Scalability, Elasticity, and Efficiency in Cloud Computing: a Systematic Literature Review of Definitions and Metrics

CONFERENCE PAPER · MAY 2015

DOI: 10.1145/2737182.2737185

Sebastian Lehrig

Technische Universität Chemnitz

Hendrik Eikerling

Universität Paderborn

Steffen Becker

Technische Universität Chemnitz

Scalability, Elasticity, and Efficiency in Cloud Computing: a Systematic Literature Review of Definitions and Metrics[*]

Sebastian Lehrig Hendrik Eikerling Steffen Becker

{sebastian.lehrig|hendrik.eikerling|steffen.becker}@informatik.tu-chemnitz.de

Software Engineering Chair
Chemnitz University of Technology, Chemnitz, Germany

ABSTRACT

Context In cloud computing, there is a multitude of definitions and metrics for scalability, elasticity, and efficiency. However, stakeholders have little guidance for choosing fitting definitions and metrics for these quality properties, thus leading to potential misunderstandings. For example, cloud consumers and providers cannot negotiate reliable and quantitative service level objectives directly understood by each stakeholder.

Objectives Therefore, we examine existing definitions and metrics for these quality properties from the viewpoint of cloud consumers, cloud providers, and software architects with regard to commonly used concepts.

Methods We execute a systematic literature review (SLR), reproducibly collecting common concepts in definitions and metrics for scalability, elasticity, and efficiency. As quality selection criteria, we assess whether existing literature differentiates the three properties, exemplifies metrics, and considers typical cloud characteristics and cloud roles.

Results Our SLR yields 418 initial results from which we select 20 for in-depth evaluation based on our quality selection criteria. In our evaluation, we recommend concepts, definitions, and metrics for each property.

Conclusions Software architects can use our recommendations to analyze the quality of cloud computing applications. Cloud providers and cloud consumers can specify service level objectives based on our metric suggestions.

Categories and Subject Descriptors

D.2.8 [**Software Engineering**]: Metrics—*Scalability, Elasticity, Efficiency*

Keywords

Systematic Literature Review, Definitions, Metrics, Cloud

1. INTRODUCTION

Typical requirements of cloud computing applications target scalability, elasticity, and efficiency [3, 14]. These quality properties are unstandardized and a multitude of definitions and metrics are available in existing literature. This multitude shows the importance of these properties and induces the need for stakeholders (e.g., software architects, cloud consumers, and cloud providers) to be guided in choosing definitions and metrics that fit in their context.

However, such a guidance currently does not exist: stakeholders have to manually find fitting definitions and metrics in existing literature and adapt such definitions to their context. Such tasks either lead to high effort if an exhaustive amount of literature is investigated or to insufficient results if too little literature is surveyed; important concepts can then easily be missed. For example, some authors (e.g., [10, 8, 7]) do not consider the concept of time in their elasticity definition while other authors (e.g., [14, 20, 9]) argue for its consideration. Such situations potentially lead to misunderstandings. For example, cloud consumers and providers cannot negotiate reliable and quantitative service level objectives directly understood by each stakeholder.

In related work, initial attempts have been made to provide guidance in finding fitting concepts, definitions, and metrics. Duboc [12] systematically analyzes scalability definitions for software in general. She therefore provides a broad guidance, however, neglects cloud computing characteristics such as rapid elasticity. Herbst et al. [14] take such characteristics into account and propose corresponding definitions and metrics. They focus on definitions and metrics for benchmarking only, thus, lack guidance in other contexts such as for specifying service level objectives.

To tackle the lack of guidance, we systematically examine existing definitions and metrics for scalability, elasticity, and efficiency from the viewpoint of cloud consumers, cloud providers, and software architects with regard to commonly used concepts. Methodically, we execute a systematic literature review (SLR) [18], in which we reproducibly collect common concepts in definitions and metrics for such properties. As quality selection criteria, we assess whether existing literature differentiates the three properties, exemplifies metrics, and considers typical cloud characteristics and cloud roles. Our SLR yields 418 initial results from which we select 20 for in-depth evaluation based on our quality selection criteria. In our evaluation, we quantify the importance of concepts and derive definitions tied to cloud computing.

The contribution of this paper is the execution of our SLR. This includes (1) a reproducible identification of important, potentially property-relating concepts (e.g., capacity) and

[*]The research leading to these results has received funding from the European Seventh Framework Programme (FP7/2007-2013) under grant no 317704 (CloudScale). This work is supported by the German Research Foundation (DFG) within the Collaborative Research Centre “On-The-Fly Computing” (SFB 901).

QoSA'15, May 4–8, 2015, Montréal, QC, Canada.

http://dx.doi.org/10.1145/2737182.2737185.

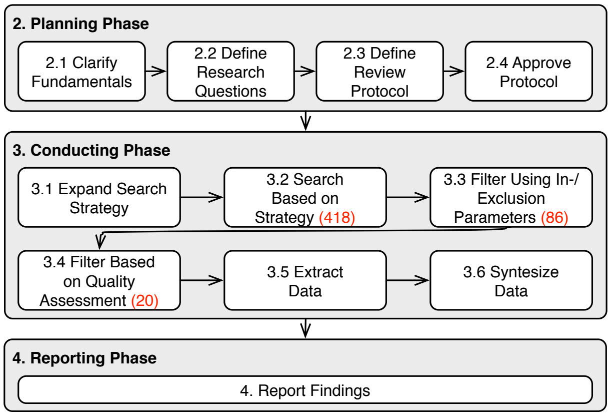
2. Planning Phase
2.1 Clarify Fundamentals
2.2 Define Research Questions
2.3 Define Review Protocol
2.4 Approve Protocol
3. Conducting Phase
3.1 Expand Search Strategy
3.2 Search Based on Strategy (418)
3.3 Filter Using In-/Exclusion Parameters (86)
3.4 Filter Based on Quality Assessment (20)
3.5 Extract Data
3.6 Syntesize Data
4. Reporting Phase
4. Report Findings

Figure 1: Systematic Literature Review – Overview

(2) suggestions for definitions and metrics for scalability, elasticity, and efficiency tied to cloud computing:

Definition 1 (Capacity). *Capacity is the maximum workload a cloud layer can handle as bound by its SLOs.*

Definition 2 (Scalability). *Scalability is the ability of a cloud layer to increase its capacity by expanding its quantity of consumed lower-layer services.*

Definition 3 (Elasticity). *Elasticity is the degree a cloud layer autonomously adapts capacity to workload over time.*

Definition 4 (Efficiency). *Efficiency is a measure relating demanded capacity to consumed services over time.*

This paper is structured as follows. As shown in Fig. 1, we follow the typical SLR phases of planning (Sec. 2), conducting (Sec. 3), and reporting (Sec. 4). We annotated each conducted action with the corresponding section and the number of found sources where feasible. After describing our SLR, we discuss potential threats to the validity of our work in Sec. 5 and related work in Sec. 6. Finally, we conclude with a summary and future work in Sec. 7.

2. PLANNING PHASE

In the planning phase, we devise the review protocol, defining the further progression of the SLR (search strategy, specification of filtering process). The planning phase involves fundamentals (Sec. 2.1), research questions (Sec. 2.2), the review protocol (Sec. 2.3) and its approval (Sec. 2.4).

2.1 Clarify Fundamentals

Our object of investigation are properties in cloud computing environments and their metrics. First, we need to clarify in which cases a source deals with our domain (cloud computing). Second, regarding metrics, we want to elaborate on their limitations within this domain: who measures metrics and where; how are they used for quantification?

Regarding the first issue, we stick to the well-accepted and standardized NIST definition of cloud computing [23]. In our SLR, a source deals with cloud computing if it cites the NIST definition directly and/or if it considers at least one of NIST's typical cloud computing characteristics [23]:

On-demand self-service: Enables consumers to request additional resources on demand, without requiring human interaction on the provider side.

Rapid elasticity: Allows cloud services to scale-in and to scale-out, depending on consumer demand, sometimes automatically through an elasticity management.

Measured service: Ensures that consumers only pay for the used resources (pay-per-use).

Regarding the second issue, we classify metrics based on who can measure and use them. The NIST describes the following cloud roles we use for this classification [22]:

Cloud consumer: Cloud consumers pick, request, and use services made available by cloud providers. Cloud consumers negotiate conditions of service operation with cloud providers; recorded in service level objectives (SLOs).

Cloud provider: Cloud providers are responsible for facilitating and managing cloud services for others. To fulfill these responsibilities, cloud providers acquire computing infrastructures and arrange network access for consumers.

Software architects can measure in either of these roles (for internal software quality as cloud provider, for external quality dependencies as cloud consumer). Moreover, these roles can measure and use metrics at different cloud layers; metrics can be layer-dependent. We therefore use NIST's service models/layers [23] to classify found metrics:

Software-as-a-Service (SaaS): At this layer, applications are provided to cloud consumers. Cloud providers are responsible for deployment, configuration, and maintenance while cloud consumers have limited administrative control.

Platform-as-a-Service (PaaS): At this layer, cloud consumers are provided a platform, e.g., a runtime environment for (web) applications. Cloud consumers can develop, test, configure, and deploy custom applications, whereas cloud providers administrate the underlying infrastructure.

Infrastructure-as-a-Service (IaaS): At this layer, cloud providers offer fundamental computing resources, i.e., virtual machines, network-accessible storage, and network infrastructure components. Cloud consumers can use the provided resources arbitrarily.

2.2 Define Research Questions

Next, we define the research questions that form the basis of the SLR. Requirements for scalability, elasticity, and efficiency come up frequently when software architects design cloud applications and when cloud consumer and provider negotiate service level objectives. Our research questions target these three properties as their definition is unclear.

Because we are also interested in measuring the scalability, elasticity, and efficiency of cloud environments, metrics are necessary in addition to definitions of the three properties. Therefore, we pose two research questions, the first one ($\text{RQ}_{\text{Def.}}$) targeting property definitions and the second one ($\text{RQ}_{\text{Met.}}$) targeting metrics for the three properties:

$\text{RQ}_{\text{Def.}}$: "How to define scalability, elasticity, and efficiency in the context of cloud computing?"

$\text{RQ}_{\text{Met.}}$: "What metrics can be used to compare the scalability, efficiency, and elasticity of different cloud computing services and how are they measured and used?"

2.3 Define Review Protocol

The most integral part of the planning phase is defining the review protocol because subsequent actions depend on this protocol. We defined our protocol in iterations of internal expert reviews and according adjustments (see Sec. 2.4). This section introduces our final review protocol.

2.3.1 *Search Strategy*

The search strategy defines the search terms and phrases (built using these terms) that are finally used to search for material. The search strategy is database-independent because Kitchenham [18] suggests to expand the search strategy during the conducting phase, when actually performing the search. Therefore, we expand the search phrases and tailor them to the searched database (e.g., because of length limitations on search phrases) in Section 3.1. In our case, we limit the searched databases to **Google Scholar**, a suggested database by Kitchenham [18].

Based on the research questions of Sec. 2.2, we derive the following basic search terms: (1) **scalability**, **elasticity**, and **efficiency** (properties), (2) **cloud computing** (domain), and (3) **metric** (focus of second research question). Connecting these search terms using Boolean operators results in these search phrases:

Phrase$_{\mathbf{Def.}}$: *(scalability OR elasticity OR efficiency) AND "cloud computing"*

Phrase$_{\mathbf{Met.}}$: *metric AND (scalability OR elasticity OR efficiency) AND "cloud computing"*

2.3.2 *Time Frame*

We limit the time frame covered by our search to sources published between **2005** and the time of search, i.e., **April 2014**. This limitation of time frame is done because the year 2005 approximately coincides with the emergence of the term "cloud computing".

2.3.3 *In- and Exclusion Parameters*

As a first step of filtering results of searching, the review protocol defines in- and exclusion parameters. This first filtering is important for coping with a huge amount of found sources. We include a found source in our SLR if:

- it deals exclusively with the definition of either one of the three properties scalability, elasticity, and efficiency (rationale: the source's goal correlates with the goal of this work)
- an explicit, detailed definition of one or more of the three properties in a cloud computing environment is given in the source (rationale: collecting and comparing as many definitions as possible is the goal of this SLR)
- metrics for measuring scalability, elasticity, and efficiency of cloud computing environments are discussed and/or exemplified (rationale: metrics are part of the second research question that this SLR is based on)

We exclude a source if:

- the general topic of the source is not computer science (e.g., if a search containing the word "cloud" yields a result in meteorology; rationale: out-of-scope)
- it uses none of the terms scalability, elasticity, and efficiency or their synonyms (rationale: we expect that no definition for either property is given in the source)
- it is in the form of a panel discussion, preface, tutorial, book review, or presented slide (rationale: too informal nature of these formats)
- the source is in a language other than English (rationale: English is accessible by a wide audience)
- the full text is unavailable through the library of the University of Paderborn[1] without payment (rationale: purchasing all texts would surpass the allocated budget)

We first apply the in- and then the exclusion parameters. Once we included a found source, we did not exclude it anymore. Moreover, we organize the in- and exclusion in stages to simplify the process. First, we apply the parameters to the title of the source. Second, if we cannot decide based on the title, we evaluate abstract and conclusion (as proposed by Brereton [6]). Third, if we still cannot decide, we read the whole source or, in the case of sources longer than 50 pages, scan the source using keywords from our parameters.

2.3.4 *Quality Indicators and their Evaluation*

After initial filtering of sources, we evaluate the quality of remaining sources. For this purpose, we have to define and evaluate quality indicators. Such quality indicators can be fulfilled fully, partly, or not at all. In our case, sources that pass quality evaluation have to fulfill each quality criterion at least partly. Based on our objectives, we derive the following quality indicators:

Q$_{\mathbf{NIST}}$: Does the source consider the NIST cloud computing definition and/or its characteristics [23], e.g., on-demand self-service, rapid elasticity, and measured service? (rationale: sources should use a widely accepted definition of cloud computing to avoid misunderstandings)

Q$_{\mathbf{Metrics}}$: Does the source exemplify metrics to quantify scalability, elasticity, and/or efficiency? (rationale: metrics are necessary for quantification)

Q$_{\mathbf{Roles}}$: Does the source consider the NIST cloud roles "cloud provider" and/or "cloud consumer"? (rationale: cloud consumers/providers have different views on a system's quality indicators, potentially leading to distinct metrics)

Q$_{\mathbf{Relation}}$: Does the source facilitate the relation of a property to the other properties? (rationale: there are interconnections between properties, e.g., elasticity may depend on scalability; this should be reflected to avoid contradictions)

2.3.5 *Data Extraction*

The data from sources that passed quality evaluation needs to be extracted systematically. Therefore, the review protocol specifies what data has to be extracted.

We extract the following data: **authors/date of publication**, **issue** the authors want to solve, their **approach**, the actual **definitions** of the properties and/or **metrics**, and the **concepts** used in definitions (workload, resources, etc.) for systematically characterizing these. For our extraction, we quote sources directly where possible to minimize bias. If a source is too complicated or relevant parts are too long, we add a reference in the data extraction documentation. Generally, we keep extracted data as concise as possible to only overview most significant data.

In case a specific definition is repeatedly referenced in sources, we omit extracting data each time but reference the original source in the definition. An example for a recurring definition is NIST's definition of rapid elasticity [23]. We do not extract references that only occur once and are not elaborated on in the source.

2.3.6 *Data Synthesis*

After extraction, we synthesize extracted data to prepare the interpretation of found results. In our case, we synthesize the distribution of publication dates, a summary of our filtering process (how many sources did we filter based on which criteria?), an evaluation of the quality parameter fulfillments, and statistics about found definitions and metrics.

[1] See `http://www.ub.uni-paderborn.de/recherchieren/primo/datenquellen-primocentral.shtml` (2015/02/16)

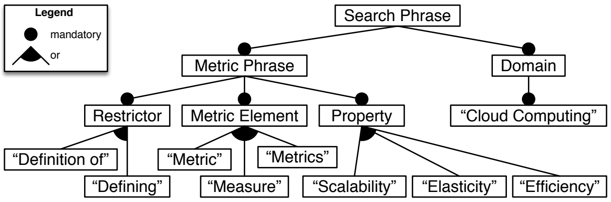

Figure 2: Feature model describing the structure of the metric search phrase.

We tabulate extracted concepts to show in which sources and how often they appear in definitions. For example, we find out that "workload" is a concept common to a large fraction of found elasticity definitions. Based on such a quantification, we derive suggestions for quality definitions, e.g., "workload" should be part of an elasticity definition.

2.4 Approve Protocol

We internally approved our review protocol in iterations as follows. First, Eikerling suggested a protocol, let Lehrig review it, and subsequently revised it according to Lehrig's remarks. We iterated this process two times. Second, we handed the revised protocol to another reviewer of our department and proceeded analogously in two iterations. Third, Becker made a final review, which led to the approved review protocol presented in the previous section.

All reviewers that took part in this review process have experience with software engineering of cloud computing applications. Two reviewers are research assistants, one reviewer is a professor of software engineering.

3. CONDUCTING PHASE

In the conducting phase, we execute the SLR according to our review protocol (cf., Sec. 2). This phase includes the expansion of the basic search phrases defined in the review protocol, the gathering and filtering of material, extracting data from the collected sources, and the data synthesis. We provide all raw data, documentation, and assessments at a dedicated website[2]; here we only summarize the results.

3.1 Expand Search Strategy

To focus the search results on the desired domain and increase the number of relevant search results, we expand the basic search phrases from the review protocol. Such an expansion adds synonyms and restricting terms, e.g., to focus on definitions of desired properties.

The feature model in Fig. 2 exemplifies our expansion for the metrics search phrase $\mathbf{Phrase_{Met.}}$ of Sec. 2.3.1. For example, we handle "Metric", "Measure", and "Metrics" as synonyms within the "Metric Element" of the phrase. The structure for the phrase $\mathbf{Phrase_{Def.}}$ is similar but omits the "Metric Element" and includes more synonyms for the domain.

Because Google Scholar imposes length restrictions on search phrases, we split the complete phrase at "OR" operators such that we can perform multiple searches with shorter phrases and subsequently merge the results.

3.2 Search for Material Based on Strategy

Our search process yielded 486 sources (262 from the definitions search phrase; 224 from the metric search phrase). In each subgroup, we removed found duplicates, i.e., sources that where the same but ended up as two separate results in Google Scholar. Without duplicates, we ended up with 418 sources (214 from the definitions search phrase; 204 from the metric search phrase).

3.3 Filter Using In- and Exclusion Parameters

After applying in- and exclusion parameters (specified in Sec. 2.3.3), we reduced the number of results for definitions and metrics to 36 and 54 sources, respectively. We combined these sources to a common pool of 90 sources. Because our combination included 4 duplicates, we removed these from the common pool and ended up with 86 sources left.

3.4 Filter Based on Quality Assessment

We further filtered the remaining 86 sources by assessing the quality indicators specified in Sec. 2.3.4. Our quality assessment resulted in 20 sources that fulfill each quality indicator at least partly. We finally use these 20 sources to extract and synthesize data.

3.5 Extract Data

Next, we extracted data from the 20 sources left after quality assessment. To lower biased extraction, we proceeded as follows. Eikerling processed all 20 sources. In parallel, Lehrig extracted data for 5 random samples of these sources and finally compared his results with the ones of Eikerling.

3.5.1 Main Data Extraction

Eikerling extracted data from the 20 remaining sources based on our extraction strategy (cf. Sec. 2.3.5). When sources were too extensive to be quoted directly and to still keep a concise extraction form, e.g., in the case of Bersani et al.'s formal language [4], Eikerling quoted only the definitions and referenced to the paper for further detail. For all sources, at least the year of publication could be extracted, in some cases the exact dates.

3.5.2 Sample Data Extraction by Another Author

Lehrig's results on the 5 samples were in line with the ones from the previous section. Because this indicated that data extraction can be reproduced using our review protocol, we agreed to continue with data synthesis.

3.6 Synthesize Data

Having extracted all relevant data, we synthesize the data in the next action of the conducting phase. In this synthesis, we summarize found definitions and metrics for each of the properties scalability, elasticity, and efficiency. The extracted concepts from definitions are used in definitions and metrics alike, so we examine them together.

For metrics, Tab. 1 summarizes our classification. For each cell, X denotes found concepts directly stated within the source while (X) denotes concepts that we identified by interpreting descriptions within the source (as a direct statement was unavailable).

For definitions, Tab. 2 to 4 overview our findings. We only added concepts that are directly stated within the source (hence, we only used X within cells).

[2] `http://code.google.com/p/ba-heikerl` (2015/02/16)

Table 1: Classification of metrics

Reference(s)	Scalability	Elasticity	Efficiency	Cloud Consumer	Cloud Provider	SaaS	PaaS	IaaS
[9]	X			(X)	X	(X)	(X)	(X)
[29, 19]	X			(X)	(X)	X		
[26]	X			(X)	(X)	(X)		
[20, 13]		X		(X)	(X)	(X)	(X)	
[25]		X		X				(X)
[15, 10]		X		(X)	(X)	(X)	(X)	(X)
[24]		X		X	X	(X)	(X)	(X)
[4]		X			(X)			(X)
[11]		X			(X)			X
[17, 27, 28]		X		X	(X)		(X)	
[1]		X		X	X		X	
[8, 7]			X		(X)			(X)
[2]			X	(X)		(X)		
[5]			X		(X)			(X)

3.6.1 Scalability Definitions and Metrics

Table 2 shows the concepts used by found definitions to characterize the scalability of cloud computing environments. Most sources utilize workload and assigned resources of a system to define scalability. Some authors replace workload with demand or the amount of services consumed by a lower layer of the cloud computing application. Where workload is not part of the definition's concepts, a concept such as demand, load, or problem size is considered. Some rarely used concepts are Service Level Agreements (SLAs), time, and performance. All other concepts shown in Tab. 2 occur only once.

As evidenced by the most used concepts in the found scalability definitions, most definitions characterize scalability as the handling of increasing workload by allocating more resources to the system. We therefore observe a general consensus regarding the definition of scalability among the investigated authors. However, different notions of scalability are described in the examined literature, mainly the differentiation of horizontal and vertical scaling. The former meaning the addition of computing nodes to the system and the latter the addition of computing power to a single node. Siklósi states that horizontal scaling is used in cloud computing environments [26]. Another differentiation of scalability definitions is proposed by Herbst et al., a group of authors that occurs three times in the selected sources for the SLR [13, 15, 20]. Herbst et al. define platform and application scalability, a definition for the PaaS and SaaS layer, respectively. In their understanding application scalability *"means that the application maintains its performance goals/SLAs even when its workload increases (up to a certain workload bound)"* [20]. Consequently, *"platform scalability is the ability of the execution platform to provide as many (additional) resources as needed (or explicitly requested) by an application"* [20]. In a later work, the two definitions are combined into a single definition: *"Scalability is the ability of the system to sustain increasing workloads by making use of additional resources"* [15]. Similarly, but with a focus on cloud layers, the CloudScale EU project [9] defines scalability for an arbitrary layer as *"the ability of the layer to sustain changing workloads while fulfilling its SLA, potentially by consuming a higher/lower quantity of lower layer services"* [9]. CloudScale introduces metrics as well, one related to service usage and another related to operational cost. Another metric is proposed by Tsai et al., based on the ratio of performance to used resources [29]. This metric is mathematically defined and is used in practice by Klaesson to evaluate a real cloud environment [19].

Table 2: Overview of scalability concepts

Reference	Workload	Resources	Performance	SLA/SLO	Load	Problem Size	Time	Capacity	Demand	Services	CPUs	Cost	Node	Quality
[20]	X	X		X										
[9]	X			X						X				
[13]	X	X		X										
[19]		X			X	X	X				X			
[26]			X		X			X	X			X	X	
[15]	X	X							X					
[24]						X								
[11]	X	X												
[17]	X	X												X
[29]	X	X	X				X							
[2]	X		X					X						

In principle, Tsai et al. calculate the resource consumption of different kinds of resources over the usage time, from which they calculate the *"performance to resource ratio"*. From this ratio, they calculate the performance change under different workloads, indicating the scalability of the system. The performance variance calculated from the performance changes serves as a second indicator of scalability [29].

Autili et al. provide a scalability definition where performance is taken into consideration but where time is considered as an integral part as well [2]. Autili et al. define scalability as a property that *"represents the capability of increasing the computing capacity of service provider's computer system and system's ability to process more users' requests, operations or transactions in a given time interval"*, going on to state that *"it is also related to performance"* [2]. The integration of time and performance into a scalability definition is unique in the pool of examined sources, though workload is omitted and instead represented by the inclusion of "users' requests", "operations", or "transactions".

3.6.2 Elasticity Definitions and Metrics

The concepts used to describe elasticity in the found definitions differ from the concepts describing scalability, as depicted in Table 3. The concept of resources is also commonly used to characterize the elasticity of a system, as is workload. However, demand appears more often in elasticity definitions. It is unclear in all definitions how demand differs from workload. Most definitions indicate that elasticity is based on how quickly a cloud environment is able to adapt. Therefore, time is a concept that is used in almost all examined definitions for elasticity.

Under- and Overprovisioned States.

When analyzing elasticity definitions and metrics, Herbst et al. stand out again among the multitude of definitions. No other group of authors appears more often in our SLR and is referenced more often within the selected sources of the review. Over time, Herbst et al. have published two definitions of elasticity. The first is based on their definition of scalability, adding a dynamic component: *"Elasticity is the ability of a software system to dynamically scale the amount of the resources it provides to clients as their workloads increase or de- crease"* [20, 13]. They also state that elastictiy *"consists of the temporal and quantitative properties of runtime resource provisioning and unprovisioning"*, stressing that mainly the change in resources and the reaction time is of importance, which indicates the importance of speed in an elastic system. Therefore, the metrics used to evaluate

Table 3: Overview of elasticity concepts

Reference	Demand	Resources	Time	Workload	Autonomy	Cost	Capabilities	Quality	Efficiency	Amount of Services	Load	SLA/SLO	Capacity
[20]	X	X	X	X									
[9]			X	X						X		X	
[13]	X	X	X	X									
[25]	X	X	X	X									
[15]	X	X	X	X	X								
[24]	X	X	X		X								
[4]	X		X	X	X		X						
[10]		X				X		X					
[11]	X	X	X	X									
[17]		X	X	X		X		X					
[27]	X	X	X			X							
[1]	X	X	X	X									
[28]	X	X	X								X		
[8]	X	X							X				
[7]	X	X							X				
[2]	X	X	X		X								X
[5]	X		X				X						

elasticity in this first definition are the effect of reconfiguration, i.e., the amount of resources added or removed, as well as the temporal distribution of reconfiguration points, i.e., how often adjustments to the assigned resources are made. Furthermore, the reaction time measuring how quickly the reconfiguration was performed is taken into account. These (sub) metrics are then combined to a single metric used to evaluate the elasticity of a system. Herbst et al. do not elaborate their metric further. In a later work, Herbst et al. alter their elasticity definition [15], stressing autonomous provisioning of resources and close matching of provided resources to demand: *"Elasticity is the degree to which a system is able to adapt to workload changes by provisioning and deprovisioning resources in an autonomic manner, such that at each point in time the available resources match the current demand as closely as possible"* [15]. To evaluate the matching of provided resources to actual demand, Herbst et al. investigate when the system has less resources than needed, i.e., it is underprovisioned, and when more resources are allocated than actually necessary, i.e., it is overprovisioned. In this later work, no single metric is proposed anymore.

A similar approach is taken by Islam et al. to evaluate elasticity [17]. They too use under- and overprovisioned states for their elasticity metrics but interpret these states as penalties. Therefore, the main concept in Islam et al.'s metric are the penalties for over- and underprovisioning, measured by cost. Because the duration of (de-)provisioning is important, time plays a significant role in this metric. Moreover, Islam et al. define their metric with high mathematical detail.

Tinnefeld et al. focus their definition on quick (ideally immediate) resource adaption and matching the demand as closely as possible to gain cost efficiency [27, 28]. They then adopt Islam et al.'s metrics with modified naming; the calculations however remain the same.

The work of Almeida et al. [1] is based on the aforementioned works by Herbst et al. and Islam et al. They too use an under- and overprovisioning penalty but define these penalties using their own metrics. These metrics are the expected response time, i.e., the time that a query may take to respond according to an SLA, and the violated execution time, which is the total time a query takes while violating the SLA. The metrics are then used to calculate the two penalties, from which in turn a single metric for elasticity evaluation is calculated. It is possible to adjust the impact of the underprovisioning penalty by the use of a factor. According to Almeida et al., the underprovisioning penalty is more interesting to cloud consumers than the overprovisioning penalty, which is supposedly more interesting for cloud providers. Therefore, Almeida et al.'s metric is adjustable to be more or less consumer-centric.

Formal Languages.

Two of the examined works present formal languages for controlling elasticity. Bersani et al. develop a temporal logic language called $CLTL^t(D)$ to formally describe elasticity [4]. While they only imprecisely define when a system is elastic and when it is not, they formally define properties of elastic cloud systems using $CLTL^t(D)$. Some examples are "eagerness", which is the reaction time to adjust after a load change, "sensitivity" is a minimum demand increase threshold that triggers resource adaptation, and "oscillation", which happens in a system which changes the resource allocation while under constant load. These properties simplify discussions about the behavior of cloud environments because they are precisely defined.

Copil et al. follow a different approach with a language for controlling cloud elasticity [10]. They first define elasticity as *"the capacity of an application to change or be changed according to the context in which it resides"* [10]. Therefore, elasticity is not limited to dynamic resource allocation but can also be adjusted in terms of cost and quality. Copil et al. accordingly model the state of cloud environments via three dimensions: resources, cost, and quality. Cloud systems should then be able to switch between different states in order to provide the best possible quality at the lowest possible cost while fulfilling resource demands. The language defined by Copil et al. allows to describe such states and to define custom quality parameters depending on the application. Examples for possible metrics to measure the parameters are given but their calculation lacks detail. In summary, these sources additionally consider cost and quality of service and do not solely focus on quick resource adaptation.

Other Approaches.

Dhingra [11] bases his elasticity definition on the ones of Herbst et al. and Islam et al. Interestingly, he does not utilize the notion of under- and overprovisioned states. His goal is to define an elasticity metric for an elasticity IaaS framework. The basis of his elasticity metric is another metric called variability, describing how often workload imposed on a system is larger than a fixed threshold, i.e., "significant" [11]. He then defines elasticity by the closeness of resources matching to actual demand while the variability is significant. Dhingra's definitions are mathematically detailed and have been used in practice. The closeness of the resource matching is also important in Herbst et al.'a metric, showing the relation between their work and Dhingra's.

Shawky et al. define elasticity as the ability of a system to *"expand and contract overtime[sic] in response to users' demands"* [25]. Based on their own elasticity definition, NIST's rapid elasticity, and elasticity as defined in physics, they derive a metric to evaluate a cloud system's elasticity. In physics, elasticity is defined by the ratio of two components: the stress and the strain on a material [25]. Shawky et al. transfer this to cloud environments by defining the stress on the cloud as the ratio between demanded and allocated computing capacity. They then define the strain as the rela-

Table 4: Overview of efficiency concepts

Reference	Resources	Work	Number of Processors	Speedup	Power consumption	Performance	Accuracy	Completeness	Operations	Time
[20]	X									
[13]	X									
[15]	X	X								
[29]			X	X						
[8]					X	X				X
[7]					X	X				X
[2]	X						X	X		
[5]									X	X

tive change in bandwidths multiplied by the time needed for the allocation. Again, time is important in this metric: the longer the allocation takes, the less elastic the system.

Schulz [24] defines elasticity as the frequent and quick adjustment to demand. Ideally, this adjustment results in a system in which provided resources perfectly match demand all the time. Schulz' work considers the cloud consumers' view for defining his metrics. He states that not elasticity itself but key performance indicators (KPIs) of the system are important to cloud consumers, e.g., response time, which depend on elasticity and vice versa. To evaluate the elasticity of a cloud system, Schulz proposes metrics that can be used in SLAs: the workload increase per time period and the adjustment delay that cloud consumers are willing to accept.

Chen et al. define the term "elastic scalability" in two works [8, 7], standing for another elasticity definition. They define elasticity as the ability of a system to adapt to demand and consider computational efficiency and energy efficiency. Especially the inclusion of energy efficiency as a parameter of elasticity is novel in comparison to others.

CloudScale's definition of elasticity [9] is based on the definition by Herbst et al. [15] but adds the concept of a layer consuming services from an underlying layer.

3.6.3 Efficiency Definitions and Metrics

Efficiency concepts vary depending on the targeted resource. For example, we found terms such as computational efficiency, power efficiency, and user efficiency.

Table 4 shows the wide spread of used concepts. The most used concept is resources, similar to scalability and elasticity. Power consumption and performance occur twice but in definitions by the same author. All other concepts only occur once in the examined material.

For efficiency definitions, Herbst et al. are again featured prominently in the SLR. In early works, they define efficiency as a measure of how well an application uses provided resources [20, 13]. In a later definition, they adapt their definition to *"the amount of resources consumed for processing a given amount of work"* [15]. In the latter definition, they stress that an efficiency definition must capture all resources and not just those that are subject to adaptation, i.e., the overhead needs to be accounted for as well.

Chen et al. define power efficiency for the context of cloud computing [8, 7]. Their definition evaluates how well power consumed by a system is used over time. As a metric, they propose to divide performance by power, performance being measured by a benchmark and power being measured as the average power consumption during benchmarking.

Autili et al. define user efficiency as the ratio of used resources *"to the accuracy and completeness with which the users achieve their goals"* [2]. They define an associated metric including a value range and methodology for value calculation. Their metric is essentially a measure of usability and not performance of the cloud environment.

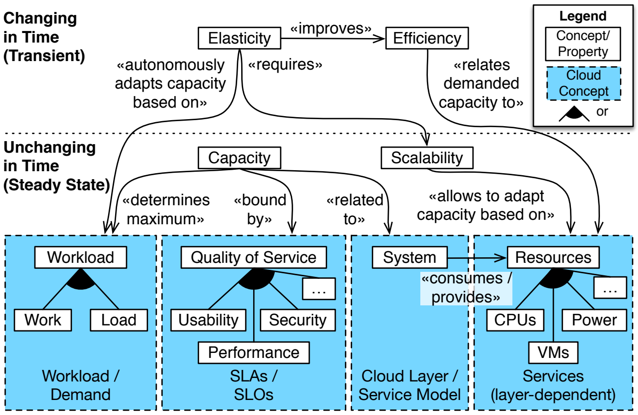

Figure 3: Summary of findings.

In a report on cloud computing, the US Defense Science Board defines computational efficiency as *"achieved operations compared to peak operations per second"* [5]. Unfortunately, they do not further elaborate this metric.

Tsai et al. [29] use a typical definition of efficiency that is based on Amdahl's Law: Speedup divided by the number of processors. They do not relate or specialize this definition to cloud computing environments.

3.6.4 Impact Analysis

As a measure for definition impact, we also counted cross-references within selected sources. The raw data for this analysis can again be found on our website (cf. Sec. 3).

4. REPORTING PHASE

In this section, we interpret the results from data synthesis and recommend definitions and metrics most fitting for cloud computing environments. Our main idea is that we can recommend definitions based on the most frequently used concepts, thus, suggesting a consensus of our 20 sources. In doing so, we will find out that all properties commonly use the concept of a maximum workload bound by the quality of service of a system. Therefore, we subsume the use of this combination of concepts in the capacity property (see Def. 5). This shared relation among all properties to capacity is a novel insight and a contribution of this paper.

Definition 5 (Capacity (General)). *Capacity is the maximum workload a system can handle as bound by its quality of service.*

Figure 3 summarizes our findings for defining scalability, elasticity, and efficiency. At the top, Fig. 3 relates the properties with each other. Elasticity requires scalability because a system that cannot scale is inherently inelastic [15]. Elasticity also improves the efficiency of a system because it typically reduces operational cost, power consumption, etc. [15]. We also distinguish properties unchanging in time (thus, allowing for steady state analyses) and properties changing in time (thus, requiring transient analyses). In doing so, we especially account for this fundamental difference between scalability (time occurred in 18%) and elasticity (time occurred in 82%).

At the bottom, Fig. 3 connects the properties to used concepts. Dashed boxes give cloud computing concepts (at the bottom) fitting to the included concepts. We use these concepts at the bottom to tie general definitions to the cloud computing domain. For example, for capacity we get Def. 1.

We report on each property based on Fig. 3 in the next subsections. In each report, we particularly propose a property definition tied to cloud computing – analogous to capacity – and point to potentially relevant metrics.

4.1 Scalability Report

Definition. Our data synthesis indicates that scalability is a well-established property used to describe the behavior of all types of distributed systems, not just cloud computing. During the SLR, we encountered references to older definitions of scalability, e.g., by Woodside from 2001 [31] and Hill from 1990 [16]. Tsai et al. examine scalability definitions from 1993 to 2005 and then derive a scalability definition for the cloud computing age [29]. This established understanding of scalability is particularly a reason that found scalability definitions often have common concepts.

The scalability definitions that we found in our SLR commonly use the concepts workload (72%) and resources (63%). The terms load and demand are used synonymously to workload, thus, increasing the concept workload to 91% uses. The concepts performance, SLA, and quality all bind system capacity (see Def. 5). Summarized like that, the concept capacity occurs in 63% of the definitions. Only Tsai et al. stress that scaling needs to be quick in cloud computing (speed); however, such concepts of time are more common to elasticity definitions (see next section).

A definition that is more tailored to cloud computing comes from the CloudScale project where used concepts include SLAs and services consumed by some upper cloud layer (e.g., the SaaS layer) from a lower layer (e.g., the PaaS layer) [9]. Also Herbst et al.'s definition uses the concept of SLAs and further describes the relation to elasticity. The concept of services is a specialization of resources, better fitting to the service-oriented cloud computing paradigm.

Based on these observations, we propose Def. 2 as a scalability definition tied to cloud computing.

Metrics. Considering the found metrics, only the metric by Tsai et al. [29] is evaluated in a practical setting [19]. The other metrics still lack evaluations [9, 26], thus leaving open whether they can be practically applied. Although Tsai et al.'s metric is based on older metrics, they adapted it with a focus on SaaS, so it fits cloud computing well. Found scalability metrics are generally not restricted to a particular role.

4.2 Elasticity Report

Definition. Contrary to scalability, elasticity definitions emerged together with cloud computing. The NIST defines "rapid elasticity" as essential part of cloud computing [23], pointing to this close connection.

Examining the found elasticity definitions, the differences to scalability are apparent in the concepts that are most often used. For elasticity, these concepts are demand/workload (together 94%), resources (82%), and time (82%). There is no explicit differentiation between demand and workload in any source; however, the term demand points to the service-oriented nature of cloud computing where demand is a common concept [23]. Similar to scalability, concepts like quality, SLA, and capabilities all bind capacity (36% in sum); but for elasticity capacity can dynamically vary over time. The concept cost is typically used to describe a benefit of elastic systems; however, is seldom (18%) used in definitions, indicating that cost is unessential for defining elasticity.

Considering the cross-references for elasticity property definitions within the selected sources, Herbst et al. and Islam et al. are referenced most often (not counting self-references). This referencing indicates that these definitions are more established among cloud researchers. In comparison, Herbst et al.'s definition stresses the demand-driven nature while Islam et al. take more concepts into account (they consider cost and quality of service). Both definitions stress that time-related concepts such as speed and autonomy are essential for a system to be elastic. Autonomy not only covers automatic scaling but also manual resource adaptations with a well-defined process, thus, allowing to calculate a maximum adaptation time [15]. If this time is short enough, a system appears elastic.

Based on these observations, we propose Def. 3 as elasticity definition (using cloud computing terms).

Metrics. Both Herbst et al. [13, 15, 20] and Islam et al. [17] define fitting metrics for their elasticity definitions. Islam et al.'s metric is the most referenced. It is quoted by Dhingra [11] and adopted by Tinnefeld et al. [27, 28] and Almeida et al. [1]. To calculate the value of their metric, they use a so-called under- and overprovisioning penalty. The metric considers cost and quality of service with the possibility for the user to add custom quality metrics to the calculation. Herbst et al. do not consider cost because cost is not a part of their elasticity definition. They also propose *several* metrics for elasticity, all based on under- and overprovisioned states. Both, Herbst et al.'s and Islam et al's, metrics can universally be applied by cloud consumers and cloud providers.

Other examined metrics do not reach the same maturity and universality of Herbst et al. and Islam et al. For example, Shawky et al. [25] use the bandwidth of the system and computing capacity, represented by the assigned CPU units and number of cores. Depending on the cloud layer, these hardware resources may be unavailable to the consumer, making the metric unusable from a cloud consumer standpoint. Dhingra's elasticity metric [11] is tailored to IaaS and his test environment, therefore not universally applicable to cloud computing in general.

4.3 Efficiency Report

Definition. Unlike for scalability and elasticity, found efficiency definitions relate to different resources, e.g., cost and power consumption. This observation explains that we found only resources and services as a commonly used efficiency concept (in sum, 50% of found definitions). Only Herbst et al. [15] account for such different resources: efficiency of a system *"expresses the amount of resources consumed for processing a given amount of work"* [15]; the lower this amount, the higher the efficiency of the system. Depending on the targeted resource, the "amount of resources" can relate to cost, power consumption, etc.

Efficiency definitions relate the amount of consumed resources to demanded work (12.5%), e.g., to calculate speedup factors (12.5%). These concepts also fit well to the capacity concept. Capacity is typically calculated over time (37.5%).

Based on these observations, we propose Def. 4.

Metrics. Existing efficiency metrics measure the "amount of lower-layer services" over time based on the aspect of interest. Found definitions focus on single cloud layers, e.g., power consumption [8, 7] of IaaS environments and usability [2] of SaaS applications. No found metric suggests to measure operation cost, which would universally quantify the "amount of lower-layer services" independent of cloud role and layer.

4.4 Key Insights

Based on quantifying the frequency of used concepts, we derived definitions that suggest a consensus of investigated sources. By detecting workload, quality of service, and system as common patterns within each definition, we were able to extract capacity as a dedicated property. Capacity allowed us to derive concise definitions that directly focus on core concepts. Identified, cloud-specific concepts (cf. Fig. 3) allowed us to tie our definitions to the cloud computing domain. Metrics are specific to cloud layers and to who measures and/or uses such measurements (cf. Tab. 1). In summary, our results lower the risk of neglecting important concepts, e.g., in service level objective negotiations and for metrics.

5. THREATS TO VALIDITY

This section describes our main threats to validity, classified in [30]: (1) conclusion validity, (2) internal validity, (3) construct validity, and (4) external validity.

Conclusion validity is the degree to which conclusions regarding the relationship between investigated sources and extracted data are correct. For us, the main threat to this validity is "low statistical power" [30] because we restricted our investigations to a limited amount of sources, e.g., due to the time frame and only using Google Scholar. Kitchenham also recommends other search databases such as ACM Digital Library, Citeseer library, IEEExplore, ScienceDirect, and SpringerLink [18]. We also conducted test searches in these databases. Our results indicated that Google Scholar covers the results from these databases as well.

Internal validity is the degree to which conclusions regarding the relationship between investigated sources and extracted data are causal (given an observed relationship). A threat to the internal validity is "maturation" [30], which describes a negative effect, e.g., tiredness, on subjects during an experiment. During the SLR, we performed repetitive tasks such as filtering the results and extracting the data. We countered tiring effects by the use of standardized forms and double-checking after a resting period.

Construct validity is the degree to which conclusions about the found concepts are correct. One threat that falls in this category is our subsumption of different concepts under the single term capacity. We did so to unify several found concepts under a common term. Moreover, capacity allowed us to have a single, precise concept used within our definition of properties. Found definitions that did not use this concept typically suffered from paraphrasing capacity. Our definitions are accordingly only a variation of possible definitions. Still, we derived our definitions by quantifying common concepts fitting to our context of cloud computing.

We consolidated definitions based on most common concepts, involving potential threats. For example, empirically evaluated concepts may be higher-ranked in future work.

External validity is the degree to which internally valid results can be generalized. An example of such a threat is the "interaction of setting and treatment" [30]. The setting refers to our SLR, being different from the setting that the results should be applied to. Therefore, the targeted audience (cloud providers, cloud consumers, software architects) has a different background in knowledge compared to the field from which the examined sources originate. To conquer this threat, we explicitly added the term "cloud computing" to our search phrases, expecting that the target audience is then better matched by our search results.

6. RELATED WORK

We already cover most related work in our systematic literature review (SLR). Still, we want to highlight especially the work of Duboc [12] and Li et al. [21] because they also focus on SLRs of cloud-related properties and metrics.

In her PhD thesis [12], Duboc analyzes scalability definitions in an SLR. Therefore, her work served us as a good starting point for such reviews, especially due to her focus on scalability. However, she focuses only on scalability and on software in general. She therefore provides too general concepts that neglect cloud computing characteristics such as rapid elasticity.

Li et al. [21] derive a catalogue of metrics important for cloud computing based on an SLR. Their catalogue is a good source for further metrics and metric classifications, nicely reflecting the state-of-the-art in this area. They also identify capacity as an important property to be considered. However, their focus is not on defining cloud properties such as elasticity, therefore, leaving open which concrete concepts are targeted by surveyed metrics. Relating their catalogue to our properties and concepts is accordingly a promising future work.

In one of our own previous papers [3][3], we also have the goal of systematically deriving metrics for scalability, elasticity, and efficiency. We provide an additional source for several metrics tied to cloud computing that is based on the goal-question-metric approach. A goal-question-metric approach is a top-down approach (deriving metrics) while SLRs are a bottom-up approach (starting with existing metrics) for exhaustive investigations of existing literature.

7. CONCLUSION

In this paper, we conduct a systematic literature review to recommend scalability, elasticity, and efficiency definitions and metrics for the cloud computing context. Based on our findings, we especially highlight commonly used concepts and propose novel definitions tied to cloud computing.

Software architects can use our metric recommendations to analyze the quality of cloud applications and to derive new metrics based on the concepts we highlight. Cloud providers and cloud consumers can use these concepts as common vocabulary and specify service level objectives based on our metric suggestions, without missing important concepts.

For future work, we suggest to externally execute our SLR, thus, analyzing its reproducibility. Such re-executions can also vary parameter of our SLR to cope with some of our threats to validity. For example, different search databases, more synonyms, and a larger time frame can be considered. Such efforts can strengthen our results and/or add further concepts to be considered.

[3] As it was published after we executed our SLR, our publication [3] did not go into our found literature sources.

8. REFERENCES

[1] R. Almeida, F. Sousa, S. Lifschitz, and J. Machado. On defining metrics for elasticity of cloud databases. In *Simpósio Brasileiro de Banco de Dados - SBBD 2013*, number i, pages 1–6, 2013.

[2] M. Autili, D. Di Ruscio, P. Inverardi, M. Tivoli, D. Athanasopoulos, A. Zarras, P. Vassiliadis, J. Lockerbie, N. Maiden, A. Bertolino, G. De Angelis, A. Ben Amida, D. Silingas, R. Bartkeviciu, and Y. Ngoko. CHOReOS Dynamic Development Model Definition (D2. 1). Technical report, 2011.

[3] M. Becker, S. Lehrig, and S. Becker. Systematically deriving quality metrics for cloud computing systems. In *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering*, ICPE '15, pages 169–174, New York, NY, USA, 2015. ACM.

[4] M. M. Bersani, D. Bianculli, S. Dustdar, A. Gambi, C. Ghezzi, and S. Krstić. Towards the formalization of properties of cloud-based elastic systems. In *6th International Workshop on Principles of Engineering Service-Oriented and Cloud Systems*, PESOS 2014, pages 38–47, New York, NY, USA, 2014. ACM.

[5] D. S. Board. Cyber Security and Reliability in a Digital Cloud. Technical Report January, 2013.

[6] P. Brereton, B. a. Kitchenham, D. Budgen, M. Turner, and M. Khalil. Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software*, 80(4):571–583, Apr. 2007.

[7] Q. Chen. Towards energy-aware VM scheduling in IaaS clouds through empirical studies. Master's thesis, University of Amsterdam, 2011.

[8] Q. Chen, P. Grosso, K. V. D. Veldt, C. D. Laat, R. Hofman, and H. Bal. Profiling Energy Consumption of VMs for Green Cloud Computing. In *2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, pages 768–775. IEEE, Dec. 2011.

[9] CloudScale EU project. Cloud Scale: Scalability Management for Cloud Computing D1.1. Technical Report 317704, SINTEF ICT, 2013.

[10] G. Copil, D. Moldovan, H.-L. Truong, and S. Dustdar. SYBL: An Extensible Language for Controlling Elasticity in Cloud Applications. In *13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing*, pages 112–119. Ieee, May 2013.

[11] M. Dhingra. Elasticity in IaaS Cloud, preserving performance SLAs. Master's thesis, Indian Institute of Science, 2014.

[12] L. Duboc. *A framework for the Characterization and Analysis of Software Systems Scalability*. Phd thesis, University College London, 2009.

[13] N. R. Herbst. Quantifying the Impact of Platform Configuration Space for Elasticity Benchmarking. Study thesis, Karlsruhe Institute of Technology, 2011.

[14] N. R. Herbst, S. Kounev, and R. Reussner. Elasticity in Cloud Computing: What It Is, and What It Is Not. In *Proceedings of the 10th International Conference on Autonomic Computing (ICAC 2013), San Jose, CA, June 24–28*, 2013.

[15] N. R. Herbst, S. Kounev, and R. Reussner. Elasticity in Cloud Computing: What It Is, and What It Is Not. *Proceedings of the 10th international conference on Autonomic computing - ICAC '13*, 2013.

[16] M. Hill. What is scalability? *ACM SIGARCH Computer Architecture News*, pages 18–21, 1990.

[17] S. Islam, K. Lee, A. Fekete, and A. Liu. How a consumer can measure elasticity for cloud platforms. In *ICPE '12 Proceedings of the 3rd ACM/SPEC International Conference on Performance Engineering*, pages 85–96, New York, New York, USA, 2012. ACM Press.

[18] B. Kitchenham and S. Charters. Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE 2007-001, Keele University and Durham University, 2007.

[19] P. Klaesson. Building a scalable social game server. Master's thesis, Chalmers University of Technology, 2013.

[20] M. Kuperberg, N. Herbst, J. V. Kistowski, and R. Reussner. Defining and quantifying elasticity of resources in cloud computing and scalable platforms. 2011.

[21] Z. Li, L. O'Brien, H. Zhang, and R. Cai. On a catalogue of metrics for evaluating commercial cloud services. In *13th International Conference on Grid Computing*, GRID '12, pages 164–173, Washington, DC, USA, 2012. IEEE Computer Society.

[22] F. Lui. NIST Cloud Computing Reference Architecture: Recommendations of the National Institute of Standards and Technology. 2011.

[23] P. Mell and T. Grance. The NIST definition of cloud computing. *NIST Special Publication*, 145(6):7, 2011.

[24] F. Schulz. Elasticity in Service Level Agreements. In *International Conference on Systems, Man, and Cybernetics*, pages 4092–4097. Ieee, Oct. 2013.

[25] D. M. Shawky and A. F. Ali. Defining a measure of cloud computing elasticity. In *Systems and Computer Science (ICSCS), 2012 1st International Conference on*, pages 1–5, Aug 2012.

[26] Z. Siklósi. Dynamically Scalable Applications in Cloud Environment. Master's thesis, Budapest University of Technology and Economics, 2013.

[27] C. Tinnefeld, D. Taschik, and H. Plattner. Providing high-availability and elasticity for an in-memory database system with ramcloud. In *GI-Jahrestagung*, pages 472–486, 2013.

[28] C. Tinnefeld, D. Taschik, and H. Plattner. Quantifying the elasticity of a database management system. In *DBKDA*, 2014.

[29] W.-T. Tsai, Y. Huang, and Q. Shao. Testing the scalability of SaaS applications. In *2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA)*, pages 1–4. Ieee, Dec. 2011.

[30] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[31] M. Woodside. Scalability metrics and analysis of mobile agent systems. *Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems*, 2001.