. com/secure/interact/2007-03/the_architecture_component. php, 2007.

[28] Zimmermann, O., P. Krogdahl, and C. Gee, Elements of service-oriented analysis and design. IBM developerworks, 2004.

[29] Wang, Y.-S. and C.-C. Yang. A best practice of Java-based applications migration with variety of software solutions. in Network Operations and Management Symposium (APNOMS), 2014 16th Asia-Pacific. 2014. IEEE.

[30] Down, B. Migrating the enterprise [software migration]. in Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on. 2004. IEEE.

[31] Chanchio, K. and X.-H. Sun. SNOW: software systems for process migration in high-performance, heterogeneous distributed environments. in Parallel Processing Workshops, 2002. Proceedings. International Conference on. 2002. IEEE.

[32] Menychtas, A., et al. ARTIST Methodology and Framework: A novel approach for the migration of legacy software on the Cloud. in Symbolic and Numeric Algorithms for Scientific ***Computing*** (SYNASC), 2013 15th International Symposium on. 2013. IEEE.

[33] Assuncao, G. and W. Klewerton. Search-based migration of model variants to software product line architectures. in Software Engineering (ICSE), 2015 IEEE/ACM 37th IEEE International Conference on. 2015. IEEE.

[34] Zhou, H., et al. OPTIMA: an ontology-based PlaTform-specIfic software migration approach. in Quality Software, 2007. QSIC'07. Seventh International Conference on. 2007. IEEE.

[35] Bohner, S.A. and R. Arnold. Software change impact analysis for design evolution. in Proceedings of the 8th International Conference on Software Maintenance and Re-engineering. 1991.

[36] Zhou, M., et al. Profmig: A framework for flexible migration of program profiles across software versions. in Code Generation and Optimization (CGO), 2013 IEEE/ACM International Symposium on. 2013. IEEE.

[37] Zou, Y. Incorporating quality requirements in software migration process. in Software Technology and Engineering Practice, 2003. Eleventh Annual International Workshop on. 2003. IEEE.

[38] Zuo, Y. Moving and relocating: A logical framework of service migration for software system survivability. in Software Security and Reliability (SERE), 2013 IEEE 7th International Conference on. 2013. IEEE.

[39] Boehm, B. W. (1991). Software risk management: principles and practices.Software, IEEE, 8(1), 32-41.

[40] Barki, H., Rivard, S., & Talbot, J. (1993). Toward an Assessment of Software Development Risk. Journal of Management Information Systems , 10 (2), 203-225.

[41] Nidumolu, S. (1995). The Effect of Coordination and Uncertainty on Software Project Performance: Residual Performance Risk as an Intervening Variable. Information Systems Research , 191-219.

[42] Wallace, L. (1999). The development of an instrument to measure software project risk. Doctoral Dissertation, Georgia State University.

[43] Oz, E., & Sosik, J. (2000). Why information systems projects are abandoned: a leadership and communication theory and exploratory study. Journal of ComputerInformation Systems , 41 (1).

[44] Schmidt, R., Lyytinen, K., Keil, M., & Cule, P. (2001). Identifying software project risks: An international Delphi study. Journal of Management Information Systems , Vol.17, No.4, pp.5-36

[45] Tesch, D., Kloppenborg, T., & Erolick, M. (2007). IT Project Risk Factors: The Project Management Professionals Perspective. Journal Of Computer Information Systems .

[46] Thomas, S., & Bhasi, M. (2008). A Study on Software Development Project Risk, Risk Management, Project Outcomes and their Inter-Relationship. Retrieved March 25, 2013, from Dyuthi: http://dyuthi.cusat.ac.in/

[47] Charalambos, L., Iacovou, & Nakatsu, R. (2008). A Risk Profile Of Offshore-Outsourced Development Projects. Communications of the ACM , 51 (6).

[48] Sundararajan, S., Bhasi, M., & Pramod, K. (2013). An Empirical Study of industry practices in Software Development Risk Management. International Journal of Scientific and Research Publications , 3 (6).

[49] Kitchenham, B., & Charters, S. (2007).Guidelines for performing systematic literature reviews in software engineering. UK: Keele

[50] Rochimah, S., & Sheku, A. (2016). Migration of Existing or Legacy Software Systems into Web Service-based Architectures (Reengineering Process): A Systematic Literature Review. International Journal of Computer Applications, 133(3), 43-54.

[51] Razavian, M., & Lago, P. (2015). A systematic literature review on SOA migration. Journal of Software: Evolution and Process, 27(5), 337-372.

[52] Abdelmaboud, A., Jawawi, D. N., Ghani, I., & Elsafi, A. (2015). A COMPARATIVE EVALUATION OF CLOUD MIGRATION OPTIMIZATION APPROACHES: A SYSTEMATIC LITERATURE REVIEW. Journal of Theoretical and Applied Information Technology, 79(3), 395.

[53] Rai, R., Sahoo, G., & Mehfuz, S. (2015). Exploring the factors influencing the cloud computing adoption: a systematic study on cloud migration. SpringerPlus, 4(1), 1.

[54] Jamshidi, P., Ahmad, A., & Pahl, C. (2013). Cloud migration research: a systematic review. IEEE Transactions on Cloud Computing, 1(2), 142-157.

[55] Yang, Z., Awasthi, M., Ghosh, M., & Mi, N. (2016, December). A Fresh Perspective on Total Cost of Ownership Models for Flash Storage in Datacenters. In Cloud Computing Technology and Science (CloudCom), 2016 IEEE International Conference on (pp. 245-252). IEEE..

[56] Bhimani, J., Yang, J., Yang, Z., Mi, N., Xu, Q., Awasthi, M., ... & Balakrishnan, V. (2016, December). Understanding performance of i/o intensive containerized applications for nvme ssds. In Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International (pp. 1-8). IEEE.

[57] Yang, Z., Wang, J., Evans, D., & Mi, N. (2016, December). AutoReplica: Automatic data replica manager in distributed caching and data processing systems. In Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International (pp. 1-6). IEEE.

[58] Bhimani, J., Mi, N., Leeser, M., & Yang, Z. (2017, June). FiM: Performance Prediction for Parallel Computation in Iterative Data Processing Applications. In Cloud Computing (CLOUD), 2017 IEEE 10th International Conference on (pp. 359-366). IEEE.

[59] Bhimani, J., Yang, Z., Leeser, M., & Mi, N. (2017, September). Accelerating big data applications using lightweight virtualization framework on enterprise cloud. In High Performance Extreme Computing Conference (HPEC), 2017 IEEE (pp. 1-7). IEEE.

[60] Yang, Z., Tai, J., Bhimani, J., Wang, J., Mi, N., & Sheng, B. (2016, December). GReM: Dynamic SSD resource allocation in virtualized storage systems with heterogeneous IO workloads. In Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International (pp. 1-8). IEEE.

[61] Roemer, J., Groman, M., Yang, Z., Wang, Y., Tan, C. C., & Mi, N. (2014, October). Improving virtual machine migration via deduplication. In Mobile Ad Hoc and Sensor Systems (MASS), 2014 IEEE 11th International Conference on (pp. 702-707). IEEE.

[62] Tai, J., Liu, D., Yang, Z., Zhu, X., Lo, J., & Mi, N. (2017). Improving flash resource utilization at minimal management cost in virtualized flash-based storage systems. IEEE Transactions on Cloud Computing, 5(3), 537-549.

[63] Wang, J., Wang, T., Yang, Z., Mi, N., & Sheng, B. (2016, December). eSplash: Efficient speculation in large scale heterogeneous computing systems. In Performance Computing and Communications Conference (IPCCC), 2016 IEEE 35th International (pp. 1-8). IEEE.

[64] Wang, J., Wang, T., Yang, Z., Mao, Y., Mi, N., & Sheng, B. (2017, January). SEINA: A stealthy and effective internal attack in Hadoop systems. In Computing, Networking and Communications (ICNC), 2017 International Conference on (pp. 525-530). IEEE.

[65] Gao, H., Yang, Z., Bhimani, J., Wang, T., Wang, J., Sheng, B., & Mi, N. (2017, July). AutoPath: Harnessing parallel execution paths for efficient resource allocation in multi-stage big data frameworks. In Computer Communication and Networks (ICCCN), 2017 26th International Conference on (pp. 1-9). IEEE.

[66] Wang, T., Wang, J., Nguyen, S. N., Yang, Z., Mi, N., & Sheng, B. (2017, July). EA2S2: An Efficient Application-Aware Storage System for Big Data Processing in Heterogeneous Clusters. In Computer Communication and Networks (ICCCN), 2017 26th International Conference on (pp. 1-9). IEEE.