Guidelines for database transitioning on production environments

Fabio Leal

September 29, 2015

1 Abstract

Component-based Software Engineering (CBSE) and Service-Oriented Architecture (SOA) became popular ways to develop software over the last years. During the life-cycle of a software, several components and services can be developed, evolved and replaced. In production environments, the replacement of core components, such as databases, is often a risky and delicate operation, where several factors and stakeholders take place.

Service Level Agreements (SLA), according to ITILv3's official glossary, is "an agreement between an IT service provider and a customer. The agreement consists on a set of measurable constraints that a service provider must guarantee to its customers.". In practical terms, it is a document that a service provider delivers to its consumers with minimum quality of service (QoS) metrics.

This work assesses and improves the use of SLAs to guide the transitioning process of databases on production environments. In particular, in this work we propose SLA-Based Guidelines/Process to support migrations from a relational database management system (RDBMS) to NoSQL one. Our study is validated by case studies.

2 Background

The goal of this chapter is to present the technical concepts for a better understanding of our job.

2.1 Cloud Computing & The technological shift

Technology evolved with big steps over the last decades. Internet is now a pervasive concept and individuals can be connected virtually everywhere on Earth. [4]

Web applications and IT-based processes followed this evolution and today almost every company rely on software in a step of its production chain. Information Technology can be a competitive advantage of a company, but it **might**

not be part of its core business[14]. In fact, this is a common scenario on most of the successful companies of the current century.

To avoid losing track of its core business, a number of companies now prefer to outsource (part of) their IT department to other companies [15]. In other words, today it is possible to outsource IT infrastructure, product development and even the entire IT department to other companies.

In the late 60's, former Stanford University professor John McCarthy [11] introduced the concept of time-sharing of computing services. In fact, Mr. McCarthy believed that computer resources would be provided as commodities, like water and electricity.

Several years later, this concept brought to life the notion of *Cloud Computing*, together with new concepts, such as Infrastructure as a Service *(IAAS)*, Platform as a Service *(PAAS)*, Software as a Service *(SAAS)* and Everything as a Service *(XaaS)* [6].

According to [6], *Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services.*

2.2 XaaS - Everything as a Service

Service-Oriented Architecture (SOA) defines several concepts of "as-a-service" models. To enumerate a few, it is possible to find mentions to Platform as a Service (PaaS), Infrastructure as a Service (IaaS), Software as a Service (SaaS), Database as a Service (DBaaS), Desktop as a Service (DaaS), Monitoring as a Service (MaaS) and Communication as a Service (CaaS) on the literature.

To summarize all these concepts, a new term arose: Everything as a service (XaaS)[8].

On the context of Cloud Computing, however, three of them are the most relevant, and we define more precisely below:

- Infrastructure as a service (IaaS): It is the most simple kind of "as-a-service" product and is located on the base of the IaaS-PaaS-SaaS Stack - Figure 1. IaaS mostly refers to (Virtual) Machines, Storage Devices, Network Infrastructure and other infrastructural services that are available on Cloud Computing vendors.

- Platform as a Service (PaaS): PaaS refers to the development environment that are available from cloud vendors. It is composed by a development stack, and generally offers databases, web servers and execution runtime. Examples of PaaSs are Amazon BeanStalk, Microsoft Azure and Google AppEngine.

- Software as a Service (SaaS): Software as a Service refers to the complex software that runs on the cloud: Webmail, CRM, Gaming Platforms, Learning Management Systems, etc. SaaSs, just as IaaSs and PaaSs generally charge its' users a monthly/yearly fee. The fee is generally conceived in a pay-as-you-go model, so users get charged in a scalable way.

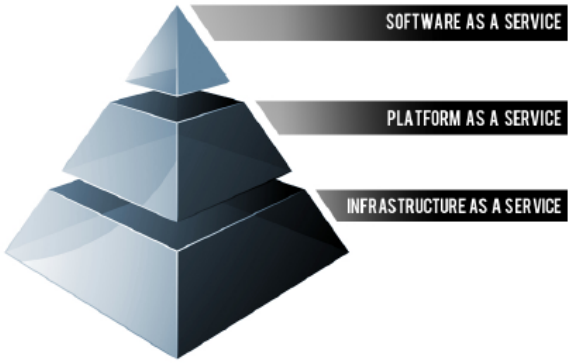

Figure 1: IaaS-PaaS-SaaS Stack [9].

2.3 Transitioning Processes

The adoption of cloud solutions is growing fast among organizations [7]. Centralized (mostly mainframe) technology is being replaced by distributed and more flexible forms of data storage and processing. This change of paradigm is motivated by the necessity to improve the use of resources, as well as by the increasing velocity in which data is produced.

In this scenario, transitions must take into account the quality of the service delivered by the new solutions.

On the early 90's it was commonplace for every Information Technology (IT) company to have its own Data Center with huge servers and mainframes. IT costs were high, and high-performance computing was available only for big companies, as data centers required a large physical infrastructure and have high costs for maintenance [4].

The regular way of building a web application was to use a client-server approach, where the server was a powerful (and expensive) machine. At the same time, new players, such as Google or Yahoo, were rising with bigger missions: *"to organize the world's information and make it universally accessible and useful"* [21]. The popularization of the internet use incentivized new ways of commerce exchange, yielding an explosion in the amount of data produced and exchanged. It was *just* impossible to store the petabytes of daily-generated data in a single server.

From this point on, the community realized the economical convenience of building and maintaining several low-performance servers, instead of a single high-performance one, even if this this requires a change of culture in the administration of the new datacentres. The new approach is also incompatible with the traditional way of building applications, that usually were designed to work on a single server and database.

Several research initiatives were conducted in this area and a common solution was rising: to distribute data storage and processing. Google, Yahoo and

other big IT players helped to build open source tools to make this approach possible, like Hadoop [19].

2.4 Data Integration & Polyglot Persistence

Over the last years, the number of Data Base (DB) Engines grew like never before [16]. Along with the NoSQL (Not only SQL) movement and expansion of Social Networks, new concepts for Database Models appeared, like Document Store, Search Engines, Key-Value store, Wide Column Store, Multi-Model and Graph DBMS. In [16] a ranking of the most popular DB engines is presented.

Today, instead of having a single Relational Database Management System (DBMS) for the whole application, it is efficient and cost-effective to have several Data Base Engines, one for each type of data that the application handles. This concept is called *Polyglot Persistence* [17].

As [20] illustrates, polyglot persistence is very useful in the context of e-commerce applications that deal with a catalog, user access logs, financial information, shopping carts and purchase transactions, for example. The notion of polyglot persistence is built upon the observation that the *nature* of each data type is significantly different (i.e: user logs imply high volume of writes on multiple nodes, shopping carts need high availability and user sessions require rapid access for reads and writes).

As computing services started to decentralize, developers started to build applications that depended of several data-sources. By this time the use of Web Services and Service Oriented Architecture (SOA) became more popular [4].

2.5 Systematic Mappings

According to [13], "*A software engineering systematic map is a defined method to build a classification scheme and structure a software engineering field of interest.*" Systematic Mapping studies provide a global view of a given research field and identify the quantity, results, and the kinds of researches in this field.

A Systematic map is composed by a number of steps (Figure 2).

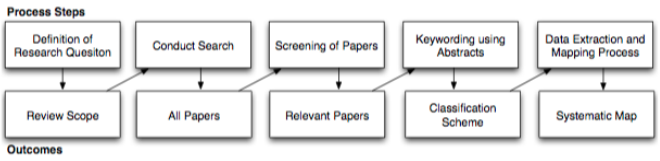

Figure 2: Systematic Mapping Steps [13].

On the first step, "Definition of Research question", the questions that must be answered on the survey are defined. On the "Review Scope" step, researchers target the papers/journal sources that will be taken into consideration on the

systematic map. After that, the "Search" step is done using a set of predefined search engines and a body of papers ("All papers") is retrieved.

After an initial "Screening of the papers", the "Relevant papers" are chosen according to inclusion and exclusion criteria defined by the research team. At this point, the papers that will participate of the study are selected. The selection is based on the title, abstracts and keywords of each paper ("Keywording using Abstracts").

After that, a "Classification Scheme" is built, defining different points-of-view (or facets) from which the body of papers will be classified. After matching each paper with the classification schema ("Data Extraction and Mapping Process"), the systematic mapping is performed. In this phase the relationships between the collected data (in the light of the classification scheme) are used to answer the research questions.

Service Level Agreements (SLAs)

According to *ITILv3's* official glossary [5], a Service Level Agreement (SLA) is "*an agreement between an IT service provider and a customer. A service level agreement describes the IT service, documents service level targets, and specifies the responsibilities of the IT service provider and the customer.*"

The agreement consists on a set of measurable constraints that a service provider must guarantee to its customers. In practical terms, it is a document that a service provider delivers to its consumers with minimum quality of service (QoS) metrics. If the service is delivered with a lower QoS than is promised on the SLA, consumers may be refunded or earn benefits that were accorded beforehand.

3 Bibliographic Review

To provide us a better understanding over the use of SLAs in component / service migrations, we have performed a Systematic Mapping study to assess the use of SLAs in database transition scenarios, specifically on migrations from relational databases with NoSQL ones.

This study is available on Appendix 4.

3.1 Identified problems

As a result, we have analyzed over 70 publications closely related to the use of SLAs in migration scenarios. The study revealed a number of interesting outcomes, and we emphasize two of them below:

- No publication was found addressing the problem of measuring the overall improvements after a database transition. Several benchmarking frameworks, such as TPC-H, TPC-DS and YCSB were identified[12] during our survey, though. These benchmarking frameworks could be a good starting point to develop new tools and specialized frameworks to solve this

problem and might be used in our study to validate that a migration was successful.

- [18], [23], [10] and [22] propose SLA-centric/User-Centric solutions to monitor the performance of web applications. All these solutions are technology-agnostic and could be used to monitor the performance improvements promised by a database transitioning process. Industry experts also pointed out that there are some services, such as New Relic [3], Appsee [1] and Datadog [2] that provide SLA-monitoring tools for web apps.

  The systematic mapping revealed no open source solution to monitor Application SLAs in a user-centered view (application level).

3.2 Proposed solution

Service-Oriented Architecture (SOA) suggests that a Software Component can be seen as a set of sub-services. In the context of a Web Application, for example, it is possible to entirely replace a component just by changing the API address that it uses. This scenario of component replacement on cloud-based applications can be seen on (Figure 3).

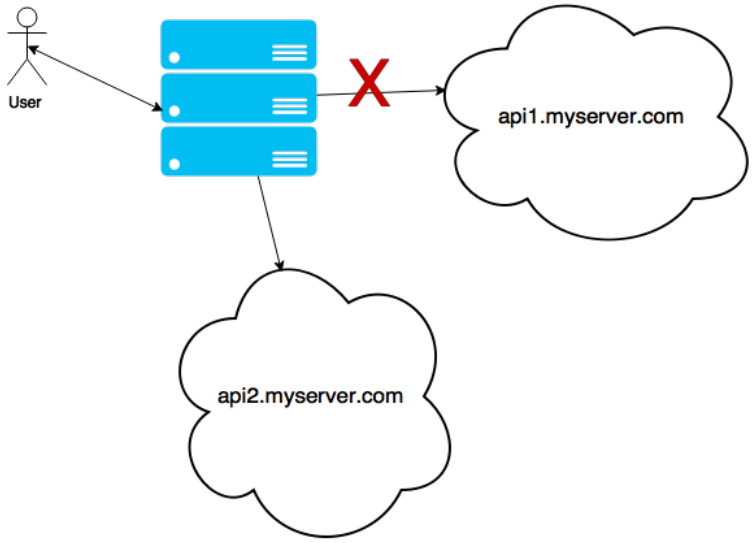

Figure 3: Component replacement on cloud-based software.

As Section 3 presents, contributions can be made in the context of proposing SLA-Based and user-centered solutions to monitor component replacement scenarios.

In this work we will assess the use of a SLA-Guided process to support the migration/replacement of relational databases with NoSQL ones. This process will be developed and assessed with case studies.

The boundaries of this work could then be broadened to support the conception of a SLA-Guided process to support the migration/replacement of sotware components based on the cloud in future works.

3.3 Roadmap

To provide a better understanding of our work, we splitted the research in six main phases:

Phase	Title	Description
1	Identification of Case Studies & SLAs	On this step we aim to identify examples where a Database transition is needed or recommended in order to satisfy a SLA. We will try to work on production-ready and open-source softwares. If the complexity of these projects is too large for our scope, we will design and develop our own scenarios.
2	Plan	After the scenarios have been identified, we will propose architectural changes that could satisfy the SLA. These changes will be proposed by literature reviews and survey of industry experts.
3	Do	On this step we implement the architecture proposed on the previous step.
4	Check	On the check step we will verify if the proposed architecture and implementation satisfies the SLAs identified on the first step.
5	Act	Tweaks can be needed on the proposed architecture and implementation if the SLA is still not satisfied by the changes made on the previous step. On the act phase we investigate what else can be done to satisfy the SLA and refine the process defined on step 2.
6	Final Results	On the final step we aim to publish the results of our work on relevant database-related conferences and workshops.

Each of these phases is composed by a number of steps, described below:

1. Phase 1 - Identification of Case Studies & SLAs
   (a) Step 1.1 - Scenario identification / Implementation: On this step we will search for open source projects and real-world scenarios where a Relational Database bottleneck has been identified. If the scope of these scenarios become too large, we will implement our own scenarios;

(b) Step 1.2 - Identification of broken SLAs: We need to identify that the a set a constraints (i.e: execution time of a query) is not being met by the current architecture;

(c) Step 1.3 - Implementation of "runnable SLAs" : On this step we will implement executable versions of the SLA identified on the previous step. These "runnable SLAs" will be used to verify that a set of constraints is not being met by the current architecture.

(d) Step 1.4 - Execution reports: After an executable SLA has been identified and implemented, execution reports will be consolidated to prove that the constraints of the SLA are being broken by the current architecture of the scenario.

2. Phase 2 - Plan

   (a) Step 2.1 - Literature Review for each scenario: We will evaluate and search for solutions on how each scenario can make use of a NoSQL Database to meet the desired SLA;

   (b) Step 2.2 - Survey of industry experts: We will survey industry experts on how they would propose a NoSQL architecture to solve the problem described on each scenario.

3. Phase 3 - Do

   (a) Step 3.1 - Planning of changes: We will gather the results from the previous phase and design the changes that will be performed on each scenario;

   (b) Step 3.2 - Implementation: We will implement the changes identified on the previous step.

4. Phase 4 - Check

   (a) Step 4.1 - New Execution Reports: The same SLAs identified on the first step will be run on the modified scenarios, and execution reports will be consolidated.

   (b) Step 4.2 - Comparison of Results: The reports extracted on steps 4.1 and 1.4 will be compared to check if the changes made on Phase 3 satisfied the proposed SLA.

5. Phase 5 - Act

   (a) Step 5.1 - Tweaks on the proposed architecture: If the SLA isn't being met yet, new changes might be needed, and on this step we join together the phases 2, 3 and 4 to iterate over the needed changes.

6. Phase 6 - Final Results

   (a) Step 6.1 - Publish the results: We will submit the results of this study to academical conferences to have feedback from the community.

(b) Step 6.2 - Write the final results: All the documents produced by our study and a final dissertation will be sent to the Universidade Federal do Rio Grande do Norte (UFRN).

3.4 Schedule

A detailed view of the execution flow of our steps can be seen on Figure 4.

A	B	C	D	E	F	G	H	I	J
Phase	Step Description	25/07/2015	15/08/2015	5/9/2015	26/09/2015	17/10/2015	7/11/2015	28/11/2015	19/12/2015
	Scenario identification / Implementation	x							
	Identification of broken SLAs		x						
	Implementation of "runnable SLAs"		x						
Phase 1	Execution reports		x						
	Literature Review for each scenario			x					
Phase 2	Survey of industry experts			x					
	Planning of changes			x					
Phase 3	Implementation			x	x	x			
	New Execution Reports					x			
Phase 4	Comparison of Results					x			
Phase 5	Tweaks on the proposed architecture					x	x		
	Publish the results						x	x	x
Phase 6	Write the final results						x	x	x

Figure 4: Schedule.

References

[1] Appsee. https://www.appsee.com/. Accessed: 2014-05-01.

[2] Datadog. https://www.datadoghq.com/. Accessed: 2014-05-01.

[3] New relic. http://newrelic.com. Accessed: 2014-05-01.

[4] Michael Armbrust, O Fox, Rean Griffith, Anthony D. Joseph, Y Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. M.: Above the clouds: A berkeley view of cloud computing. Technical report, 2009.

[5] Axelos. Best management practice portfolio: common glossary of terms and definitions, October 2012.

[6] A. et al. CONNOLY, D. FOX. A view of cloud computing, April 2010. [Online; posted 01-April-2010].

[7] G. Copil, D. Moldovan, Hong-Linh Truong, and S. Dustdar. Sybl: An extensible language for controlling elasticity in cloud applications. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pages 112–119, May 2013.

[8] Yucong Duan, Guohua Fu, Nianjun Zhou, Xiaobing Sun, N.C. Narendra, and Bo Hu. Everything as a service (xaas) on the cloud: Origins, current and future trends. In *Cloud Computing (CLOUD), 2015 IEEE 8th International Conference on*, pages 621–628, June 2015.

[9] Ben Kepes. Understanding the cloud computing stack. *Rackspace White Paper*, 2011.

[10] M. Klems, D. Bermbach, and R. Weinert. A runtime quality measurement framework for cloud database service systems. In *Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the*, pages 38–46, Sept 2012.

[11] John McCarthy. Comments on time sharing. *Commun. ACM*, 10(9):531, 1967.

[12] R. Moussa. Benchmarking data warehouse systems in the cloud. In *Computer Systems and Applications (AICCSA), 2013 ACS International Conference on*, pages 1–8, May 2013.

[13] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swinton, UK, UK, 2008. British Computer Society.

[14] Thomas C Powell and Anne Dent-Micallef. Information technology as competitive advantage: The role of human, business, and technology resources. *Strategic management journal*, 18(5):375–405, 1997.

[15] J Quinn, T Doorley, and P Paquette. Technology in services: rethinking strategic focus. *Sloan Management Review, Win-ter1990QuinnWinterSloan Management Review1990*, 2013.

[16] DB Ranking. Db engines ranking, May 2014.

[17] Pramod J Sadalage and Martin Fowler. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. Pearson Education, 2012.

[18] S. Sakr and A. Liu. Sla-based and consumer-centric dynamic provisioning for cloud databases. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 360–367, June 2012.

[19] K. Shvachko, Hairong Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10, May 2010.

[20] Genoveva Vargas Solar. Addressing data management on the cloud: tackling the big data challenges, May 2014.

[21] Alfred Spector, Peter Norvig, and Slav Petrov. Google's hybrid approach to research. *Commun. ACM*, 55(7):34–37, July 2012.

[22] Pengcheng Xiong, Yun Chi, Shenghuo Zhu, Junichi Tatemura, Calton Pu, and Hakan HacigümüŞ. Activesla: A profit-oriented admission control framework for database-as-a-service providers. In *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, SOCC '11, pages 15:1–15:14, New York, NY, USA, 2011. ACM.

[23] L. Zhao, S. Sakr, and A. Liu. A framework for consumer-centric sla management of cloud-hosted databases. *Services Computing, IEEE Transactions on*, PP(99):1–1, 2013.

4 Appendix

Using SLA to guide database transition to NoSQL on the cloud: a systematic mapping study

Fabio Leal ⟨sousaleal.fabio@gmail.com⟩
Martin A. Musicante ⟨mam@dimap.ufrn.br⟩

June 17, 2015

Abstract

Cloud computing became a reality, and many companies are now moving their data-centers to the cloud. A concept that is often linked with cloud computing is Infrastructure as a Service (IaaS): the computational infrastructure of a company can now be seen as a monthly cost instead of a number of different factors. Recently, a large number of organizations started to replace their relational databases with hybrid solutions (NoSQL DBs, Search Engines, ORDBs). These changes are motivated by (*i*) performance improvements on the overall performance of the applications and (*ii*) inability to a RDBMS to provide the same performance of a hybrid solution given a fixed monthly infrastructure cost. However, not always the companies can exactly measure beforehand the future impact on the performance on their services by making this sort of technological changes (replace RDBMS by another solution). The goal of this systematic mapping study is to investigate the use of Service-Level-Agreements (SLAs) on database-transitioning scenarios and to verify how SLAs can be used in this processes.

1 Introduction

The adoption of cloud solutions is growing fast among organizations [21]. Centralized (mostly mainframe) technology is being replaced by distributed and more flexible forms of data storage and processing. This change of paradigm is motivated by the necessity to improve the use of resources, as well as by the increasing velocity in which data is produced.

In this scenario, transitions must take into account the quality of the service delivered by the new solutions. The notion of Service-Level-Agreement (SLA) [44] may be used as a parameter in this context. SLAs are widely used to to provide service thresholds between clients and providers and are present in almost every service contract over the internet.

SLAs or OLAs - Operational Level agreements, are particularly useful to guide the process of choosing the most convenient service from a pool of service providers.

In this paper, we survey the use of SLA on database-transitioning scenarios, trying to investigate how they might be used to help the execution of this process. Our study is performed using the systematic mapping [41] technique: A set of papers is retrieved from the most popular bibliography repositories; this set is then filtered according to predefined parameters and finally, the analysis of the remaining papers is guided by a small number of research questions.

This paper is organized as follows: Section 2 presents some of the concepts that are related to the transition from a traditional setting to a cloud-aware one. Section 3 presents our research questions and each step of our survey. The outcomes of our Systematic Mapping study can be seen on Section 4.

2 The Technological Shift

On the early 90's it was commonplace for every Information Technology (IT) company to have its own Data Center with huge servers and mainframes. IT costs were high, and high-performance computing was available only for big companies, as data centers required a large physical infrastructure and have high costs for maintenance [13].

The regular way of building a web application was to use a client-server approach, where the server was a powerful (and expensive) machine. At the same time, new players, such as Google or Yahoo, were rising with bigger missions: *"to organize the world's information and make it universally accessible and useful"* [53]. The popularization of the internet use incentivized new ways of commerce exchange, yielding an explosion in the amount of data produced and exchanged. It was *just* impossible to store the petabytes of daily-generated data in a single server.

From this point on, the community realized the economical convenience of building and maintaining several low-performance servers, instead of a single high-performance one, even if this this requires a change of culture in the administration of the new datacentres. The new approach is also incompatible with the traditional way of building applications, that usually were designed to work on a single server and database.

Several research initiatives were conducted in this area and a common solution was rising: to distribute data storage and processing. Google, Yahoo and other big IT players helped to build open source tools to make this approach possible, like Hadoop [50].

This revolution brought to life the notion of *Cloud Computing*, together with new concepts, such as Infrastructure as a Service *(IAAS)*, Platform as a Service *(PAAS)* and Software as a Service *(SAAS)* [18]. According to [18], *Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services.*

Data Integration & Polyglot Persistence On the last years, the number of Data Base (DB) Engines grew like never before [3]. Along with the NoSQL (Not only SQL) movement and expansion of Social Networks, new concepts for Database Models appeared, like Document Store, Search Engines, Key-Value store, Wide Column Store, Multi-Model and Graph DBMS. In [3] a ranking of the most popular DB engines is presented.

Today, instead of having a single Relational Database Management System (DBMS) for the whole application, it is efficient and cost-effective to have several Data Base Engines, one for each type of data that the application handles. This concept is called *Polyglot Persistence* [43].

As [52] illustrates, polyglot persistence is very useful in the context of e-commerce applications that deal with a catalog, user access logs, financial information, shopping carts and purchase transactions, for example. The notion of polyglot persistence is built upon the observation that the *nature* of each data type is significantly different (i.e: user logs imply high volume of writes on multiple nodes, shopping carts need high availability and user sessions require rapid access for reads and writes).

As computing services started to decentralize, developers started to build applications that depended of several data-sources. By this time the use of Web Services and Service Oriented Architecture (SOA) became more popular [13].

Service Level Agreements (SLAs) According to *ITILv3's* official glossary [14], a Service Level Agreement (SLA) is "*an agreement between an IT service provider and a customer. A service level agreement describes the IT service, documents service level targets, and specifies the responsibilities of the IT service provider and the customer.*"

The agreement consists on a set of measurable constraints that a service provider must guarantee to its customers. In practical terms, it is a document that a service provider delivers to its consumers with minimum quality of service (QoS) metrics. If the service is delivered with a lower QoS than is promised on the SLA, consumers may be refunded or earn benefits that were accorded beforehand.

Systematic Mappings According to [41], "*A software engineering systematic map is a defined method to build a classification scheme and structure a software engineering field of interest.*" Systematic Mapping studies provide a global view of a given research field and identify the quantity, results, and the kinds of researches in this field.

A Systematic map is composed by a number of steps (Figure 1).

On the first step, "Definition of Research question", the questions that must be answered on the survey are defined. On the "Review Scope" step, researchers target the papers/journal sources that will be taken into consideration on the systematic map. After that, the "Search" step is done using a set of predefined search engines and a body of papers ("All papers") is retrieved. After an initial "Screening of the papers", the "Relevant papers" are chosen according to

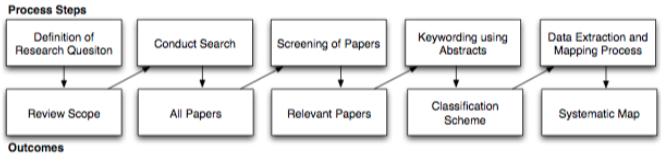

Figure 1: Systematic Mapping Steps [41].

inclusion and exclusion criteria defined by the research team. At this point, the papers that will participate of the study are selected. The selection is based on the title, abstracts and keywords of each paper ("Keywording using Abstracts"). After that, a "Classification Scheme" is built, defining different points-of-view (or facets) from which the body of papers will be classified. After matching each paper with the classification schema ("Data Extraction and Mapping Process"), the systematic mapping is performed. In this phase the relationships between the collected data (in the light of the classification scheme) are used to answer the research questions.

In the next section we build a systematic mapping to evaluate the use of SLA in database transition scenarios, from relational database management systems to NoSQL.

3 A Systematic Mapping

In this section we develop our systematic mapping, in the way described in section 2.

We start by defining our research scope and by pointing out the questions to be answered by the end of the survey (Section 3.1), followed by the selection of keywords that composed the search string (Section 3.2). The other steps of the systematic mapping procedure follow.

After that, we defined a classification schema and classified the relevant papers. These steps (screening of papers to classification scheme) were not linear as shown on Figure 1. We had to iterate on the screened papers a few times to find the ideal classification schema to our systematic map.

3.1 Definition of the research questions

As we wanted to investigate the way in which SLAs has been used in database transitioning processes, we proposed three main research questions and two associated questions, as follows:

RQ1) What are the reasons to change from RDBMSs to NoSQL solutions? This question answers *why* technological changes are needed on Information Systems and what is the *motivation* behind them.

AQ1.1) What are the pros and cons to migrate from RDBMSs to NoSQL solutions? This question is particularly important, as we also want to *point out the down sides* of DB migrations.

AQ1.2) How can we measure the overall improvements promised by this change? After a migration is performed, it is also important to *measure the benefits* of this migration, and this question evidences this point.

RQ2) How can SLAs be used to guide database transitioning processes from RDBMSs to NoSQL databases in cloud-based apps? This is one of the main questions of our study, as we want to evidence the state-of-art on the use of SLAs in migration scenarios.

RQ3) Is there a standard representation of SLAs in cloud services? We also try to search for standards on the representation of SLAs to verify if future works can be done under this field.

These are relevant questions for our survey as our focus is to determine how SLAs can be used in Database migrations scenarios. Our questions are intended to cover *why* it is good to migrate from RDBMSs to a NoSQL solution and *how* we can do it. In the end of our survey we also reveal what are the most popular technologies related to migrations.

3.2 Definition of keywords

Our research scope can be splitted in three main categories:
1) Databases: As we wanted to investigate RDBMS to NoSQL transitions, we defined that a representative query string for this category would be

NoSQL AND (rdbms OR relational)

2) Migration: A number of terms terms were oftenly linked with database migration and transition scenarios. Most of them, however, are consistently represented by the radicals "migr" and "transit". Thus the representative search string for this category is defined by:

migr OR transit**

3) Service level agreements: Service Level Agreements can also be represented by its acronym (SLA). A search string for this category is defined as

"Service Level Agreement" OR SLA

As we defined the three main categories of interest, what we wanted to survey is their conjunction. In this way, the final search string can be represented as:

(migr OR transit*) AND (NoSQL AND (rdbms OR relational)) AND (SLA OR "Service Level agreement")*

3.3 Conduct Search for Primary Studies (All Papers)

The next step of our systematic map was to identify relevant publishers. We surveyed academia experts and initially chose five main sources: **Springer, ACM, Sciencedirect, IEEE and Elsevier**. All of these publishers have relevant publications about databases and Service Level Agreements, as they index publications from reputable international conferences & journals.

Each of these publishers had their own search engine, however some inconsistent results were obtained when querying some of them directly. For instance, on one occasion IEEE search engine returned 0 results to the query string "*changes AND database AND nosql AND sla*". When we searched the same query on Google Scholar filtering *only IEEE publications* we found **90** results. We supposed that this erratic behavior is due to network traffic conditions. In order to mitigate the risks of having different results for the same query on different search engines, we used Google Scholar as a meta-search engine. It is important to mention that the use of Google Scholar made it possible to enlarge our search space. Our scope is not limited to the list of publishers identified above; these publishers represent only the minimal coverage of the literature in this systematic map.

During this step, we also noticed that 2009 would be the starting year of our survey, since this was the year when the term NoSQL was reintroduced to the community, by Johan Oskarsson (Last.fm)[4].

3.4 Search Strategy

Google Scholar returned 74 results for our search string. We have also manually included other particularly relevant publications that were picked from selected sources, such as [34], a master thesis about Relational Database Schema Migration to NoSQL.

These publications were either not published by the selected publishers or were not indexed by Google Scholar.

Our search strategy was composed by the following steps:

Step 1 - Basic Search: Search publications from 2009 to 2015 that matched the query string and add other relevant publications manually. This step was performed from 01 April 2015 to 11 April 2015 and returned 78 publications.

Step 2 - Dump the retrieved results on a spreadsheet. This spreadsheet is publicly available on [8].

Step 3 - Screening for keywords in of title and abstract: On the spreadsheet each publication has title, abstract, year and referenced URL. Title and abstract were considered on the initial screening. On this step we discarded publications that are notably not related to the topic of our study.

Step 4 - Apply inclusion/exclusion criteria: The inclusion/exclusion criteria are shown below. We first applied the inclusion criteria over the selected works, and then exclusion criteria removed the out of scope publications. Only papers clearly not relevant for the purposes of the study were removed. The publications that were considered on this study are marked on the spreadsheet with a

green background.

1. Inclusion Criteria:
    - The publication is about a migration from RDBMS to a NoSQL technology;
    - The main focus of the publication is on RDBMs or NoSQL systems;
    - The publication makes use or references a SLA.
2. Exclusion Criteria
    - Non-English written publications;
    - Access-restriction to the original publication (could not access the source of the publication);
    - Non-technical publications;
    - The work is about migrations within the same database DBMS;
    - The work is not related to databases or SLAs.
    - The work is related to databases but no comparison is made between two technologies.

A total of 34 publications were selected after this phase.

Step 5 - Fast Reading of the papers: We've performed a fast read of all publications that were not excluded on the previous step. This helped us to classify each paper accurately. This step is extensively detailed on the next section.

3.5 Classification of the Papers

We classified each selected publication in three facets: **Contribution Type**, **Technologies used** and **SLA representation**.

1. **Contribution type**: On the initial screening of papers we have defined 5 main types of contribution to our study. Other systematic mapping studies, such as [55] and [10] use a similar classification schema for contribution type.
    - Benchmark
    - Migration Experience Report
    - Bibliographical Review
    - Tool
    - Framework / Process

    When a migration of databases is described in a publication, we also classify specific data to answer the questions of this study: Source and Target technology of the migration, motivation to migrate and "Uses SLA or other artifact to validate the migration?";

2. **Technologies used**: To classify and rank the technologies mentioned on each paper we have developed a *"Batch-PDF-Tokens-Matcher"*. This tool matches a list of strings against a list of PDF files. The tool was made open-source and is publicly available on [5].

   In our case, we wanted to find the most cited technologies on relevant publications. As it was not possible to include all database types on this category, we chose to match only the 250 most-popular databases, ranked and made publicly available by [3].

3. **SLA representation**: One of the questions that our study wanted to answer was RQ3 (Is there a standard representation of SLAs in cloud services?). There are several ways to represent an SLA, and we clusterized some of these ways. It is also possible that no information is given about SLA representation on the publication, so we added two subcategories to this facet: "No SLA is used" and "Missing information".

   - Tool/Code/Database records
   - Language
   - Missing Information
   - No SLA is used

A graphical representation of our classification schema is detailed on Figure 2. We have listed only a subset of the several technology types that were found on the review.

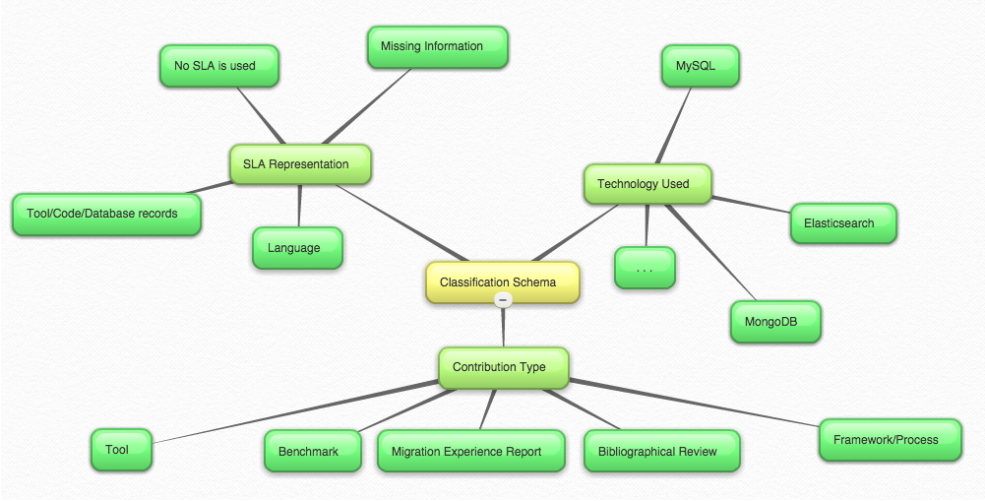

Figure 2: Classification scheme for selected publications.

4 Outcomes

As mentioned in Section 3.4, 74 publications were found matching our query string on IEEE, Elsevier, ACM, Springer and Sciencedirect using Google Scholar as a meta-search engine. We have manually added other 4 publications to our study. These publications were either not published by the selected publishers or were not indexed by Google Scholar.

34 out of the total of 78 publications were selected, after the screening of abstract and analysis of the inclusion/exclusion criteria[1].

The considered publications are: [30] [44] [58] [60] [37] [56] [9] [39] [59] [47] [45] [12] [37] [16] [51] [40] [26] [28] [20] [19] [57] [54] [22] [42] [46] [36] [24] [23] [29] [49] [48] [15] [27] [21].

The results are given in two steps: *Frequency Analysis between relevant criteria* and *Answers to research questions*.

4.1 Frequency Analysis between relevant criteria

Each analysis is presented below and is followed by a brief interpretation of the result.

- Publications by year: The publications by year table - Figure 3 - shows us that 2013 was the year when most of the publications were made on this research area. As we have just reached the middle of 2015 it is expected that not many publications are indexed on the search engines on this year.

Count of Year	
Row Labels	Total
2011	3
2012	5
2013	17
2014	8
2015	1
Grand Total	**34**

Figure 3: Publications by year.

- Frequency - Publication type vs Years: An interesting pattern can be found on the table Publication Type vs Years (Figure 4); "Migrations Experience Reports" were the main type of publication that our study aimed to discover, but only 3 publications of this type were found on our systematic map.

- Publications by country: Figure 5 shows that USA, Australia and Romania were responsible for almost 50% of the total selected publications.

[1] A column in our spreadsheet [8] contains the justification for the rejection of each discarded publication.

Row Labels	Benchmark	Migration Ex	Bibliographic	Tool	Framework/Process	Total
2011			2		2	4
2012	1	1	1	2	3	8
2013	5		5	5	6	21
2014	3	2		2	5	12
2015	1		1			2
Grand Total	**10**	**3**	**9**	**9**	**16**	47

Figure 4: Frequency - Publication type vs Years.

The other half of publications were distributed among other 16 countries, proving that the research area is not being developed by a single research group.

Count of Researchers Country	
Row Labels	Total
Australia	6
Austria	1
Canada	1
China	2
Estonia	1
France	1
Greece	1
India	1
Ireland	1
Japan	1
Romania	4
Russia	1
Saudi Arabia	1
Singapore	2
Spain	1
Taiwan	1
Tunis	1
Tunisia	1
USA	6
(blank)	
Grand Total	**34**

Figure 5: Publications by country.

- Popular Technologies: Figure 6 shows that Cassandra, MongoDB and MySQL were the most-cited databases on the papers that we analyzed. This chart can be used as a starting point to propose new works in the area of database migration. The full count of each technology is available on [6].
- SLA Representation: 29% of the publications didn't make any use of SLAs. Another big part - 32% - didn't even mention how the SLA is represented internally. 3 publications propose or use DSLs (Domain Specific Languages) to represent SLAs. This evidences a clear absence of standards when defining/using SLAs. The full table is presented on Figure 7. [6].

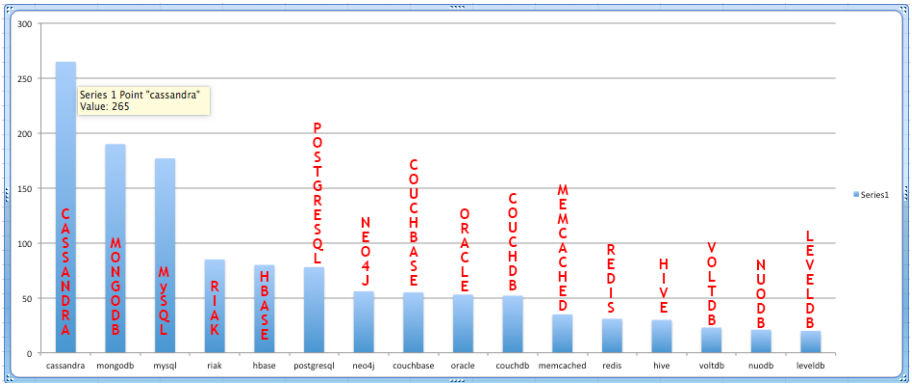

Figure 6: Most-cited database technologies.

Values	Column Labels 2011	2012	2013	2014	2015	Grand Total
Count of [SLA] Language		1	2			3
Count of [SLA] Do not specify how	1		7	2	1	11
Count of [SLA] Doesn't use SLAs		2	3	5		10
Count of [SLA] Other						
Count of [SLA] Tool/Implemented code/Database	2	2	5	1		10

Figure 7: SLAs representation.

4.2 Answers to research questions

In this mapping study we have surveyed the state of the art in transitions from RDBMSs to NoSQL databases. We created a rigorous protocol which analyzed 78 publications to answer the research questions that we identified previously. We may consider the answer to these questions as the main outcome of this paper. The answers are summarized below:

RQ1 (What are the reasons to change from RDBMSs to NoSQL solutions?) and **AQ1.2** - (What are the pros and cons to migrate from RDBMSs to NoSQL solutions?) investigate the motivation of a database transition. We found a number of reasons to migrate from a relational database to a NoSQL/Hybrid model. As the migration is most of the time motivated by the benefits of these models, the answer to the questions RQ1 and AQ1.2 is presented together.

The publications that reference the benefits and disadvantages of NoSQL / Hybrid solutions were [48] [15] and [27].

1. Benefits:
    - Segments of the data to be read and processed in parallel using a MapReduce framework, such as Hadoop;
    - Schema-less data model;
    - Support for large files;

- Scalability - relational databases tend to "Scale up", which is opposed to the "Scale horizontally" strategy used by hybrid solutions;

2. Disadvantages:

   - A big disadvantage on moving from a RDBMS to a NoSQL/Hybrid solution is the need for changes on the data models of the application. Several publications address this problem, such as [48] [17] and [35]
   - Another disadvantage of NoSQL and Hybrid solutions is the fact that these concepts are relatively new. As we mentioned previously, the term NoSQL was used to reference these types of databases for the first time in 2009[4]. The relational model has been in use for more than 30 years;As it is a new concept, it is particularly hard to find developers with a large experience in NoSQL databases.

(AQ1.2) How can we measure the overall improvements promised by this change?

No publication was found addressing the problem of measuring the overall improvements after a database transition. In fact, as it is shown on 4.1, there are few publications with the "Migration Experience Report" type.

Several benchmarking frameworks, such as TPC-H, TPC-DS and YCSB were identified[37] during our survey, though. These benchmarking frameworks could be a good starting point to develop new tools and specialized frameworks to solve this problem. This seems to be a promising research theme for future works.

RQ2) How can SLAs be used to guide database transitioning processes from RDBMSs to NoSQL databases in cloud-based apps? A number of works were found relating SLAs with Quality of Service (QoS) and Quality of Experience (QoE). Several publications, such as [56], [33] and [32] propose a SLA-centric approach to monitor and control the performance of cloud-based apps. [44], [58], [31] and [57] propose SLA-centric/User-Centric solutions to monitor the performance of web applications. All these solutions are technology-agnostic and could be used to monitor the performance improvements promised by a database transitioning process.

The question RQ2 was subject of discussion with industry experts and it was found out that there are some services, such as New Relic [7], Appsee [1] and Datadog [2] that provide SLA-monitoring tools for web apps.

RQ3) Is there a standard representation of SLAs in cloud services? The selected publications did not present a standardized and common representation for SLAs. In fact, 32% of the selected publications did not even mention how the SLA was represented. 29% represented the SLAs as tables/documents stored on databases without any sla-oriented guidelines. This evidences a clear absense of standards when defining/using SLAs.

[21] proposes SYBL: *An Extensible Language for Controlling Elasticity in Cloud Applications. SYBL allows specifying in detail elasticity monitoring,*

constraints, and strategies at different levels of cloud applications, including the whole application, application component, and within application component code. In other words, SYBL can also be seen as a language to specify SLAs in cloud environments.

Some other initiatives that tried to define the mechanisms by which Web-service SLAs are established are WS-Agreement [11] and WSLA [38]. Also, [25] presents a framework for managing the mappings of the Low-level resource Metrics to High-level SLAs (LoM2HiS framework).

It is worth to notice that future works on standardizing the representations of SLAs are needed.

5 Conclusions

By analyzing the charts and the answers to the research questions we can conclude that the research area of this mapping study is still not mature.

There is a lack of consensus and standards on this research area, as no official guidelines for migrating from a relational database to a NoSQL/Hybrid database were found. It was also not found a standard way to represent and create SLAs. These two points seems to be promising research topics and will be covered on future works.

References

[1] Appsee. `https://www.appsee.com/`. Accessed: 2014-05-01.

[2] Datadog. `https://www.datadoghq.com/`. Accessed: 2014-05-01.

[3] Db engines ranking. `http://db-engines.com/en/ranking`. Accessed: 2014-05-01.

[4] Eric evans blog - nosql. `http://blog.sym-link.com/2009/05/12/nosql_2009.html`. Accessed: 2014-05-01.

[5] Fabio leal - pdf tokens matcher. `https://github.com/fabiosl/pdf-tokens-finder`. Accessed: 2015-05-01.

[6] Full db json. `https://gist.github.com/fabiosl/18b5e826c9daebda5165`. Accessed: 2015-04-21.

[7] New relic. `http://newrelic.com`. Accessed: 2014-05-01.

[8] Systematic mapping spreadsheet. `https://docs.google.com/spreadsheets/d/1N3DboEqthdiKG3VDMKlqyjFdHC3pxrN1XtXtNoNrhX8/edit#gid=0`. Accessed: 2015-05-01.

[9] Ebtesam Alomari, Ahmed Barnawi, and Sherif Sakr. Cdport: A framework of data portability in cloud platforms. In *Proceedings of the 16th International Conference on Information Integration and Web-based Applications*

& Services, iiWAS '14, pages 126–133, New York, NY, USA, 2014. ACM.

[10] David Ameller, Xavier Burgus, Oriol Collell, Dolors Costal, Xavier Franch, and Mike P. Papazoglou. Development of service-oriented architectures using model-driven development: A mapping study. *Information and Software Technology*, 62(0):42 – 66, 2015.

[11] Alain Andrieux, Karl Czajkowski, Asit Dan, Kate Keahey, Heiko Ludwig, Toshiyuki Nakata, Jim Pruyne, John Rofrano, Steve Tuecke, and Ming Xu. Web Services Agreement Specification (WS-Agreement). Technical report, Global Grid Forum, Grid Resource Allocation Agreement Protocol (GRAAP) WG, September 2005.

[12] Vasilios Andrikopoulos, Tobias Binz, Frank Leymann, and Steve Strauch. How to adapt applications for the cloud environment. *Computing*, 95(6):493–535, 2013.

[13] Michael Armbrust, O Fox, Rean Griffith, Anthony D. Joseph, Y Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. M.: Above the clouds: A berkeley view of cloud computing. Technical report, 2009.

[14] Axelos. Best management practice portfolio: common glossary of terms and definitions, October 2012.

[15] Cristina BĂZĂR and Cosmin Sebastian IOSIF. The transition from rdbms to nosql. a comparative analysis of three popular non-relational solutions: Cassandra, mongodb and couchbase. *Database Systems Journal BOARD*, page 49.

[16] E.A. Boytsov. Designing and development of an imitation model of a multitenant database cluster. *Automatic Control and Computer Sciences*, 48(7):437–444, 2014.

[17] Rick Cattell. Scalable sql and nosql data stores. *SIGMOD Rec.*, 39(4):12–27, May 2011.

[18] A. et al. CONNOLY, D. FOX. A view of cloud computing - http://delivery.acm.org/10.1145/1730000/1721672/p50-armbrust.pdf, April 2010. [Online; posted 01-April-2010].

[19] A. Copie, T.-F. Fortis, and V.I. Munteanu. Determining the performance of the databases in the context of cloud governance. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on*, pages 227–234, Oct 2013.

[20] A. Copie, T.-F. Fortis, V.I. Munteanu, and V. Negru. Service datastores in cloud governance. In *Parallel and Distributed Processing with Applications (ISPA), 2012 IEEE 10th International Symposium on*, pages 473–478, July 2012.

[21] G. Copil, D. Moldovan, Hong-Linh Truong, and S. Dustdar. Sybl: An extensible language for controlling elasticity in cloud applications. In *Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on*, pages 112–119, May 2013.

[22] M. Dayarathna and T. Suzumura. Towards scalable distributed graph database engine for hybrid clouds. In *Data-Intensive Computing in the Clouds (DataCloud), 2014 5th International Workshop on*, pages 1–8, Nov 2014.

[23] Miyuru Dayarathna and Toyotaro Suzumura. Graph database benchmarking on cloud environments with xgdbench. *Automated Software Engineering*, 21(4):509–533, 2014.

[24] Aaron J. Elmore, Sudipto Das, Alexander Pucher, Divyakant Agrawal, Amr El Abbadi, and Xifeng Yan. Characterizing tenant behavior for placement and crisis mitigation in multitenant dbmss. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 517–528, New York, NY, USA, 2013. ACM.

[25] V.C. Emeakaroha, I. Brandic, M. Maurer, and S. Dustdar. Low level metrics to high level slas - lom2his framework: Bridging the gap between monitored metrics and sla parameters in cloud environments. In *High Performance Computing and Simulation (HPCS), 2010 International Conference on*, pages 48–54, June 2010.

[26] Georgios Giannikis, Darko Makreshanski, Gustavo Alonso, and Donald Kossmann. Workload optimization using shareddb. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 1045–1048, New York, NY, USA, 2013. ACM.

[27] Abraham Gomez, Rafik Ouanouki, Alain April, and Alain Abran. Building an experiment baseline in migration process from sql databases to column oriented no-sql databases. *J Inform Tech Softw Eng*, 4(137):2, 2014.

[28] Katarina Grolinger, Wilson Higashino, Abhinav Tiwari, and Miriam Capretz. Data management in cloud environments: Nosql and newsql data stores. *Journal of Cloud Computing: Advances, Systems and Applications*, 2(1):22, 2013.

[29] Han Hu, Yonggang Wen, Tat-Seng Chua, and Xuelong Li. Toward scalable systems for big data analytics: A technology tutorial. *Access, IEEE*, 2:652–687, 2014.

[30] Chao-Wen Huang, Wan-Hsun Hu, Chia-Chun Shih, Bo-Ting Lin, and Chien-Wei Cheng. The improvement of auto-scaling mechanism for distributed database - a case study for mongodb. In *Network Operations and Management Symposium (APNOMS), 2013 15th Asia-Pacific*, pages 1–3, Sept 2013.

[31] M. Klems, D. Bermbach, and R. Weinert. A runtime quality measurement framework for cloud database service systems. In *Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the*, pages 38–46, Sept 2012.

[32] Markus Klems, David Bermbach, and Rene Weinert. A runtime quality measurement framework for cloud database service systems. In *Proceedings of the 2012 Eighth International Conference on the Quality of Information and Communications Technology*, QUATIC '12, pages 38–46, Washington, DC, USA, 2012. IEEE Computer Society.

[33] Ioannis Konstantinou, Evangelos Angelou, Dimitrios Tsoumakos, Christina Boumpouka, Nectarios Koziris, and Spyros Sioutas. Tiramola: Elastic nosql provisioning through a cloud management platform. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, SIGMOD '12, pages 725–728, New York, NY, USA, 2012. ACM.

[34] Rilinda Lamllari. Extending a methodology for migration of the database layer to the cloud considering relational database schema migration to nosql. Master's thesis, University of Stuttgart, 2013.

[35] C. Mohan. History repeats itself: Sensible and nonsensql aspects of the nosql hoopla. In *Proceedings of the 16th International Conference on Extending Database Technology*, EDBT '13, pages 11–16, New York, NY, USA, 2013. ACM.

[36] Jess Montes, Alberto Snchez, Bunjamin Memishi, Mara S. Prez, and Gabriel Antoniu. Gmone: A complete approach to cloud monitoring. *Future Generation Computer Systems*, 29(8):2026 – 2040, 2013. Including Special sections: Advanced Cloud Monitoring Systems & The fourth {IEEE} International Conference on e-Science 2011 e-Science Applications and Tools & Cluster, Grid, and Cloud Computing.

[37] R. Moussa. Benchmarking data warehouse systems in the cloud. In *Computer Systems and Applications (AICCSA), 2013 ACS International Conference on*, pages 1–8, May 2013.

[38] S. Nepal, J. Zic, and Shiping Chen. Wsla+: Web service level agreement language for collaborations. In *Services Computing, 2008. SCC '08. IEEE International Conference on*, volume 2, pages 485–488, July 2008.

[39] C. Pahl and Huanhuan Xiong. Migration to paas clouds - migration process and architectural concerns. In *Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the*, pages 86–91, Sept 2013.

[40] Dana Petcu, Georgiana Macariu, Silviu Panica, and Ciprian Crciun. Portable cloud applicationsfrom theory to practice. *Future Generation Computer Systems*, 29(6):1417 – 1430, 2013. Including Special sections:

High Performance Computing in the Cloud & Resource Discovery Mechanisms for {P2P} Systems.

[41] Kai Petersen, Robert Feldt, Shahid Mujtaba, and Michael Mattsson. Systematic mapping studies in software engineering. In *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, EASE'08, pages 68–77, Swinton, UK, UK, 2008. British Computer Society.

[42] Lin Qiao, Kapil Surlaker, Shirshanka Das, Tom Quiggle, Bob Schulman, Bhaskar Ghosh, Antony Curtis, Oliver Seeliger, Zhen Zhang, Aditya Auradar, Chris Beaver, Gregory Brandt, Mihir Gandhi, Kishore Gopalakrishna, Wai Ip, Swaroop Jgadish, Shi Lu, Alexander Pachev, Aditya Ramesh, Abraham Sebastian, Rupa Shanbhag, Subbu Subramaniam, Yun Sun, Sajid Topiwala, Cuong Tran, Jemiah Westerman, and David Zhang. On brewing fresh espresso: Linkedin's distributed data serving platform. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 1135–1146, New York, NY, USA, 2013. ACM.

[43] Pramod J Sadalage and Martin Fowler. *NoSQL distilled: a brief guide to the emerging world of polyglot persistence*. Pearson Education, 2012.

[44] S. Sakr and A. Liu. Sla-based and consumer-centric dynamic provisioning for cloud databases. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 360–367, June 2012.

[45] S. Sakr and A. Liu. Is your cloud-hosted database truly elastic? In *Services (SERVICES), 2013 IEEE Ninth World Congress on*, pages 444–447, June 2013.

[46] S. Sakr, A. Liu, D.M. Batista, and M. Alomari. A survey of large scale data management approaches in cloud environments. *Communications Surveys Tutorials, IEEE*, 13(3):311–336, Third 2011.

[47] Sherif Sakr. Cloud-hosted databases: technologies, challenges and opportunities. *Cluster Computing*, 17(2):487–502, 2014.

[48] Aaron Schram and Kenneth M. Anderson. Mysql to nosql: Data modeling challenges in supporting scalability. In *Proceedings of the 3rd Annual Conference on Systems, Programming, and Applications: Software for Humanity*, SPLASH '12, pages 191–202, New York, NY, USA, 2012. ACM.

[49] David Shue and Michael J. Freedman. From application requests to virtual iops: Provisioned key-value storage with libra. In *Proceedings of the Ninth European Conference on Computer Systems*, EuroSys '14, pages 17:1–17:14, New York, NY, USA, 2014. ACM.

[50] K. Shvachko, Hairong Kuang, S. Radia, and R. Chansler. The hadoop distributed file system. In *Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on*, pages 1–10, May 2010.

[51] B. Sodhi and T.V. Prabhakar. Assessing suitability of cloud oriented platforms for application development. In *Software Architecture (WICSA), 2011 9th Working IEEE/IFIP Conference on*, pages 328–335, June 2011.

[52] Genoveva Vargas Solar. Addressing data management on the cloud: tackling the big data challenges, May 2014.

[53] Alfred Spector, Peter Norvig, and Slav Petrov. Google's hybrid approach to research. *Commun. ACM*, 55(7):34–37, July 2012.

[54] S.N. Srirama and A. Ostovar. Optimal resource provisioning for scaling enterprise applications on the cloud. In *Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6th International Conference on*, pages 262–271, Dec 2014.

[55] A. Tahir and S.G. Macdonell. A systematic mapping study on dynamic metrics and software quality. In *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*, pages 326–335, Sept 2012.

[56] Pengcheng Xiong. Dynamic management of resources and workloads for rdbms in cloud: A control-theoretic approach. In *Proceedings of the on SIGMOD/PODS 2012 PhD Symposium*, PhD '12, pages 63–68, New York, NY, USA, 2012. ACM.

[57] Pengcheng Xiong, Yun Chi, Shenghuo Zhu, Junichi Tatemura, Calton Pu, and Hakan HacigümüŞ. Activesla: A profit-oriented admission control framework for database-as-a-service providers. In *Proceedings of the 2Nd ACM Symposium on Cloud Computing*, SOCC '11, pages 15:1–15:14, New York, NY, USA, 2011. ACM.

[58] L. Zhao, S. Sakr, and A. Liu. A framework for consumer-centric sla management of cloud-hosted databases. *Services Computing, IEEE Transactions on*, PP(99):1–1, 2013.

[59] Liang Zhao, S. Sakr, and A. Liu. Application-managed replication controller for cloud-hosted databases. In *Cloud Computing (CLOUD), 2012 IEEE 5th International Conference on*, pages 922–929, June 2012.

[60] Zibin Zheng, Jieming Zhu, and M.R. Lyu. Service-generated big data and big data-as-a-service: An overview. In *Big Data (BigData Congress), 2013 IEEE International Congress on*, pages 403–410, June 2013.